Towards Intelligent Monitoring System in Wire Arc Additive Manufacturing: A Surface Anomaly Detector on a Small Dataset

Yuxing Li

University of Wollongong

Haocheng Mu

University of Wollongong

Joseph Polden

University of Wollongong

Huijun Li

University of Wollongong

Lei Wang

University of Wollongong

Chunyang Xia (✉ chunyang_xia@sdu.edu.cn)

Institute of Materials Joining, Shandong University <https://orcid.org/0000-0002-3943-1321>

Zengxi Pan

University of Wollongong

Research Article

Keywords: Wire arc additive manufacturing, Intelligent monitoring system, Deep learning, Surface defects detection, YOLOv3

Posted Date: December 28th, 2021

DOI: <https://doi.org/10.21203/rs.3.rs-1165098/v1>

License: © ⓘ This work is licensed under a Creative Commons Attribution 4.0 International License.

[Read Full License](#)

Version of Record: A version of this preprint was published at The International Journal of Advanced Manufacturing Technology on March 28th, 2022. See the published version at <https://doi.org/10.1007/s00170-022-09076-5>.

Towards Intelligent Monitoring System in Wire Arc Additive Manufacturing: A Surface Anomaly Detector on a Small Dataset

Yuxing Li^a, Haocheng Mu^a, Joseph Polden^a, Huijun Li^a, Lei Wang^b, Chunyang Xia^{c,*}, Zengxi Pan^{a,*}

^a School of Mechanical, Materials, Mechatronic and Biomedical Engineering, University of Wollongong, Wollongong, NSW, 2522, Australia

^b School of Computing and Information Technology, University of Wollongong, Wollongong, NSW, 2522, Australia

^c MOE Key Laboratory for Liquid–Solid Structural Evolution & Processing of Materials (Ministry of Education), Institute of Materials Joining, Shandong University, Jinan, Shandong, 250061, China

Corresponding author: Chunyang Xia, yl452@uowmail.edu.au

14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56

Abstract

Rapid developments in artificial intelligence and image processing have presented many new opportunities for defect detection in manufacturing processes. In this work, an intelligent image processing system has been developed to monitor inter-layer deposition quality during a Wire Arc Additive Manufacturing (WAAM) process. Information produced from this system is to be used in conjunction with other quality monitoring systems to verify the quality of fabricated components. It is tailored to identify the presence of defects relating to lack-of-fusion and voids immediately after the deposition of a given layer. The image processing system is built upon the YOLOv3 architecture and through moderate changes on anchor settings, achieves 53% precision on surface anomaly detection and 100% accuracy in identifying the fabricated components' location, providing a prerequisite for high precision assessment of welding quality. The work presented in this paper presents an inter-layer vision-based defect monitoring system in WAAM and serves to highlight the feasibility of developing such intelligent computer vision systems for monitoring the WAAM process for defects.

Keywords

Wire arc additive manufacturing; Intelligent monitoring system; Deep learning; Surface defects detection; YOLOv3;

1. Introduction

Wire arc additive manufacturing (WAAM) is a direct energy deposition (DED) process suitable for the fabrication of large metal components due to its high deposition rate and relatively low manufacturing cost. WAAM is becoming a popular alternative to traditional manufacturing processes for materials such as the titanium alloys used in aerospace [1, 2] and nickel aluminium bronzes used in marine applications [3]. In the research domain, work has been published across a number of different aspects of the technology, such as process planning, material property improvements and process control. In pursuit of a fully automated WAAM system, Ding, et al. [4] proposed a systematic strategy featuring a series of algorithms, including 2D path planning [5], bead modelling [6] and multi-directional slicing algorithms [7]. The majority of material research tends to focus on material properties for various metals such as titanium alloys [8], aluminium alloys [9], steels [10] and nickel-based alloys [11]. A comprehensive review of material characteristics is provided by Wu, et al. [12]. Other research efforts investigate methods to improve geometric accuracy [13-15], surface roughness [16] and internal stress build-up [17] in deposited WAAM components. Nevertheless, many challenges remain to be addressed due to the overall complexity of the multi-faceted WAAM process [18], such as defect detection, manufacturing process optimization and automatic manufacturing.

Interior defect monitoring is an essential component in ensuring the quality of deposited WAAM components. Although some non-destructive methods, such as X-ray [19, 20], computed tomography [21], ultrasonic testing [22], and eddy current [23] are developed to detect interior defects, high equipment costs, along with complex detection and analysis processes restrict the utility of these methods. Image capture provides a convenient way to inspect deposition quality for a given layer during the WAAM process since many internal defects also exhibit visual

characteristics at the surface interface. For example, interior voids and lack of fusion defects within the component (two of the most commonly encountered defects associated with WAAM) can often be detected at the surface. These types of defect limit the commercial applications of WAAM, particularly the naval and aerospace sectors, which require components with a high degree of quality. With this in mind, it becomes necessary to develop reliable and intelligent monitoring systems for the WAAM processes in order to predict the existence of potential anomalies in the manufacturing process. These systems will be necessary for the eventual qualification of WAAM components for use in their relevant industries.

In the last decade, deep Convolution Neural Network (CNN) algorithms have pushed to the forefront for image classification and object detection tasks due to their accuracy and speed in extracting static 2D image features. In most scenarios, object detection can be considered for useful than image classification, though it is a more challenging task to compute. Computer vision research communities have invested significant effort into the development of these methods, and they have improved rapidly as a result. Object detectors such as region CNN (R-CNN) [24], single shot detector (SSD) [25], YOLO [26], and their derivatives Faster R-CNN [27], Cascade R-CNN [28], deconvolutional single shot detector (DSSD) [29] and YOLOv3 [30] have all demonstrated their capabilities in object detection with high accuracy. Researchers in the welding domain have begun applying various algorithms and object detectors to assist technicians in localisation tasks and identification of weld defects. Kim, et al. [31] applied extraction and noise removal algorithms to identify locations of occluded weld seams from arbitrary directions. Zou, et al. [32] proposed a mixed object detector combined with sequence multifeature combination network (SMFCN) and recurrent neural network (RNN) to detect continuous weld seams from images featuring a significant amount of noise. A simplified YOLOv3 model combined with image augmentation was selected for the task after evaluating performance against Faster R-CNN, YOLOv2, and YOLOv3 [33] methods. Another interesting application of YOLOv3 in the welding domain is presented by Dai, et al. [34], where positions and evaluated quality of spot welds on an automobile body were accurately detected. The lightweight network MobileNetV3 replaced the backbone network of YOLOv3 achieving a fast detection speed. Based on the previous research work, popular object detectors attained reliable detection results and fast detection speed both in beads location and defect detection in the last several years. Nevertheless, it can be argued that training a defect detector provides more utility than a position detector since defect boundary boxes also provide indication of the the location of weld seams [34, 35]. A good defect detector can be used to identify the locations of weld seams with some straightforward optimizations.

The dimension of the training dataset is a big issue limiting the development of intelligent systems on WAAM. Specifically, the large amount of images (e.g. $>10^4$ samples) required for training are not obtainable in real world manufacturing scenarios due to incomplete sensor setups and complicated manufacturing environments. To address these challenges, a small dataset is widely applied in welding science. Feng, et al. [36] used 487 data points to train a deep neural network (DNN) to predict solidification defects, which provides the feasibility to training a DNN model to predict defects with a small dataset. A small X-ray image dataset for weld defects diagnosis achieves more than 90% F1-score in a deep CNN [37]. Therefore, models that apply small datasets are good choices when big datasets are not available in real WAAM manufacturing.

This paper introduces an intelligent model based on the YOLOv3 algorithm. The goal is to utilise a small dataset to identify surface anomalies which occur during the WAAM process.

101 Following this introduction, the paper will introduce the methodology, related important
102 parameters, and algorithms of the YOLOv3. Then three WAAM components featuring a series of
103 defects are fabricated to provide datasets for the training of the model. To test the model, three
104 additional components featuring fewer defects are fabricated and then analysed by the model. To
105 improve the performance of the defect prediction model, some alterations in anchor settings are
106 attempted in order to develop a robustly trained model. The model with the best performance is
107 identified and discussed in the result section, followed by conclusions and a conceptual discussion
108 on future work.

109

110 **2. Methodology**

111 In the WAAM process, a fabricated component's quality is dependent on the quality of each
112 individual layer. Deposited weld beads are a product of carefully selected process parameters and
113 task-based parameters such as the infill pattern used overlapping distance between weld beads. With
114 recent developments in computer vision, a number of object detectors have demonstrated
115 capabilities which are likely applicable to the WAAM process. In this section, a brief introduction
116 to the classic object detectors is first provided. Then, as a representation of the newer variety of fast
117 and accurate object detectors, YOLOv3 is discussed in the context of defect detection in the
118 WAAM process.

119

120 **2.1 Related work**

121 R-CNN [24] is a typical two-stage region-based convolution network detector. R-CNN uses a
122 selective search to extract around 2000 region proposals. Each region proposal is warped into a
123 fixed pixel size (227×227) to be compatible with the CNN architecture, which then extracts
124 features for each region proposal. These features are fed into a linear Support Vector Machine
125 (SVM) to identify a specific class, followed by a linear regression model to predict and modify
126 detection windows. R-CNN achieves excellent object detection accuracy based on region proposal.
127 However, each region proposal's features need to be extracted by CNN and saved to disk, resulting
128 in slow detection speed and expensive training cost.

129 Fast R-CNN [38] is developed to address these drawbacks by sharing computation for
130 proposals. Fast R-CNN also makes use of selective search for region proposals, however, it feeds
131 the whole image and a set of object proposals into the CNN to obtain a convolutional feature map.
132 Then, a pooling layer processes each proposal to extract a corresponding fixed-length feature vector
133 from the feature map. After that, the fixed-sized feature map is transferred to a feature vector and
134 finally branched into two output layers to produce probability estimates and boundary box
135 positions. These changes help to share the convolution result from CNN, avoid repeated
136 computation for region proposals and therefore attain a speed approximately 25 times faster than
137 that of R-CNN. Faster R-CNN [27] focuses on optimizing the method of selecting region proposals.
138 Both R-CNN and fast R-CNN determine region proposal by selective search, while Faster R-CNN
139 trains a neural network, defined as region proposal network (RPN), to find all region proposals.
140 Firstly, Faster R-CNN sends input images to the CNN and obtains feature maps. Then a pre-trained
141 RPN extracts the proposals from feature maps and unites the proposal information with the feature
142 maps to obtain useful feature information. Linear classifiers and linear regression techniques are
143 then used to predict the presence of select objects. Based on Faster R-CNN, many region-based

algorithms have been developed to improve the performance and speed of object detection, for instance, Libra R-CNN [39] and Cascade R-CNN [28].

The YOLO series are powerful one-stage object detection algorithms which attempt to balance accuracy and speed. In 2016, YOLO [26] was first published. YOLO [26] divides a 224×224 image into a set of 7×7 grids and then uses a single convolutional network to output a $7 \times 7 \times 30$ tensor. This tensor composes two kinds of boundary box details and the probabilities for different object classes (20 classes in the PASCAL VOC dataset). Box details include x , y coordinates and w , h . The x and y coordinates represent the centre coordinates of the box relative to the bounds of the grid cell. The w and h denote the grid dimensions relative to the dimensions of the whole image. Finally, YOLO predicts whether an object exists in each individual grid. The limitations of YOLO are well known: Significant localization errors compared with Fast R-CNN; low recall compared with two-stage algorithms and difficulty in detecting small objects. YOLOv2 [40] was developed to solve these issues via (amongst other things) batch normalization, a high-resolution classifier and anchor box convolution. In addition, YOLOv2 proposed a new network architecture called Darknet-19, which has 19 convolutional layers and five max-pooling layers. Darknet-19 achieved 91.2% accuracy on ImageNet, which is higher than the 88% achieved by its YOLO backbone network. The output tensor of YOLOv2 is in the size of $13 \times 13 \times 125$ (five prediction boundary boxes, each boundary box has four box parameters, one confidence score and 20 class probabilities). YOLOv3 [30] further improved the prediction performance and small object detection capabilities. The backbone of YOLOv3 is modified to Darknet-53, which features 53 convolution layers and 23 residual blocks. Another innovation of YOLOv3 is using three different scaled output tensors (in the size of 13×13 , 26×26 , and 52×52) to predict different sized objects by three anchor boxes for each scaled output tensor. YOLOv3 has significant advantages both on accuracy and speed when the intersection of union (IOU, refer to 2.2.iv) is set to 0.5.

Figure 1 displays a performance comparison between popular object detectors. YOLOv3 has a clear advantage in detection speed while maintaining detection accuracy, which is an essential feature for online real-time detection. Thus, in this paper, we selected YOLOv3 for object detection tasks .

Figure 1 Comparison of the performance of popular object detectors.

2.2 YOLOv3 algorithm

2.2.i Backbone

Different from its two predecessors (namely, YOLO and YOLOv2) YOLOv3 implements a new network architecture for performing feature extraction. The backbone of YOLOv3, Darknet-53, is a hybrid architecture composed of Darknet-19 [40] and residual block [41]. In Darknet-53, 53 convolutional layers and 23 residual blocks are used to reduce the shape of the input image (416×416) to three feature maps: 13×13 , 26×26 , 52×52 . Then, YOLOv3 uses a feature pyramid network to merge the smaller feature map with the large feature map. Specifically, the feature map produced by Darknet-53 and five convolutional calculations are upsampled and concatenated with the 26×26 feature map. A $26 \times 26 \times 255$ output, including all class information and boundary box information, is obtained through a series of convolution layers. In similar fashion, the 26×26 feature map is merged with the 52×52 feature map. Thus, through these merging operations, three different outputs ($13 \times 13 \times 255$, $26 \times 26 \times 255$, $52 \times 52 \times 255$) are obtained. Figure 2 provides an overview of complete YOLOv3 architecture. Darknet-53 is regarded as a backbone to extract image feature maps.

Figure 2. Architecture of YOLOv3

2.2.ii Loss function

YOLOv3 uses mean squared error and binary cross entropy to evaluate errors between targets and outputs. Error values for each boundary box width and height use mean squared error, while other errors including the (x, y) coordinates of the bounding box, confidence scores for objects and non-objects and the probability on class identification all use binary cross entropy error [30]. In this paper, in order to facilitate algorithm optimization, we use mean squared error for box information $(x, y, \text{ and } w, h)$ and binary cross entropy error for other parameters. Equation (1) is the loss function used in our training:

$$\begin{aligned} \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^B 1_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right] &+ \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^B 1_{ij}^{\text{obj}} \left[(w_i - \hat{w}_i)^2 + \right. \\ &\left. (h_i - \hat{h}_i)^2 \right] &+ \sum_{i=0}^{S^2} \sum_{j=0}^B 1_{ij}^{\text{obj}} \left[\hat{C}_i \cdot \log C_i + (1 - \hat{C}_i) \cdot \log(1 - C_i) \right] &+ \\ \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^B 1_{ij}^{\text{noobj}} \left[\hat{C}_i \cdot \log C_i + (1 - \hat{C}_i) \cdot \log(1 - C_i) \right] &+ \sum_{i=0}^{S^2} 1_i^{\text{obj}} \sum_{c \in \text{classes}} \left[\hat{p}_i(c) \cdot \right. \\ &\left. \log p_i(c) + (1 - \hat{p}_i(c)) \cdot \log(1 - p_i(c)) \right] \end{aligned} \quad (1)$$

$$\begin{aligned} \text{loss} = & \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^B 1_{ij}^{\text{obj}} [(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2] \\ & + \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^B 1_{ij}^{\text{obj}} [(w_i - \hat{w}_i)^2 + (h_i - \hat{h}_i)^2] \\ & + \sum_{i=0}^{S^2} \sum_{j=0}^B 1_{ij}^{\text{obj}} [\hat{C}_i \cdot \log C_i + (1 - \hat{C}_i) \cdot \log(1 - C_i)] \\ & + \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^B 1_{ij}^{\text{noobj}} [\hat{C}_i \cdot \log C_i + (1 - \hat{C}_i) \cdot \log(1 - C_i)] \\ & + \sum_{i=0}^{S^2} 1_i^{\text{obj}} \sum_{c \in \text{classes}} [\hat{p}_i(c) \cdot \log p_i(c) + (1 - \hat{p}_i(c)) \cdot \log(1 - p_i(c))] \end{aligned} \quad (1)$$

The first two terms denote localization loss, terms 3 and 4 represent confidence losses, and the last term indicates the classification loss. λ_{coord} and λ_{noobj} are constants giving weights to the loss terms. We set $\lambda_{\text{coord}} = 1$ and $\lambda_{\text{noobj}} = 100$ in this paper. In YOLOv3, every grid cell receives three prediction boxes based on the sizes of anchor boxes, meaning that B is three. S is the number of grid cells, which is the feature map's shape (for 13×13 feature map, S is 13). 1_{ij}^{obj} and 1_{ij}^{noobj} state if this grid cell is responsible for the object or not. 1_{ij}^{obj} equals 1 if the (i, j) cell contains and be responsible for the object. Otherwise, 1_{ij}^{obj} equals 0. 1_{ij}^{noobj} is 1 if there is no object shown in the grid

cell (i, j) or the grid cell is not responsible for the object. Parameters such as x_i, y_i, w_i, h_i, C_i and $p_i(c)$ correspond to x, y coordinates, width and height of prediction boundary box, prediction confidence, and class probabilities. $\hat{x}_i, \hat{y}_i, \hat{w}_i, \hat{h}_i$ are the ground truth of the labeled boxes. $\hat{C}_i$ and $\hat{p}_i(c)$ are the confidence of the cell's responsibility (which is always 1) and class probability. This loss function tactfully manipulates all losses to a single equation, simplifying the individual optimization processes facing different loss functions.

2.2.iii Anchors

Anchor boxes [27] provide prior knowledge from ground truth boxes of the training dataset. They deduce the truth box's dimensions through a K-means algorithm. However, instead of Euclidean distance, the intersection of union (IOU) between boundary boxes and centroids are used as the optimization criterion. With this prior knowledge, the trained model predicts the sizes of boundary boxes, improving the model's training convergence rate. The pre-setting of anchor boxes to the model solves the issue of multi-scaled objects. In YOLOv3, nine anchor boxes with different dimensions are pre-set for training and sorted into three groups by size. For instance, three minimum-sized anchor boxes are used as the prior knowledge to predict three boxes for every grid cell in the 13×13 feature map. The predicted boundary box having the largest IOU with the ground truth box is regarded as the output boundary box. Similar predictions and calculations are designed for 26×26 and 52×52 feature maps, which are designed to precept middle- and small-sized objects, respectively. The nine anchors of YOLOv3 on the COCO dataset [42] are: $(10 \times 13), (16 \times 30), (33 \times 23), (30 \times 61), (62 \times 45), (59 \times 119), (116 \times 90), (156 \times 198), (373 \times 326)$.

2.2. iv Boundary box

Through the preset of anchor boxes, every prediction box acquires the best sizes via prior knowledge from anchor boxes. YOLOv3 bonds the anchor boxes' size and prediction boxes' size via Equation $b_w = A_w e^{t_w}$ (2) and $b_h = A_h e^{t_h}$

(3):

$$b_w = A_w e^{t_w} \quad (2)$$

$$b_h = A_h e^{t_h} \quad (3)$$

where b_w and b_h are the width and height of the final rescaled boundary box. t_w and t_h are the original width and height of the prediction, while A_w and A_h are the scaled anchor box's dimensions. Parameter A_w and A_h are scaled proportionally as the scaled ratio of the feature map ($1/32, 1/16, 1/8$ for $13 \times 13, 26 \times 26, 52 \times 52$ feature maps, respectively). As shown in Figure 3, the final prediction box's dimensions will be bonded with the anchor box's sizes and scaled into an object area. The centre of the boundary box is calculated by Equation $b_x = \sigma(t_x) + c_x$

(4) and $b_y = \sigma(t_y) + c_y$ (5):

$$b_x = \sigma(t_x) + c_x \quad (4)$$

$$b_y = \sigma(t_y) + c_y \quad (5)$$

where (t_x, t_y) is the original prediction of the boundary box central coordinates, and (c_x, c_y) is the left corner coordinates of the grid cell. b_x and b_y are the adjusted coordinates of the boundary box

251 centre. As the red point displayed in Figure 3, the coordinate of the grid cell is $(c_x, c_y) = (3, 2)$. σ is
 252 the sigmoid function, which is used to limit the prediction coordinates to absolute coordinates
 253 within $(0, 1)$, guaranteeing the centre of the boundary box stays within the grid cell.

Figure 3. Illustration of boundary box parameters

256 As previously described in the ‘Anchors’ section, every grid cell in a feature map receives
 257 three boundary boxes of different sizes. After the calculation (Equation $b_w = A_w e^{t_w}$
 258 (2) to $b_y = \sigma(t_y) + c_y$ (5)), YOLOv3 outputs four
 259 coordinate parameters (b_x, b_y, b_x, b_y) and one confidence parameter for each boundary box. To
 260 select the best and most reliable boundary box, two criteria are designed:

- 261 1. Confidence threshold, which is used to show the confidence that an object exists or not. The
 262 boundary box with a confidence lower than the preset threshold is discarded.
- 263 2. Non-maximum suppression (NMS) threshold: A technique used to remove redundant
 264 boundary boxes and is evaluated by an IOU (Equation $IOU =$
 265 $\frac{\text{intersection area of prediction box and ground truth box}}{\text{union area of prediction box and ground truth box}}$ (6)). Boundary boxes
 266 having an IOU larger than the NMS threshold will be removed, avoiding using multiple
 267 boundary boxes to indicate the same object.

$$IOU = \frac{\text{intersection area of prediction box and ground truth box}}{\text{union area of prediction box and ground truth box}} \quad (6)$$

269 3. Experimental Result of YOLOv3 on WAAM Dataset

270 3.1 Experimental setup

271 A conventional WAAM system is used to deposit three components for training and then three
 272 additional components for testing. The setup of the WAAM system is shown in Figure 4. It consists
 273 of a control PC (Figure 4 (a)), an ABB robotic system (Figure 4 (b), (d)), a Fronius welder (Figure 4
 274 (c), (e)), a weld process current and voltage collection system (Figure 4 (g)), and a national
 275 instrument USB 6008 board (Figure 4 (h)).

The control PC has software that generates code files to control robot movements and synchronize welding parameter settings. The shielding gas consists of 100% Argon. BOC 4043 aluminium MIG wire is used in these tests. Synergic welding control generates weld parameter sets, meaning that the welder automatically determines current and voltage values through pre-stored programs in the Fronius weld controller. In order to get enough defects, especially for the training dataset, the Fronius proprietary Cold Metal Transfer (CMT) welding process is used for the fabrication of the training components. For the test components, to simulate a realistic manufacturing scenario, the CMT-pulse process is used.

Figure 4. The setup of the WAAM system in this experiment. **(a)** PC controller. **(b)** **(d)** ABB robot system: IRC5 controller and IRB2600 robot. **(c)** **(e)** Fronius welder and weld torch. **(f)** actual workpiece. **(g)** self-build high-frequency current and voltage collection box. **(h)** NI USB 6008 board.

3.2 Dataset construction

Three different components, featuring a variety of geometric features, were designed for the training experiments. Three components (square-shaped, hexagon-shaped and star-shaped) are designed in order to generate potential defects when processing right angles, obtuse angles and acutely angled features. Figure 5 provides an illustration of the required welding paths. During the manufacturing process, neighbouring contours were welded in clockwise and anticlockwise directions, as displayed in Figure 5. Figure 5 also highlights the positions where defects are likely to appear. The primary defects which arise at these positions are surface voids because the distance between two vertexes is too large to be covered by the width of welding beads. Lack of fusion between weld beads is another defect encountered in the WAAM process, often generated when the overlap distance between weld paths is too wide.

Figure 5. Path planning for three components of different shapes: square, hexagon, and star.

Figure 6 shows the WAAM components after fabrication. During the manufacturing process, after the deposition of each layer, images of the deposited surface are taken with a camera mounted on the welding robot. Three components, each consisting of 30 deposited layers, were built (Figure 6 (a) - (c)) and images of each deposited layer were captured and cropped to be used in the training dataset. In total, 51 images are designated for use in the training dataset, whilst the remaining 17 images were used in the validation dataset. Since each image will contain a different number of anomalies, 1646 anomaly boxes are included in the training dataset and 602 anomaly boxes in the validation dataset. To increase the number of defects in the training set components, the width of the welding beads was purposefully decreased. On the other hand, appropriate wire feed speed and travel speeds were used for the fabrication of components in the test dataset (Figure 6 (d) - (f)). Each component in the test dataset consists of 12 welded layers, each of which has several accompanying images to go along with it. In total, 146 images are stored in the test dataset. The welding parameters, wire feed speed (WFS) and travel speed (TS) for each test component are shown in Table 1. Furthermore, each image includes two types of labels: component and anomaly. In general, each image features only one component label.

Figure 6. Three welding components demonstration. From (a) – (c) and (d) – (f) are square, hexagon, and star components, respectively.

Table 1. Components for test dataset's welding parameters

	Square (exterior to interior order)		Hexagon (exterior to interior order)		Star (exterior to interior order)	
Layer	WFS(m/min)	TS (m/min)	WFS (m/min)	TS (m/min)	WFS (m/min)	TS (m/min)
1	6 → 5 → 5 → 6	0.5	6 → 5 → 5 → 6	0.5	6 → 5 → 5 → 6	0.5
2	6 → 5 → 6 → 6	0.5	6 → 5 → 6 → 6	0.5	6 → 6 → 5 → 6	0.5
3	6 → 5 → 5 → 6	0.5	6 → 5 → 5 → 6	0.5	6 → 5 → 5 → 6	0.5
4	5 → 5.5 → 5.5 → 5	0.5	5 → 5.5 → 5.5 → 5	0.5	5 → 5.5 → 5.5 → 5	0.5
5	5 → 6 → 6 → 5	0.5	5 → 6 → 6 → 5	0.5	5.5 → 6 → 6 → 5.5	0.5
6	5.5 → 6 → 6 → 5.5	0.5	5 → 5.5 → 5.5 → 5	0.5	5.5 → 6 → 6 → 5.5	0.5
7	5.5 → 6 → 6 → 5.5	0.5	5 → 5.5 → 5.5 → 5	0.5	5.5 → 6 → 6 → 5.5	0.5
8	6 → 6.5 → 6.5 → 6	0.5	5.5 → 6 → 6 → 5.5	0.5	6 → 6.5 → 6.5 → 6	0.5
9	5.5 → 5.5 → 5.5 → 5.5	0.5	5 → 5.5 → 5.5 → 5	0.5	5.5 → 6 → 6 → 5.5	0.5
10	5.5 → 5.5 → 5.5 → 5.5	0.5	5 → 5 → 5 → 5	0.5	5.5 → 6 → 6 → 5.5	0.5
11	5 → 6 → 6 → 5	0.5	5 → 5 → 5 → 5	0.5	5.5 → 6 → 6 → 5.5	0.5
12	5.5 → 6 → 6 → 5.5	0.5	5.5 → 6 → 6 → 5.5	0.5	5.5 → 6 → 6 → 5.5	0.5

3.3 Anchors

An Anchor provides prior knowledge about the dimensions of objects. Redmon and Farhadi [30] determined that nine different anchor box dimensions obtain the best performance on the COCO dataset with 80 object classes. However, in our welding dataset, there are only two classes (anomaly and component) in the whole dataset. Furthermore, the dimensions of the anomaly and

component boxes are not similar to the objects in the COCO dataset. Thus, new anchor settings are required for our dataset.

K-means algorithm is used to discover clusters along with size information of the ground truth boundary boxes. The IOU between the anchor boxes is used to optimize cluster centroids. Similar to the anchor sets for the COCO dataset, we keep nine anchors for the training model.

We modified the original COCO dataset’s anchor setting. As the relative size of the anomaly inside the image is small, only the anchor settings for the small object anchors and the largest anchor size (used for component detection) are modified, as displayed in Table 2.

As shown in Table 2, “1-anchor-change” indicates that we only changed the largest anchor size used for component prediction. 2-anchor-change to 4-anchor-change modify anchors for small objects gradually to find out the best anchor setting for our dataset. Finally, 9-anchor-change means that all the ground-truth bounding boxes used during training are clustered into nine classes and completely replace the original anchor settings. To describe the modifications succinctly, we named the six different anchor settings from Group ‘A’ to ‘F’, as outlined in Table 2. It is worth mentioning that to determine the modified anchor box size, boxes in the training dataset are first clustered into nine clusters. Then the nine anchors without the largest anchor size are clustered into clusters of 1-3 respectively to obtain the changed anchor data for Group C to Group E.

Table 2. The modified anchor box settings.

Anchor	original anchors	1-anchor-change	2-anchor-change	3-anchor-change	4-anchor-change	9-anchor-change
1	(10, 13)	(10, 13)	(10, 13)	(10, 13)	(22, 14)	(14, 17)
2	(16, 30)	(16, 30)	(21, 20)	(15, 23)	(15, 23)	(23, 12)
3	(33, 23)	(33, 23)	(33, 23)	(26, 15)	(30, 17)	(12, 24)
4	(30, 61)	(30, 61)	(30, 61)	(30, 61)	(30, 61)	(20, 16)
5	(62, 45)	(62, 45)	(62, 45)	(62, 45)	(62, 45)	(17, 21)
6	(59, 119)	(59, 119)	(59, 119)	(59, 119)	(59, 119)	(32, 14)
7	(116, 90)	(116, 90)	(116, 90)	(116, 90)	(116, 90)	(19, 29)
8	(156, 198)	(156, 198)	(156, 198)	(156, 198)	(156, 198)	(28, 20)
9	(373, 326)	(350, 287)	(350, 287)	(350, 287)	(350, 287)	(350, 287)
Group	A	B	C	D	E	F

3.4 Training

The training process is performed in two steps: firstly, models are trained with only a small number of epochs and the best optimizer is then selected; secondly, the training epoch number is increased so as to compare the models obtained with different anchor settings. The model with the best performance is then selected.

This paper compared model performances from different optimizers, including Adadelata [43], RMSprop [44], SGD [45], and Adam [46]. To select the optimizer having rapid convergence, a large learning rate and a small epoch number are set in advance, which are 1e-3 and 64, respectively. Then the best optimizer is trained with a series of small learning rates. Several different random learning rates between 1e-5 to 1e-3 are compared. With the best optimizer and

359 learning rate, we increase the iteration number to 128 and compare the performances on different
 360 anchor settings. Lastly, we find out the best model by comparing the performances in the test
 361 dataset. Mean average precision (mAP) is used to evaluate the performance of each model, which is
 362 calculated by taking the average precision over all classes. In this study, both component object and
 363 anomaly class are calculated. It is defined as below:

$$364 \quad \text{mAP} = \frac{\sum \text{precision for all classes}}{\text{number of classes}} \quad (7)$$

365 4. Results

367 4.1 Training results

368 Figure 7. Training performances (a) total loss; (b) precision; (c) recall with 0.5 IOU; (d) recall with
 369 0.75 IOU on different optimizers: Adadelata, RMSprop, SGD, and Adam.
 370

371 To find the best optimizer for our training dataset, the number of epochs is first set to a small
 372 number (128) while the learning rate is set to a large value (1e-3). Optimizers tested are the
 373 aforementioned Adadelata, RMSprop, SGD and Adam methods. The training processes are shown in
 374 Figure 7. It is found that SGD performs poorly under these hyperparameter settings. The evaluation
 375 criteria such as precision and recall are poorly reflected on our dataset. Adadelata was found to
 376

377 outperform RMSprop. Both precision and recall show unstable performances during the training.
 378 Figure 8 is the model performance on the validation dataset after each epoch. Based on Figure 8,
 379 generally, the Adam optimizer performs better than the other three optimizers. Therefore, we select
 380 Adam as our optimizer in the subsequent experiments. After further tuning the learning rate, we
 381 found that the model performs best at the validation dataset when the learning rate is $1.26e-4$.

382
 383 Figure 8. Validation performances (a) f1 score; (b) mAP; (c) precision with 0.5 IOU; (d) recall with
 384 0.5 IOU on different optimizers: Adadelta, RMSprop, SGD, and Adam.

385
 386 As shown in Figure 8, the results on the validation dataset improve with increasing epoch
 387 number, which suggests that extending beyond 128 will further improve the prediction model. For
 388 further models' comparisons, 500 is set for the number of epochs to achieve reliable results. To
 389 determine the best anchor setting, we designed and trained a series of experiments and models to
 390 find out the best model for anomaly detection.

391 Firstly, a model with nine anchors replacement (Group F, as described in Table 2) is trained.
 392 For comparison, a model having the original anchor setting (Group A in Table 2) is trained. The
 393 performance of the validation dataset is shown in Figure 9. The model performance for Group A is
 394 always better than that of Group F, especially for the F1 score and the mAP criterion. It is mainly
 395 because the feature maps from YOLOv3 have three different sizes (13×13 , 26×26 , and 52×52),
 396 which are designed to identify objects of various sizes. However, sizes of the boundary boxes of the
 397 anomaly are always similar, which are located within the small object ranges. Therefore, if we
 398 replace all anchors with small object sizes, the feature maps from YOLOv3 will get confused when

399 detecting medium- and large-sized objects, performing poorly on our training dataset. Furthermore,
 400 the erratic precision for Group F indicates that the model is not stable and reliable for defect
 401 prediction. Thus, the anchor setting in Group F cannot be used to detect anomalies in our dataset.

Figure 9. The model performances with anchor setting on Group A and Group F: (a) F1 score; (b) mAP; (c) precision with 0.5 IOU; (d) recall with 0.5 IOU

Figure 10. Model performances on validation dataset of Group A, B, C, D, E, on (a) F1 score; (b) mAP; (c) precision with 0.5 IOU; (d) recall with 0.5 IOU.

Secondly, the performance of Group A is compared with Group B, C, D, and E. After 500 epochs training, as shown in Figure 10, Group E is not able to perform as well as Group A, while Group B, C, and D all have better performances than Group A. The comparison results of Group A, E, and F imply that too many anchor changes reduce the performance and stability of the model. Although Group B, C, and D all have better performance than Group A (based on F1 score and mAP), Group C performs more accurately on the anomaly prediction than the other two groups roughly within 500 epochs. On the other hand, Group C and D are not as stable as Group B, since evident ups and downs for Group C and D are presented in Figure 10. Due to the fluctuation in the performance of Group D, it is hard to compare its performance with Group B.

In conclusion, the model with Group B's anchor setting demonstrates the most stable performance. Anchor settings for Group C and D have obvious fluctuations. However, within 500 epochs, the performance of Group C is the best compared with Group B and D. Besides, as shown in Figure 10, when the epoch number is set to 500, the model's performance improves with the increase of epoch number, especially for the F1 score and mAP criterion. Group B and D have more obvious tendencies to improve performances at the end of the 500 epoch, while Group C has a faster convergence rate.

As discussed earlier, the best model's performance cannot be obtained within 500 epochs, especially for Groups B and D. Thus the epoch number is increased to 1000 for training. The

model's performance of anchor settings on Groups B, C, and D will be compared directly on images in the test dataset to determine the best anchor setting. The performance of the model with anchor settings for Groups A-D will be displayed and compared in the next section.

4.2 Model performance on test images

With the best learning rate mentioned above ($1.26e-4$), the four best models on anchor settings of Groups A, B, C, and D within 1000 epochs are selected and compared on the test dataset. To find out the correct confidence threshold and NMS threshold intervals, the performance on the validation dataset is compared in advance. Preliminary values for the confidence threshold varies from 0.3 to 1, and the box with confidence smaller than 0.3 is regarded as a negative sample. Through our comparison, high confidence thresholds (larger than 0.7) fail to detect the existence of anomalies. On the other hand, too large NMS thresholds keep boundary boxes with high intersection areas, resulting in redundant boundary boxes in predictions. Therefore, in this paper, confidence thresholds are set between 0.4 and 0.7, and NMS threshold is set as 0.05.

Table 3 reports the best model's performance when we set the confidence thresholds and the NMS threshold in the determined intervals. NMS threshold of 0.05 indicates that almost all redundant boundary boxes with intersection areas will be removed after the NMS algorithm. The original anchor setting (Group A) performs not as well as the other three groups, which is consistent with the results shown in Figure 10. Based on the results shown in Table 3, Groups B, C and D improve at least 5% on the mAP with respect to the original anchor setting. With the increase of the number of iterations (after 500 epochs), the performances of Groups B and D both achieve better results than that of Group C. Consistent with the previous discussion, Group C has a quick convergence rate, resulting in low improvements with the increase of epoch number, while Groups B and D improve their prediction results on both component and anomaly areas with the increased training of epoch number. Comparing with Groups B and C, Group D performs better both on component and anomaly identifications. Group D has a 5.8% improvement on the component identification compared with the original anchor setting (Group A) and an 8.9% improvement on anomaly prediction when the confidence threshold is set to 0.4. There is about a 7.5% improvement on Group D's mAP when the threshold is set at 0.4 and 0.5 compared with Group A. About 2% mAP rises on Group D relative to Group B and C from Table 3.

Table 3. The four best models (at epoch 988, 795, 942, 947, respectively) performance on the validation dataset.

	Confidence threshold	NMS threshold	Component	Anomaly	mAP
Origin anchors (Group A)	0.4	0.05	0.9412	0.4836	0.7124
	0.5	0.05	0.9412	0.4794	0.7103
	0.6	0.05	0.7647	0.4703	0.6175
	0.7	0.05	0.5294	0.4541	0.4918
1-anchor-change (Group B)	0.4	0.05	1.0000	0.5299	0.7650
	0.5	0.05	1.0000	0.5273	0.7637
	0.6	0.05	0.8235	0.5157	0.6696
	0.7	0.05	0.7059	0.4848	0.5954
2-anchor-change (Group C)	0.4	0.05	1.0000	0.5264	0.7632
	0.5	0.05	1.0000	0.5157	0.7579
	0.6	0.05	0.9412	0.4990	0.7201

	0.7	0.05	0.7647	0.4889	0.6268
3-anchor-change (Group D)	0.4	0.05	1.0000	0.5728	0.7864
	0.5	0.05	1.0000	0.5723	0.7862
	0.6	0.05	0.7647	0.5622	0.6635
	0.7	0.05	0.6471	0.5432	0.5952

462 Figure 11. Best model’s performance on three test components images at 0.5 confidence threshold,
463 respectively, (a) – (c): original anchor setting (Group A); (d) – (f): 1-anchor-change (Group B); (g)
464 – (i): 2-anchor-change (Group C); (j) – (l): 3-anchor-change (Group D).

466 The four best models are evaluated on the test dataset. Figure 11 displays results for the best
467 models of Group A – D at a 0.5 confidence threshold. Some obvious undetected anomalies are also

highlighted. The best model of the original anchor setting (Group A) performs appropriately on both anomaly and component. However, the accuracy of anomaly detection and the size of boundary boxes are the main issues of the original anchor setting. Some anomalies are not detected in Figure 11 (a) to (c), resulting in the drop of mAP. Prior knowledge of anchor size is not used in this group, bringing inconsistent anomaly boundary box sizes (as shown in Figure 11 (a) – (c)). The performance results are consistent with the evaluation data in Table 3: The models from Groups B, C and D identify component boxes in the right positions, and the performance of anomaly detection on Group D is the best, followed by Group B and then Group C. However, it is worth mentioning that the boundary boxes in Groups B and C (Figure 11 (d) – (i)) have acceptable dimensions for the component prediction, while the boundary boxes in Group D are too large for components. An abrupt and significant drop (23.5% from Table 3) of component identification accuracy happens when the confidence threshold increases from 0.5 to 0.6, indicating that the use of larger box sizes is improper.

Next, models from Groups B and C are candidate models for future applications. Further identification results on the test dataset are shown in Figure 12. For the model with only 1-anchor-change, the prediction for anomalies is acceptable when the confidence threshold is 0.4 (Figure 12 (a) – (c)). Most of the unfused parts are identified and marked within boundary boxes (around 53% accuracy). The model with a 0.5 confidence threshold (Figure 12 (d) – (f)) removes some low confidence boundary boxes, resulting in missing some real anomaly parts. Similar results for 2-anchor-change models are shown in Figure 12 (g) – (l). The high confidence threshold deletes some boxes with right anomalies inside. When comparing with 1-anchor-change model and 2-anchor-change model, it is noticed that the sizes of the component boxes are always larger in the 2-anchor-changes model than the 1-anchor-change model, even exceeding the size of the image sizes.

Therefore, the model with 1-anchor-change and a 0.4 confidence threshold is selected lastly as the best prediction model for anomalies and component identifications. Based on the data in Table 3, the precision on component prediction achieves 100% precision, while anomaly prediction receives 53% accuracy, with a 76.5% mAP globally.

Figure 12. Performances comparisons on 1-anchor-change and 2-anchor-change: (a) – (c): 1-anchor-change at confidence threshold 0.4; (d) – (f): 1-anchor-change at confidence threshold 0.5; (g) – (i): 2-anchor-changes at confidence threshold 0.4; (j) – (l): 2-anchor-change at confidence threshold 0.5.

5. Case study

A component featuring right angles, obtuse angles and acute angles was fabricated to test the model performance in a case study replicating a real-world manufacturing scenario. The component, shown in Figure 13, is comprised of eight layers and welding parameters were kept constant throughout the manufacturing process: wire feed rate at 5.5 m/min and travel speed of the robot is 0.5 m/min. The shielding gas used in this case study is 100% Argon. The welding process

508 used for this component is Fronius' CMT-pulse. Figure 13 also provides illustrations of the
 509 component structure, path planning layout and prediction results. In similar fashion to previously
 510 fabricated components, this component is manufactured in clockwise and anticlockwise
 511 commutatively, as shown in Figure 13 (a) - (b).

512 As discussed in section 4, the 1-anchor-change model performs best for defect detection on our
 513 dataset and therefore is used to determine the performance for different shapes under 0.4
 514 confidence. Figure 13 (d) - (f) are prediction results from the model. Although the shape of the
 515 component differs from our training dataset, most of the lack-of-fusion defects are detected
 516 accurately by the model. However, some defects still go undetected. For example, in Figure 13 (e)
 517 and (f), the model misjudges component edges with lack-of-fusion defects on a number of
 518 occasions. However, for our application, a model which occasionally predicts normal part as
 519 defective is always better than a model that misses possible defects, which reduces the impact of
 520 these errors. For the detection of the component, the model produces a number of inaccurate
 521 predictions. The model only recognizes the component in Figure 13 (d), indicating that the model
 522 can identify components of different shapes with low accuracy. The main reason causing this low
 523 accuracy in component identification is the small training dataset, especially for the component
 524 samples (less than 1000).

525

526

527 Figure 13. Illustrations of the component and the detection results: (a) - (b): Path plans and
 528 component dimensions. (c) Real photo after eight-layer manufacture. (d) - (f): The detection results
 529 using the pre-trained best 1-anchor-change model.

530

531 6. Conclusion and Future Work

532 In this work, the YOLOv3 algorithm is applied to a WAAM process to detect: a) the position
 533 of the welded component and, b) the existence of welding defects on the welded surface. This paper
 534 investigates YOLOv3's performance with different anchor settings with a small training dataset.

The best model performance based on the validation dataset is selected for the component position identification and detection of defects. It was found that the original anchor setting for the COCO dataset is unsuitable for the WAAM defect dataset. Moderate changes to the anchor settings (1-2 changes) bring the prior knowledge of the component and anomaly sizes to prediction models, improving the prediction mAP by about 5%. In this paper, the 1-anchor-change model achieves 100% accuracy on component identification and 53% precision on the anomaly prediction. This model proves the feasibility of using a small dataset to train a YOLOv3 model to identify defects in a WAAM process. Furthermore, the accurate identification of component locations provides a prerequisite for high precision assessment of WAAM components qualities.

The work presented in this paper forms an essential component in a wider intelligent monitoring system for WAAM. It will provide image-based defect detection capabilities that can be integrated with other defect detection sub-systems, such as the electrical signals monitoring system presented in [47]. Future work will focus on embedding the model with a functional Graphical User Interface (GUI), which will assist technicians in the practical task of detecting the presence of anomalies. An interactive GUI will also provide a more streamlined means of integrating these models into the wider defect detection monitoring system.

Funding

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

Conflict of interest

The authors declare no competing interests.

Code availability

No applicable.

Authors' contributions

Yuxing Li: Conceptualization, Methodology, Writing - original draft

Haochen Mu: Writing, revising - review and editing

Joseph Polden: Writing - review and editing

Huijun Li: Supervision

Lei Wang: Supervision

Chunyang Xia: Writing, revising – review and editing

Zengxi Pan: Supervision, Writing - review and editing

References

- [1] F. Martina, S. W. Williams, and P. A. Colegrove, "Improved microstructure and increased mechanical properties of additive manufacture produced Ti-6Al-4V by interpass cold rolling," 2013.
- [2] S. W. Williams, F. Martina, A. C. Addison, J. Ding, G. Pardal, and P. Colegrove, "Wire + Arc Additive Manufacturing," *Materials Science and Technology*, vol. 32, no. 7, pp. 641-647, 2016, doi: 10.1179/1743284715y.0000000073.

- 580 [3] D. Ding, Z. Pan, S. van Duin, H. Li, and C. Shen, "Fabricating Superior NiAl Bronze
581 Components through Wire Arc Additive Manufacturing," *Materials*, vol. 9, no. 8, 2016, doi:
582 10.3390/ma9080652.
- 583 [4] D. Ding *et al.*, "Towards an automated robotic arc-welding-based additive manufacturing
584 system from CAD to finished part," *Computer-Aided Design*, vol. 73, pp. 66-75, 2016, doi:
585 10.1016/j.cad.2015.12.003.
- 586 [5] D. Ding, Z. Pan, D. Cuiuri, and H. Li, "A practical path planning methodology for wire and
587 arc additive manufacturing of thin-walled structures," *Robotics and Computer-Integrated
588 Manufacturing*, vol. 34, pp. 8-19, 2015, doi: 10.1016/j.rcim.2015.01.003.
- 589 [6] D. Ding, Z. Pan, D. Cuiuri, H. Li, S. van Duin, and N. Larkin, "Bead modelling and
590 implementation of adaptive MAT path in wire and arc additive manufacturing," *Robotics
591 and Computer-Integrated Manufacturing*, vol. 39, pp. 32-42, 2016, doi:
592 10.1016/j.rcim.2015.12.004.
- 593 [7] D. Ding, Z. Pan, D. Cuiuri, H. Li, N. Larkin, and S. van Duin, "Automatic multi-direction
594 slicing algorithms for wire based additive manufacturing," *Robotics and Computer-
595 Integrated Manufacturing*, vol. 37, pp. 139-150, 2016, doi: 10.1016/j.rcim.2015.09.002.
- 596 [8] T. B. Kim, S. Yue, Z. Zhang, E. Jones, J. R. Jones, and P. D. Lee, "Additive manufactured
597 porous titanium structures: Through-process quantification of pore and strut networks,"
598 *Journal of Materials Processing Technology*, vol. 214, no. 11, pp. 2706-2715, 2014, doi:
599 10.1016/j.jmatprotec.2014.05.006.
- 600 [9] P. Wang, S. Hu, J. Shen, and Y. Liang, "Characterization the contribution and limitation of
601 the characteristic processing parameters in cold metal transfer deposition of an Al alloy,"
602 *Journal of Materials Processing Technology*, vol. 245, pp. 122-133, 2017, doi:
603 10.1016/j.jmatprotec.2017.02.019.
- 604 [10] G. Bissacco, H. N. Hansen, and L. De Chiffre, "Micromilling of hardened tool steel for
605 mould making applications," *Journal of Materials Processing Technology*, vol. 167, no. 2-3,
606 pp. 201-207, 2005, doi: 10.1016/j.jmatprotec.2005.05.029.
- 607 [11] B. Baufeld, "Mechanical Properties of INCONEL 718 Parts Manufactured by Shaped Metal
608 Deposition (SMD)," *Journal of Materials Engineering and Performance*, vol. 21, no. 7, pp.
609 1416-1421, 2011, doi: 10.1007/s11665-011-0009-y.
- 610 [12] B. Wu *et al.*, "A review of the wire arc additive manufacturing of metals: properties, defects
611 and quality improvement," *Journal of Manufacturing Processes*, vol. 35, pp. 127-139, 2018,
612 doi: 10.1016/j.jmapro.2018.08.001.
- 613 [13] J. Xiong, G. Zhang, J. Hu, and L. Wu, "Bead geometry prediction for robotic GMAW-based
614 rapid manufacturing through a neural network and a second-order regression analysis,"
615 *Journal of Intelligent Manufacturing*, vol. 25, no. 1, pp. 157-163, 2012, doi:
616 10.1007/s10845-012-0682-1.
- 617 [14] J. Xiong and G. Zhang, "Adaptive control of deposited height in GMAW-based layer
618 additive manufacturing," *Journal of Materials Processing Technology*, vol. 214, no. 4, pp.
619 962-968, 2014, doi: 10.1016/j.jmatprotec.2013.11.014.
- 620 [15] J. Xiong, Z. Yin, and W. Zhang, "Closed-loop control of variable layer width for thin-walled
621 parts in wire and arc additive manufacturing," *Journal of Materials Processing Technology*,
622 vol. 233, pp. 100-106, 2016, doi: 10.1016/j.jmatprotec.2016.02.021.
- 623 [16] J. Xiong, Y.-J. Li, Z.-Q. Yin, and H. Chen, "Determination of Surface Roughness in Wire
624 and Arc Additive Manufacturing Based on Laser Vision Sensing," *Chinese Journal of
625 Mechanical Engineering*, vol. 31, no. 1, 2018, doi: 10.1186/s10033-018-0276-8.

- [17] R. Li, J. Xiong, and Y. Lei, "Investigation on thermal stress evolution induced by wire and arc additive manufacturing for circular thin-walled parts," *Journal of Manufacturing Processes*, vol. 40, pp. 59-67, 2019, doi: 10.1016/j.jmapro.2019.03.006.
- [18] T. D. Ngo, A. Kashani, G. Imbalzano, K. T. Q. Nguyen, and D. Hui, "Additive manufacturing (3D printing): A review of materials, methods, applications and challenges," *Composites Part B: Engineering*, vol. 143, pp. 172-196, 2018, doi: 10.1016/j.compositesb.2018.02.012.
- [19] D. Babot, G. Berodias, and G. Peix, "Detection and sizing by X-ray Compton scattering of near-surface cracks under weld deposited cladding," *NDT & E International*, vol. 24, no. 5, pp. 247-251, 1991.
- [20] S. Naito, S. Yamamoto, and S. Yamamoto, "Novel X-ray backscatter technique for detecting crack below deposit," *Toshiba Corporation, Japan*. <http://www.ndt.net/article/jrc-nde2009/papers/110.pdf>, 2009.
- [21] J. Mireles, S. Ridwan, P. A. Morton, A. Hinojos, and R. B. Wicker, "Analysis and correction of defects within parts fabricated using powder bed fusion technology," *Surface Topography: Metrology and Properties*, vol. 3, no. 3, 2015, doi: 10.1088/2051-672x/3/3/034002.
- [22] D. Cerniglia, M. Scafidi, A. Pantano, and J. Rudlin, "Inspection of additive-manufactured layered components," *Ultrasonics*, vol. 62, pp. 292-8, Sep 2015, doi: 10.1016/j.ultras.2015.06.001.
- [23] W. Du, Q. Bai, Y. Wang, and B. Zhang, "Eddy current detection of subsurface defects for additive/subtractive hybrid manufacturing," *The International Journal of Advanced Manufacturing Technology*, vol. 95, no. 9-12, pp. 3185-3195, 2017, doi: 10.1007/s00170-017-1354-2.
- [24] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 580-587.
- [25] W. Liu *et al.*, "Ssd: Single shot multibox detector," in *European conference on computer vision*, 2016: Springer, pp. 21-37.
- [26] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779-788.
- [27] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in neural information processing systems*, 2015, pp. 91-99.
- [28] Z. Cai and N. Vasconcelos, "Cascade r-cnn: Delving into high quality object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 6154-6162.
- [29] C.-Y. Fu, W. Liu, A. Ranga, A. Tyagi, and A. C. Berg, "Dssd: Deconvolutional single shot detector," *arXiv preprint arXiv:1701.06659*, 2017.
- [30] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018.
- [31] J. Kim, J. Lee, M. Chung, and Y.-G. Shin, "Multiple weld seam extraction from RGB-depth images for automatic robotic welding via point cloud registration," *Multimedia Tools and Applications*, vol. 80, no. 6, pp. 9703-9719, 2020, doi: 10.1007/s11042-020-10138-7.

- [32] Y. Zou, M. Zhu, and X. Chen, "A Robust Detector for Automated Welding Seam Tracking System," *Journal of Dynamic Systems, Measurement, and Control*, vol. 143, no. 7, 2021, doi: 10.1115/1.4049547.
- [33] L. Yang, J. Fan, Y. Liu, E. Li, J. Peng, and Z. Liang, "Automatic Detection and Location of Weld Beads With Deep Convolutional Neural Networks," *IEEE Transactions on Instrumentation and Measurement*, vol. 70, pp. 1-12, 2021, doi: 10.1109/tim.2020.3026514.
- [34] W. Dai *et al.*, "Deep learning assisted vision inspection of resistance spot welds," *Journal of Manufacturing Processes*, vol. 62, pp. 262-274, 2021, doi: 10.1016/j.jmapro.2020.12.015.
- [35] H. Geng, J. Li, J. Xiong, X. Lin, and F. Zhang, "Optimization of wire feed for GTAW based additive manufacturing," *Journal of Materials Processing Technology*, vol. 243, pp. 40-47, 2017, doi: 10.1016/j.jmatprotec.2016.11.027.
- [36] S. Feng, H. Zhou, and H. Dong, "Using deep neural network with small dataset to predict material defects," *Materials & Design*, vol. 162, pp. 300-310, 2019, doi: 10.1016/j.matdes.2018.11.060.
- [37] C. Ajmi, J. Zapata, J. J. Martínez-Álvarez, G. Doménech, and R. Ruiz, "Using Deep Learning for Defect Classification on a Small Weld X-ray Image Dataset," *Journal of Nondestructive Evaluation*, vol. 39, no. 3, 2020, doi: 10.1007/s10921-020-00719-9.
- [38] R. Girshick, "Fast r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2015, pp. 1440-1448.
- [39] J. Pang, K. Chen, J. Shi, H. Feng, W. Ouyang, and D. Lin, "Libra r-cnn: Towards balanced learning for object detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 821-830.
- [40] J. Redmon and A. Farhadi, "YOLO9000: better, faster, stronger," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 7263-7271.
- [41] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770-778.
- [42] T.-Y. Lin *et al.*, "Microsoft coco: Common objects in context," in *European conference on computer vision*, 2014: Springer, pp. 740-755.
- [43] M. D. Zeiler, "Adadelta: an adaptive learning rate method," *arXiv preprint arXiv:1212.5701*, 2012.
- [44] G. Hinton, N. Srivastava, and K. Swersky, "Neural networks for machine learning lecture 6a overview of mini-batch gradient descent," *Cited on*, vol. 14, no. 8, p. 2, 2012.
- [45] L. Bottou, "Large-scale machine learning with stochastic gradient descent," in *Proceedings of COMPSTAT'2010*: Springer, 2010, pp. 177-186.
- [46] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.
- [47] Y. Li *et al.*, "A defect detection system for wire arc additive manufacturing using incremental learning," *Journal of Industrial Information Integration*, 2021, doi: 10.1016/j.jii.2021.100291.