Journal of Manufacturing Processes 71 (2021) 374–383

Contents lists available at ScienceDirect

Journal of Manufacturing Processes

journal homepage: www.elsevier.com/locate/manpro

Hybrid machine learning-enabled adaptive welding speed control

Joseph Kershaw [a,c], Rui Yu [b,c], YuMing Zhang [b,c], Peng Wang [a,b,c,*]

[a] Department of Mechanical Engineering and Institute for Sustainable Manufacturing, University of Kentucky, Lexington, KY 40506, USA
[b] Department of Electrical and Computer Engineering, University of Kentucky, Lexington, KY 40506, USA
[c] Institute for Sustainable Manufacturing, University of Kentucky, Lexington, KY 40506, USA

ARTICLE INFO

Keywords:
Robotic welding
Adaptive control
Machine learning
Convolutional neural network
Gradient descent

ABSTRACT

Industrial robots have become more diverse and common for automating manufacturing processes, such as welding. Existing robotic control, however, is incapable of adaptively adjusting its operation in response to a dynamic welding environment, whereas a skilled human welder can. Sophisticated and adaptive robotic control relies on the effective and efficient processing of perception data, characterization and prediction of highly dynamic systems, and real-time adaptative robotic reactions. This research presents a preliminary study on developing appropriate Machine Learning (ML) techniques for real-time welding quality prediction and adaptive welding speed adjustment for GTAW welding at a constant current. In order to collect the data needed to train the hybrid ML models, two cameras are applied to monitor the welding process, with one camera (available in practical robotic welding) recording the top-side weld pool dynamics and a second camera (unavailable in practical robotic welding, but applicable for training purpose) recording the back-side bead formation. Given these two data sets, correlations can be discovered through a convolutional neural network (CNN) that is good at image characterization. With the CNN, top-side weld pool images can be analyzed to predict the back-side bead width during active welding control. Furthermore, the monitoring process has been applied to multiple experimental trials with varying speeds. This allowed the effect of welding speed on bead width to be modeled through a Multi-Layer Perceptron (MLP). Through the trained MLP, a computationally efficient gradient descent algorithm has been developed to adjust the travel speed accordingly to achieve an optimal bead width with full material penetration. Because of the nature of gradient descent, the robot would change faster when the quality is further away and then fine-tune the speed when it was close to the goal. Experimental studies have shown promising results on real-time bead width prediction and adaptive speed adjustment to realize ideal bead width.

1. Introduction

Welding represents one of the earliest manufacturing processes that embrace robotics. The long history is also paired with popularity, with almost half of all industrial robots deployed for welding [1]. Even with this majority, more welding robots are being demanded by manufacturers with a predicted market increase of USD 4 billion in the next 5 years [2]. This robotic welding trend is compounded by a predicted worker shortage of around 375 thousand welders within the same period [3], implying that autonomous welding solutions are needed to maintain and grow the production throughput.

Today, robotics have been widely deployed in different welding applications. A majority of these robots are used for repetitive welding tasks that require basic welding skills on an open machine floor, but are beginning to be used for welding tasks that need to be performed in remote and confined areas that are dangerous for humans. In these robotic welding applications, skilled human welders are still needed for designing and planning the robotic welding paths and other process specifications, and robots then perform the actual welding operation along the designed paths with high precision. Basic robotic welding applications include plate and pipe welding [4]. This concept can be scaled up to include mobility to remote welding, such as welding on installed wind turbines. While it is difficult and dangerous for human welders to perform this large and aerial operation, a multiple degree-of-freedom welding robot can be mounted on a rotational track to achieve a full reach of the seam [5]. Similarly, autonomous welding in confined spaces, such as in the interior hull of large ships, typically involves highly specialized robots, such as robots mounted on a moveable rail system or walkable robots [6]. One challenge faced by both basic and specialized robotic welding is welding path planning. Optimal path

* Corresponding author at: Department of Mechanical Engineering and Institute for Sustainable Manufacturing, University of Kentucky, Lexington, KY 40506, USA.
E-mail address: Edward.Wang@uky.edu (P. Wang).

https://doi.org/10.1016/j.jmapro.2021.09.023
Received 22 July 2021; Received in revised form 10 September 2021; Accepted 16 September 2021
Available online 30 September 2021

planning aims at maximizing welding quality and efficiency, while also complementing the mobility and control capability of robots.

Many path planning techniques have been developed. In pipe welding, the trigonometric relationship was leveraged to maintain an appropriate welding angle during the joining operation [7]. In multi-pass welding, where welding parameters and paths associated with each pass need to be determined upon the last pass's quality, a formulaic relationship was established to calculate the welding speed needed to achieve the desired bead height [8]. Stochastic path planning algorithms based on graph theory were also studied for optimizing welding operation times, especially in automotive chassis spot welding. Given the welding constraints (e.g., groove edges) obtained through computer vision-based edge detection systems [9], some swarm-based optimization algorithms, such as the ant colony algorithm [10] or beetle antennae search [11] that randomly search the optimum path connecting two areas, can be applied to finding optimum welding paths. However, these algorithms mainly concern with welding paths and provide little guidance on welding speed, leading to inferior welding quality such as excessive or lack of penetration [12]. Also, many welding constraints are difficult to model or obtain because of the complicated nature of welding (e.g., overhangs caused by gravitational effects on the liquid weld pool), leading to inappropriate design of welding angles and paths [13]. Furthermore, the planning of welding path, speed, and angle, which happen before the real operation, cannot account for any process uncertainties that lead to deviation from desired quality. Hence, besides path planning, adaptive control is needed to adaptively adjust welding operations to ensure and maintain the desired quality.

In adaptive process control, operations are real-time adaptively adjusted based on the in-situ sensing measurement to ensure the welding quality. For example, the welding current was adaptively adjusted to compensate tool degradation or material defects to ensure proper bonding spot welds through an entire operation cycle [14]. In this study, robotic tool performance degradation or unexpected variations in the workpiece led to changes in the resistance of the welding circuit, which was measured based on external measurements of current and voltage. This measured resistance was then compared to a baseline value of a previously performed spot weld of good quality. If a discrepancy existed, the current would steadily increase for higher resistances and reduce for lower ones until the baseline resistance had been achieved [14]. In another study of spot welding of high-strength steel, a fuzzy proportional-integral (PI) controller was developed to avoid shunting, an effect where the current travels around the desired spot weld as opposed to through it [15]. This was done by measuring the current, power, and voltage change to detect this flaw and would leverage a fuzzy PI controller to modify the welding power and current. In the field of laser welding, welding power was adaptively adjusted to enforce a precise cooling curve to achieve desired microhardness [16]. By comparing the measured temperature and the desired temperature at each time step, a heat diffusion formula that was experimentally obtained was used to determine the adjustment of the power.

Emerging Machine Learning (ML) techniques have also been extensively investigated for adaptive robotic welding control. A back-propagation neural network-based control policy has been shown to be applicable for deep gap welds. The network correlated the information of geometries of pieces to be joined and the desired bead width to the optimal weld parameters of power, speed, and feed. These parameters were then adaptively changed in response to varying piece geometry through the network nonlinear mapping during welding operations [17]. A study on pipe welding investigated neuro-fuzzy networks in estimating the bead penetration based on the profile of the weld pool that was generated from laser sensing. Then a gradient-based control system established upon a linear model of weld pool penetration was utilized to adjust the speed and current to maintain constant penetration [18]. A reinforcement learning-based control algorithm was developed for laser welding, where radiation produced during the welding process was measured by electromagnetic sensors [19]. An encoder then compressed the data to reduce dimensionality and generate high-level features that would be critical for operation determination, followed by a convolutional neural network-based classifier to detect the quality of the weld following the determined operation. Based on the quality feedback, the controller would generate either a reward or demerit to update the encoder. While these ML-based control methods do improve the adaptability of a welding robot, these methods were primarily used to improve the quality and reliability of robotic welding operations already in effect. To approach the goal of achieving technically advanced welding operations only performable by humans, the welding robots should learn from human welders in terms of perception, processing, control, and action.

The proposed adaptive controller takes a humanistic approach to the robotic welding process. Instead of using sensors that are only usable by machines, the decision process is based entirely on senses available to humans. Furthermore, the only parameters that humans can change while welding is their movements, as power and feeds are set before each task. While welding, an experienced welder would observe the formation of the molten weld pool and adjust their motions to achieve the desired weld. Accordingly, the proposed controller mimics the perception and decision-making process of a welder through a hybrid ML framework, granting the adaptability of humans with the reliability of robotics. In this framework, the robotic perception is enabled by a camera recording the top-side weld pool dynamics, and estimation of welding quality is enabled by a convolutional neural network (CNN) that correlates weld pool images to backside bead width (ground truth coming from a second camera recording the back-side bead formation, only available for training purpose). Subsequently, adaptive decision-making is realized through a Multi-Layer Perceptron (MLP) that correlates welding speeds to bead width, upon which a computationally efficient gradient descent algorithm is developed to adjust the travel speed accordingly to achieve an optimal bead width with full material penetration. The entire framework is computationally light, ensuring real-time adaptive control.

The remaining of the paper is organized as follows: Section 2 introduces the theoretical development, including CNN for sensing imaging processing and estimation of bead width, MLP for correlating welding speed and time to bead width, and the gradient descent-based speed adjustment. Section 3 covers the details of a preliminary experimental evaluation, including experimental setup, data acquisition, image pre-processing, configurations of CNN and MLP, simulation, experimental cases of adaptive speed adjustment to achieve desired bead width, results and discussions. Conclusions are drawn finally.

2. Methodology

Adaptive robotic control in response to dynamic welding conditions relies on two elements: 1) real-time characterization of welding state through processing the sensing data; and 2) determination of adjustment of process parameters based on the difference between actual and ideal welding state. The latter can be further decomposed to 2.a) identifying the correlations between process parameters and welding state and 2.b) mapping the difference between actual and ideal welding state to the adjustment of process parameters. In the presented study where a topside camera is available to capture the weld pool dynamics, a hybrid machine learning framework is proposed to realize adaptive control in robotic welding: 1) Convolutional Neural Network (CNN) is proposed to correlate weld pool images to back-side bead width; 2) Multi-Layer Perceptron (MLP) is leveraged to discover the dependency of bead width on welding speed and welding time; and 3) A gradient descent algorithm is applied for real-time adjustment of welding speed to achieve and maintain ideal bead width.

2.1. CNN for prediction of bead width

Attributed to the advancement in computing power, non-linear

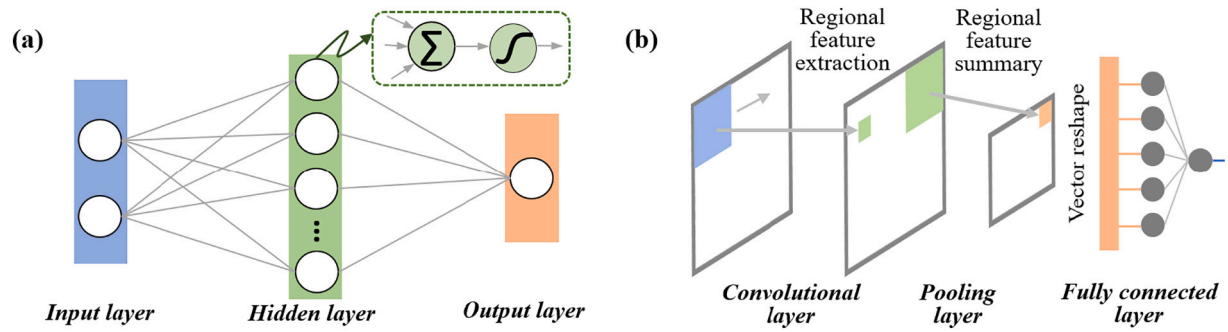

Fig. 1. Illustration of MLP and CNN architectures.

activation, regularization, and many other optimization techniques, a stack of many layers in a network is allowed for modeling nonlinear and non-stationary systems without suffering severe gradient vanishing problems. Many deep neural networks have occurred in the past two decades, including deep belief networks [20], CNN [21], recurrent neural network [22], and CNN is the most popular one. CNN is developed for image processing and pattern recognition, and hence suitable for categorical classification or regression sensing images [21]. In this study, front-side weld pool images are correlated to back-side bead width values through CNN, representing a regression problem. Since weld pool dynamics directly determine the welding penetration and thereafter the bead width, one-to-one mapping exists between weld pool images and bead width values, and CNN-based regression is hence suitable to establish the mapping.

A typical CNN architecture includes three types of layers: convolutional layer, pooling layer, and fully connected layer, as illustrated in Fig. 1 (b). Among the layers, convolutional and pooling layers are used to extract spatial features (e.g., motif shape, relative locations between motifs) from the images, and the fully connected layers correlate the extracted features to the image categories to be classified into or the targets to be regressed into. While the fully connected layers are commonly seen in standard artificial neural networks, a convolutional layer convolves regional image pixels with a trainable kernel to extract features. In a two-dimensional case, the convolutional process with a $n \times n$ kernel can be described as:

$$Y_{i,j} = \sum_{k=1}^{n} \sum_{l=1}^{n} K_{k,l} \cdot X_{i+k-1,j+l-1} \tag{1}$$

where $Y_{i,j}$ denotes the i_{th} row and j_{th} column of the obtained feature map, $K_{k,l}$ is the k_{th} row and l_{th} column of the kernel, and $X_{i+k-1,j+l-1}$ denotes the $(i + k\text{-}1)_{\text{th}}$ row and $(j + l\text{-}1)_{\text{th}}$ column of the input image. Eq. (1) essentially represents the weighted sum of image pixels over a predetermined reception field $n \times n$. The kernel weights, after appropriate training, could detect if local regions of input images contain a certain shape of interest. Multiple kernels can be applied in a convolutional layer for the detection of multiple spatial features of interest. Along with the network propagation, multiple convolutional layers can be stacked together for multi-level feature extraction, scaling from low-level features (e.g., curves, hedges) to high-level motifs (e.g., weld pool-related shapes). Towards the end of the network architecture, the extracted features are more abstract and related to the physical object.

Pooling layers are usually attached to convolutional layers to reduce the feature map size and computational effort, by summarizing extracted features in regions. Typically utilized pooling kernels include max(•) and average(•), which outputs the maximum pixel value or averaged pixel value over the reception field. Also, introducing pooling layers force the network to pay attention to relative (instead of absolute) locations of motifs, allowing the network to be translation-invariant in feature extraction. Fully connected layers can then be applied to map extracted weld pool-related features to the bead width.

The training of a CNN can be realized by many optimization algorithms, such as Stochastic Gradient Descent (SGD) and Adam [23], which are available in most DL toolboxes. For this regression problem, standard Mean Square Error (MSE) can be utilized as the network training loss function.

2.2. MLP for correlating welding speed to bead width

To determine how to adaptive adjust welding speed to minimize the difference between actual and ideal bead width, the correlation between welding speed and bead width needs to be identified first, so that the bead width differences can be reversely mapped through the correlation to deltas of welding speed. Throughout the welding process, the welding dynamics, such as thermal fusion and convection, would take some time to stabilize. Especially in the early stage of a welding process, welding time is an important factor that determines welding penetration and bead width. Hence, the dependence of bead width on both welding speed and time should be discovered, assuming other welding parameters (e.g., current, voltage) remain constant. The dependence is discovered through MLP in this study.

MLP provides an effective way to correlate input features to variables of interest or classes of conditions, where the relationships are nonlinear [24,25]. A typical 3-layer MLP configuration includes one input layer, one or multiple hidden layers, and one output layer, as illustrated in Fig. 1(a). Each layer contains one or more neurons; while the number of input and output neurons are determined by the number of input features and output variables, the number of hidden layers and hidden neurons can be arbitrarily defined. Neurons among layers are connected through weights. Along the network forward propagation, weighted inputs are fed to the hidden neurons, where hidden features are generated after nonlinear activations (e.g., sigmoid and hyperbolic tangent) and passed to subsequent hidden layers or the output layer. This is very similar to the convolutional operation in Eq. (1), except for the weighted sum is calculated in a 2 or higher-dimensional space in a CNN but 1-dimensional space in an MLP. The mapping from inputs to predicted output in a 3-layer perceptron can be expressed as:

$$BW_{pred} = \sigma(\sigma([V, T]^{*}W_1)^{*}W_2) \tag{2}$$

where BW_{pred} represents the predicted bead width, V and T denote the welding speed and time, W_1 and W_2 are the weights connecting input and hidden layers as well as hidden and output layers, * denotes the matrix-based multiplication, and σ is the sigmoid function. The training of an MLP can be realized through a standard gradient descent algorithm. Given the network prediction loss in terms of the difference between actual and predicted output, the network weights can be optimized through backpropagating the prediction loss and applying the gradients of loss w.r.t. weights to old weight values. The forward calculation and backward weight update can be iterated for multiple iterations until the network performance stabilizes.

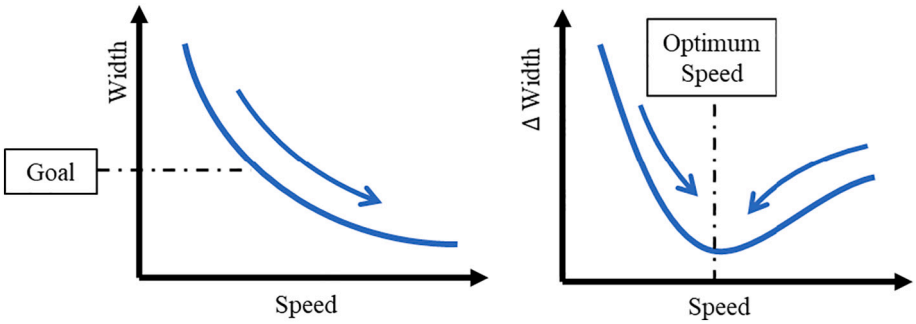

Fig. 2. Illustration of GD-based speed adjustment. (left) The dependency of bead width on welding speed: the slower speed, the wider bead width would be obtained; (right) calculated gradients of speeds: if current speed is larger than the optimum speed, a positive gradient is obtained; if current speed is smaller than the optimum speed, a negative gradient is obtained.

2.3. Gradient descent for real-time welding speed adjustment

Given the predicted bead width from in-situ weld pool sensing and welding speed-bead width correlation, welding speed can be adaptively adjusted to realize the ideal bead width. Many optimization techniques can be applied for adaptive process control, efficiency is prioritized over other evaluation metrics for the selection of optimization techniques considering the highly dynamic welding environment. Gradient Descent (GD) [26] is selected because of its high computational efficiency and performance stability, with an illustration of GD-based speed adjustment shown in Fig. 2.

The GD-based adaptive speed adjustment is different from GD-based network parameter optimization. In the latter, the gradients of the network loss function (represented by the MSE between actual and predicted bead width) with respect to network parameters are calculated and applied to recursive update of parameters. In the former, the gradient of optimization loss, represented by the MSE of predicted and ideal bead width $f = \frac{1}{2}(BW_{pred} - BW_{ideal})^2$, with respect to welding speed at a certain welding time is calculated and applied to adjust the speed. The gradient of f w.r.t. welding speed V_k can be calculated as:

$$\Delta V = \frac{\partial f}{\partial V_k} = \frac{\partial f}{\partial BW_{pred}} \frac{\partial BW_{pred}}{\partial V_k}$$
$$= \underbrace{(BW_{pred} - BW_{ideal})}_{\text{related to output layer}} \cdot \underbrace{[(BW_{pred} \cdot (1 - BW_{pred}))^* W_2^T]}_{\text{related to hidden layer}} \cdot \underbrace{[(\sigma([V,T]^* W_1) \cdot (1 - \sigma([V,T]^* W_1)))^* W_1^T]}_{\text{related to input layer}} \Big|_{V=V_k, T=T_k} \quad (3)$$

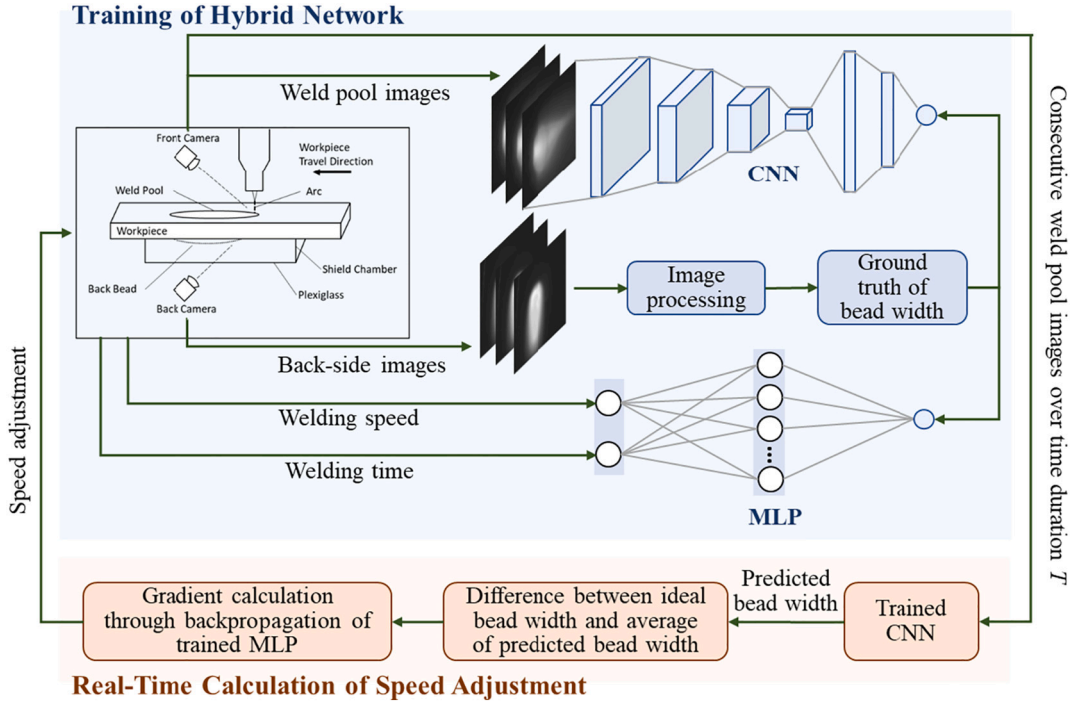

Fig. 3. Illustration of hybrid ML for Adaptive Welding Speed Adjustment, including training of hybrid CNN and MLP network as well as real-time calculation of speed adjustment.

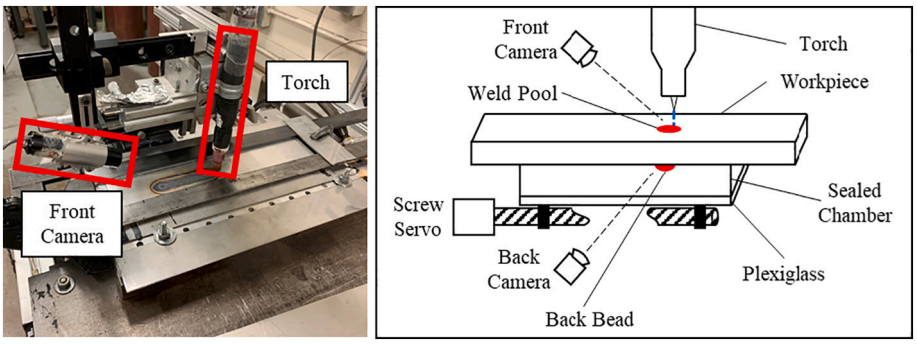

Fig. 4. Picture of system configuration (left) along with a labeled diagram(right).

It can be seen from Eq. (3) that following a chain rule, the gradient is a product of three items, derivative of optimization loss w.r.t. predicted bead width, derivative of predicted bead width w.r.t. hidden layer output, and derivative of hidden layer output w.r.t. welding speed. Since welding time is also a network input besides welding speed, the gradient of current welding speed V_k should be evaluated upon the current welding time T_k. With the gradient, the welding speed can then be adjusted through:

$$V_{k+1} = V_k - \eta \cdot \Delta V \tag{4}$$

where η is the rate of the speed adjustment. If the current speed is larger than the optimum speed, a positive gradient will be obtained. A negative speed will be obtained if the current speed is smaller than the optimum speed. Eq. (4) ensures the speed adjustment converges at the optimal speed corresponding to the ideal bead width.

The selection of η is very critical in real-time process control. A larger η would enable the speed quickly to approach the optimum speed, but with larger overshoot and longer oscillation. A smaller η would enable the speed gradually to approach the optimum speed, but with a longer convergence time. Hence, an optimal η needs to be determined experimentally with a minimum convergence time, which will be further discussed in the experimental evaluation section. Another issue that affects the real-time speed adjustment is the time duration between two adjustments. The minimum time duration should cover the communication time for the sensing images being transmitted to the computer for processing, as well as the decision time of the CNN and GD. The calculation of Eq. (3) is very efficient. For the CNN, once the network is appropriately training, processing one image to bead width value is also fast. But on the other hand, the bead width cannot be evaluated upon just one weld pool image, which will not account for process and measurement uncertainties. Instead, it should be evaluated upon a series of acquired weld pool images, by averaging the bead width values predicted from individual images. Hence, an optimal speed adjustment frequency should also be experimentally determined based on the practical application scenario.

2.4. Integrated ML framework for adaptive welding speed adjustment

The integrated framework of hybrid ML for real-time, adaptive welding speed adjustment to achieve ideal bead width is illustrated in Fig. 3.

During the network training phase, both front-side weld pool images and back-side bead formation images are collected from experimental studies, where multiple welding speeds are tested. While the weld pool images are directly fed as inputs to the CNN after straightforward preprocessing (e.g., resizing and cropping), the bead formation images go through a series of processing steps for calibration of the ground truths of bead width (to be elaborated in next Section). The calibrated bead width values are then utilized as the training outputs of both CNN and MLP. It should be noted that the weld pool and bead width images need to be synchronized first, which can be realized through the time stamping of image acquisition. Once the CNN is appropriately trained, it is supposed to be capable of accurately processing the weld pool images and predicting the bead widths. MLP is also trained to discover the dependence of bead width on welding speed and time.

In practical application, only the front-side camera and information on welding speed and time (can be acquired from the robot motion planning) are needed. Consecutive weld pool images obtained over a certain period (e.g., 0.5 s) are fed to the trained CNN for the prediction of bead widths. The averaged bead width (to account for process and measurement uncertainty) is then compared to the ideal bead width, leading to the optimization loss. Subsequently, the speed adjustment is obtained by backpropagating the optimization loss to the current speed in the MLP structure using the trained MLP parameters, using Eq. (3). Finally, the speed adjustment is routed to the welding machine for implementation per Eq. (4), for example at a frequency of every 1 s. The entire framework enables compensating process randomness and maintaining ideal welding quality. One major advantage of the framework is its computational efficiency.

Table 1
Summary of welding parameters.

Parameter	Value
Welding type	GTAW
Welding current	150 A
Tungsten diameter	2.4 mm
Shielding gas	Argon
Work piece material	T304 Stainless Steel
Work piece dimensions	300 × 30 × 2 mm
Testing speeds	4.5 mm/s – 8.0 mm/s, each speed repeated for 4 trials
Testing time	15 s per trial

3. Experimental study

To experimentally evaluate the performance of the hybrid ML framework on real-time estimation of bead width and adaptive adjustment of welding speed to achieve and maintain the desired bead width, a robotic Gas Tungsten Arc Welding (GTAW) process testbed was established. First, welding experiments under different welding speeds were conducted, while images were collected from two cameras with one recording weld pool dynamics and the other recording bead formation. The images were then processed used to train the CNN and MLP networks. With the trained networks, both simulations and experimental tests were conducted to evaluate the effectiveness and efficiency of the hybrid ML framework in adjusting the process from different initial speeds to achieve the optimal bead width.

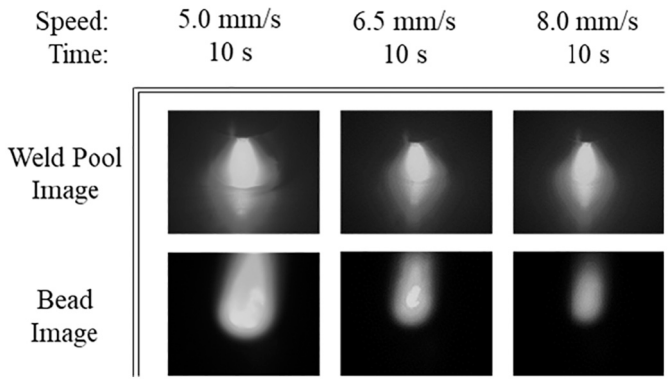

Fig. 5. Collected data samples under different welding speeds.

3.1. Experimental setup

In this setup, a GTAW torch was mounted on a moving stage to simulate the robotic GTAW process. This stage consisted of a sealed chamber with a plexiglass base and a gas intake that was mounted to a computer-controlled screw servo. Back bead shielding gas was applied to ensure the welding quality. Linear welds were made on T304 stainless steel without filler metal under variable speeds using the machine. Two cameras were statically mounted with one focused on the active weld pool and the other on the forming back bead. This system is shown in Fig. 4.

Throughout the tests, the welding current was set as a constant 150 A. Argon gas was used for both the arc and back bead shielding. The back bead shield was applied by clamping metal over the seams between the stage and workpiece followed by saturating the chamber with a continuous stream of argon gas. Speeds in the range of between 4.5 mm/s and 8.0 mm/s (with a 0.5 mm/s interval), in total 8 different welding speeds, were tested. To account for both process and measurement uncertainty, each speed was repeated for 4 trials, and each trial lasted for 15 s. The experimental details are summarized in Table 1.

3.2. Data acquisition and processing

As mentioned above, two highspeed cameras were utilized to gather images of the welding process. The cameras used were Point Grey FL3-FW-03S1C-C highspeed cameras. Each camera had a resolution of 648 × 488 and maximum sampling rate of 120 frames per second. The exposure times were set at 0.2 s for the top-side camera, and 0.3 s for back-side camera. The gamma and sharpness of both cameras were set to 1.5 and 3000, respectively, to enable an ideal observation of the weld pool and bead with clear boundary. Since they are not explicitly designed to capture bright images, both had a filter made of a welding lens placed behind the camera lens to prevent overexposure in the images. One camera recorded the weld pool dynamics and the other was positioned through the plexiglass base of the back shielding chamber onto the location of the back bead formation. The collected images from two cameras were used for training the CNN and MLP networks, but in practical applications, only the weld pool camera is needed as the trained CNN can estimate the bead width upon processing the weld pool images. Both cameras were positioned on the tail-side bead and captured 640 × 480 gray-scale images at 60 frames per second. The cameras' iris and strobe time were adjusted to avoid underexposure or overexposure that would result in all black or all white images, respectively.

With a 60 frame rate, the following information was recorded every 0.017 s during operation: the image of the front weld pool, the image of the back bead, the current time, and the welding speed. This collection resulted in 28,700 sets of samples, approximately 3600 per speed test. An illustration of collected samples under three different welding speeds is shown in Fig. 5.

While the weld pool images can be directly fed into the CNN after simple cropping and resizing, the bead image needs to be processed first to obtain the ground truths of bead width values. Previous work [22] by the authors introduced a method to convert bead images to bead width values. In this method, a brightness threshold is first set to determine the region of an image that is corresponding to the bead formation, and the area of the determined region is then mapped to bead width through a linear transformation that is obtained experimentally. However, the study is based on spot welding that produces circular beads, while the linear welding in this study produces oblong beads of inconsistent length. To address this issue, an improved version of the method introduced in [22] was developed. Fig. 6 shows a procedure of processing and converting an original bead image to a bead width value.

First, a bead image was appropriately cropped to capture only the bead formation-related region. The cropped image was then binarized into black and white, with all pixels exceeding the brightness threshold set to white. This step allows a clear boundary between bead formation and non-bead areas. The white pixels in each row were then counted, leading to the width of the bead at discrete lengths. In this study, the ground truth of bead width calibrated from individual images was set as the average of the top ten row widths, to account for uncertainties from measurement and data processing. Finally, this width was multiplied by a constant to convert from pixels to millimeters. The threshold was chosen through an iterative, trial-and-error process. An arbitrary value

Table 2
Summary of CNN configuration.

Layer name	Feature map size	Kernel size	Stride	Padding
Input	128 × 128 × 1	–	–	–
Conv 1	128 × 128 × 32	5	1	2
MaxPool	64 × 64 × 32	2	2	–
Conv 2	32 × 32 × 64	3	2	1
MaxPool	16 × 16 × 64	2	–	–
Conv 3	8 × 8 × 128	3	2	1
MaxPool	4 × 4 × 128	2	–	–
Conv 4	2 × 2 × 100	3	2	1
MaxPool	1 × 1 × 100	2	–	–
FC1	100	–	–	–
FC2	64	–	–	–
Output	1	–	–	–

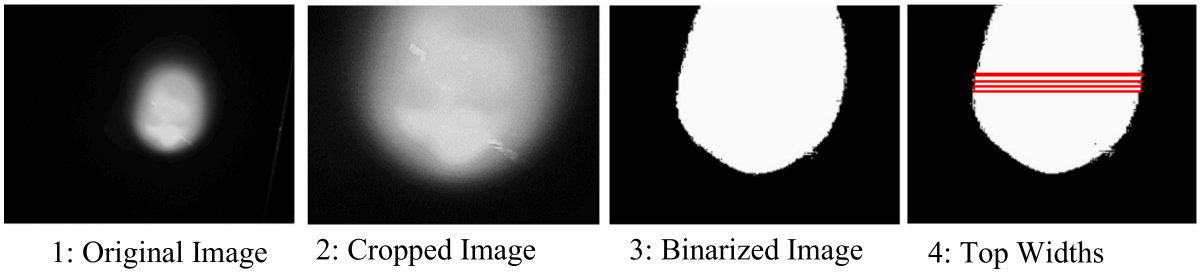

Fig. 6. Bead width calibration from original bead image, including image cropping, binarizing, counting and averaging top 10 widths.

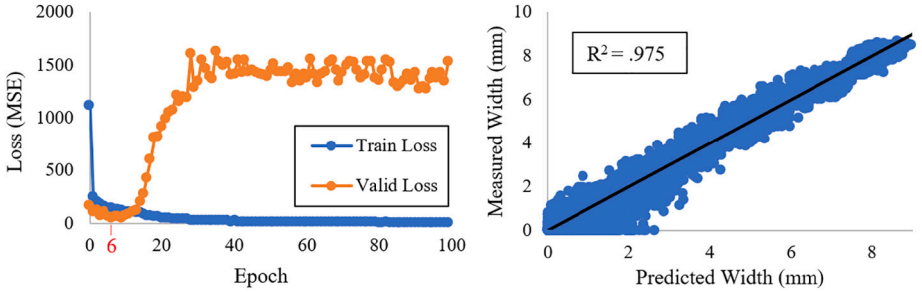

Fig. 7. CNN training curve (left) and performance on validation data (right).

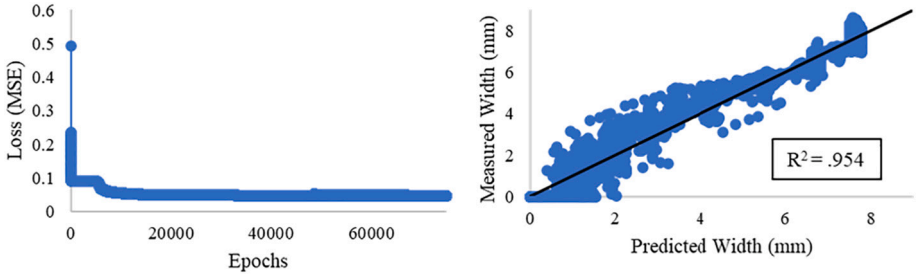

Fig. 8. MLP Training (left) and Performance (right).

was applied to the algorithm and the resulting widths were plotted against their distance. The trends in the graph were then compared to the trends in the produced back bead. If the resulting graph showed penetration where there was none, the threshold was lowered. Conversely, if there was penetration not reflected in the graph, the threshold was raised. The finally chosen threshold was 170 with a conversion factor of 0.06. The conversion factor was determined by manually measuring a stable bead width and dividing this measurement by the corresponding pixel width.

3.3. CNN and MLP configuration and training

In most ML especially DL techniques, the network structure plays a significant role in the network performance and reliability. Given a cropped weld pool image with size 128 by 128, the final selected CNN structure contains 4 convolutional (Conv) layers, 4 max-pooling layers, and 2 fully connected layers (FC). Between each convolution layer and max-pooling layer, batch normalization was performed followed by a ReLU activation function. The features extracted from the last convolution layer were directly flattened and fed to the first FC layer, without batch normalization and dropout. ReLU activation function was also performed between two FC layers. The second FC layer was directly connected to the output layer without ReLU activation. The architecture is detailed in Table 2.

The basic guideline on determining the CNN structure is to gradually reduce the feature map size throughout the entire network. The first convolution kernel was set to not cause a reduction, but each following convolution and max-pooling kernel parameters were chosen to half the feature maps of their inputs.

Among the collected 28,700 weld pool image-bead width pairs, 70% (20,090) of the gathered data were for CNN training and the remainder (8610) was used for network validation. The network was trained using the Adam optimizer with a learning rate of 0.0001, Mean Square Error (MSE) loss, and a batch size of 32 images. The choice of batch size was chosen from the best performance between networks with batch sized of 16, 32, 64, and 100. Learning rate was similarly chosen from networks with 0.01, 0.001, and 0.0001. The training curve, showing the training and validation loss over 100 epochs, is shown in Fig. 7 (left). It is noted that the CNN achieved the lowest validation loss at epoch 6 and then began to rise, while the training loss continued to decrease over the entire training period. Adopting an early stopping strategy, the network with the lowest validation loss was selected. The network was able to strongly correlate the weld pool image with the back bead width, leading to a coefficient of determination of 0.975, as indicated in Fig. 7 (right). It is also noted that the predictions are more variable at low widths, probably caused by measurement uncertainty.

As explained in Section 2, the perceptron correlates the welding speed and current welding time to bead width. A simple 3-layer perceptron with a hidden layer of eight neurons and a sigmoid activation function was applied. The hidden layer size was chosen by comparing the performance of networks with hidden layer sizes ranging from three to nine neurons. Training for this network was done with a standard gradient descent algorithm over 75,000 epochs with a learning rate of 0.0005 and MSE loss function. Again, different learning rates were examined between 0.001, 0.0005, and 0.0001. The training curve and network performance are shown in Fig. 8.

The network training converged around epoch 20,000 after seemingly converging until epoch 5000. The fully trained network was also able to strongly correlate time and speed to back bead width, achieving a coefficient of determination of 0.954. The predictions also have a larger variation for small beads, and slightly underestimate larger beads. Overall, both networks have good performance and should be adequate for subsequent adaptable process control.

3.4. Adaptive speed adjustment results

The trained CNN and MLP can then be applied for real-time adaptive welding speed adjustment. In practice, the weld pool images were continuously sent from the camera to the trained CNN for estimation of bead width. The processing of a single image by the trained CNN can be realized in real-time. Then the estimated bead width was compared to the ideal bead width for speed adjustment by calculating the speed gradient through the trained MLP. One issue to be determined is the speed adjustment frequency. An optimum frequency is a trade-off between real-time controllability and quality estimation accuracy. In this study, the speed is adjusted every half second. This time interval was partitioned with the first 0.4 s utilized by the CNN, called the perception phase, and the final 0.1 s was saved for the MLP, called the action phase.

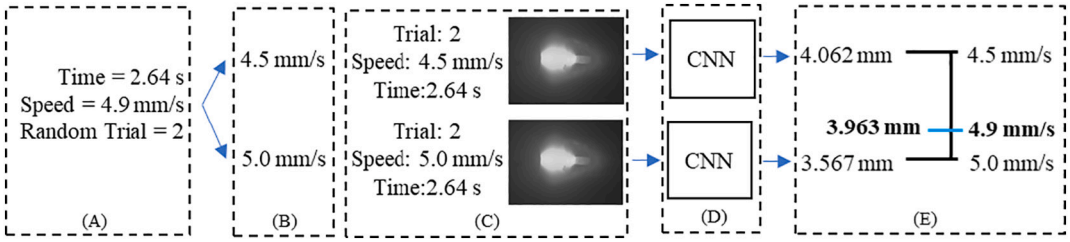

Fig. 9. Visualization of data creation for non-measured speeds: A) the simulated current welding time and speed; B) Two nearby speeds around the current welding speed determined: C) one image sampled from each speed given the current welding time; D) both images evaluated by the CNN to estimate the back bead width; E) final width determined through weighted averaging of two estimated bead width values.

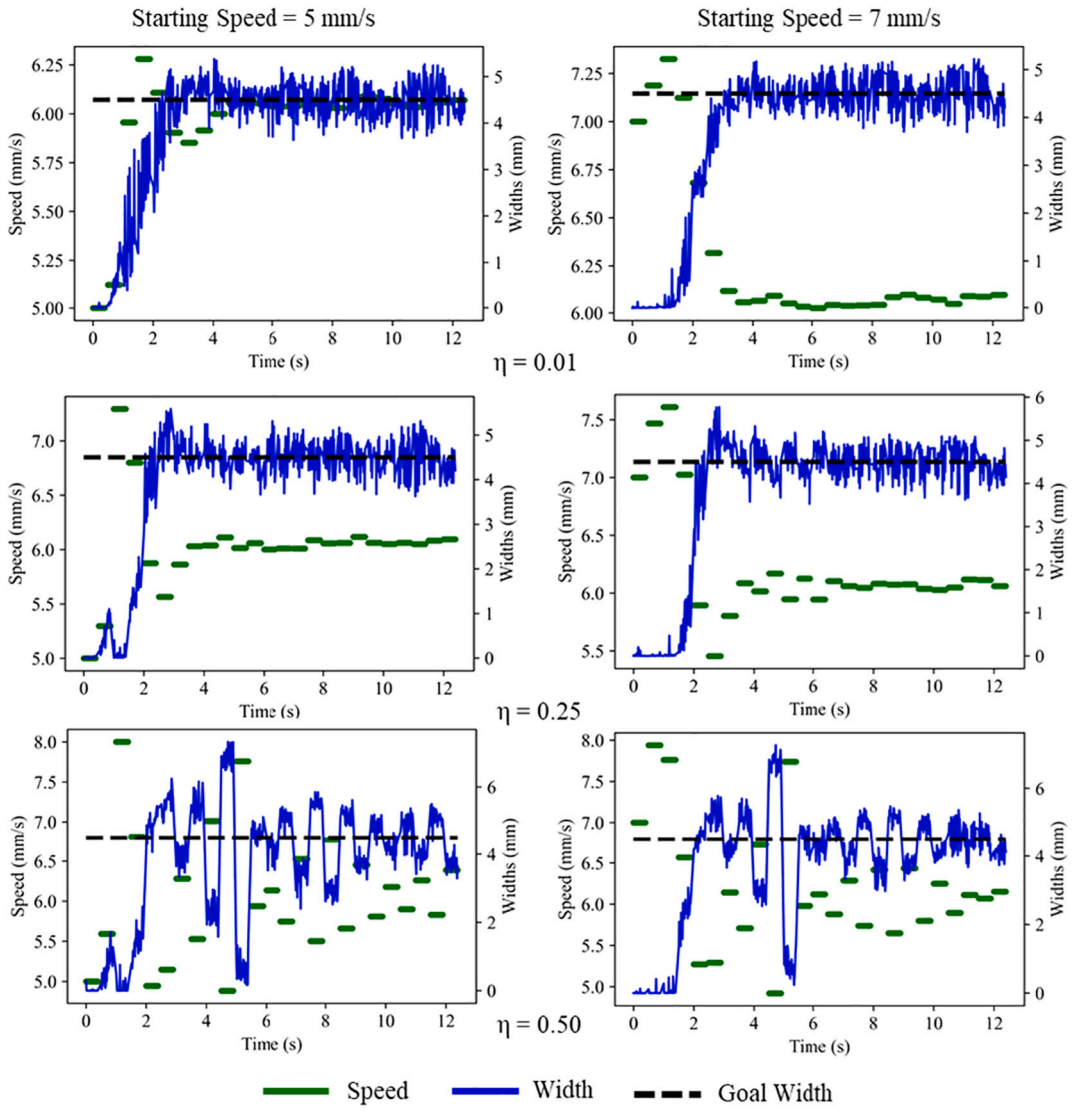

Fig. 10. Simulation results of speed adjustments under two different starting speeds and three different rates of gradient speeds.

During the perception phase, a maximum of 24 images could be gathered, leading to 24 bead width values. During the action phase, the gathered widths were averaged (to account for process uncertainty), and this value along with the current time and speed were input into the MLP to determine the required speed change. Finally, the welding speed was updated once the previous half-second interval ended.

The developed controller was first tested in a simulated welding environment. In this simulation, given a starting speed and welding time, sample weld pool images and corresponding bead width values (after CNN processing) would be selected from the experimental database and fed into the MLP for determining the speed adjustment. Once the speed was updated, the same procedures were performed for another round of speed adjustment. Since only 8 distinct speeds were tested in the experiment when the welding speed was adjusted to a value that is not covered in the 8 speeds, images were sampled from the nearest measured speeds, and a weighted bead width was generated, which is illustrated in Fig. 9.

The simulation tested 2 starting welding speeds, with a goal of 4.5

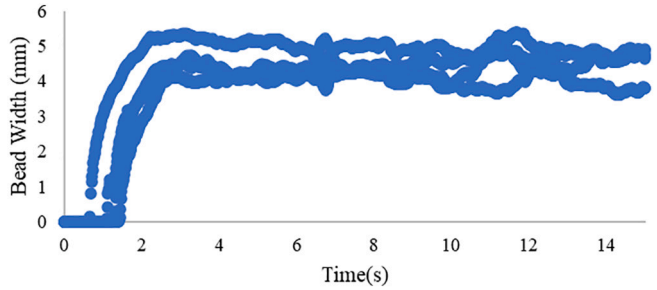

Fig. 11. Experimental data collected with welding speed 6.5 mm/s.

mm bead width. Also, three different rates of gradient speeds (i.e., η in Eq. (4)), 0.1, 0.25, and 0.5 were tested. The results of the trials are shown in Fig. 10. The discussions (in Section 2) on the effects of learning rate are demonstrated in the results. With the lowest rate, the speed adjustment took the longest time to reach and converge to the desired width. The highest rate would lead to severe oscillation, and no convergence within the time allowed. The middle rate was the best in terms of quick convergence and having no obvious overshoot.

Another observation of note is that the controller achieved similar performance and produced similar trends in the bead width regardless of the starting speed. On the other hand, it is surprising to see the estimated bead width had a large variation after 3 s when the bead width was expected to be stabilized. This is a result of the fact that the simulation used data that was generated from experimental trials, and the trials of the stabilized speed (approximately 6.0 mm/s) produced different beads leading to the simulation's width variation, as shown in Fig. 11.

The controller was then applied to real welding tests. Different policies on the gradient descent rate were tried and compared, including constant rate, time-varying rate with an initial value and constant decay, as well as time-varying dampening rate. A constant gradient rate achieved the worst process control performance, including long convergence time and/or unstable control process. As for the time-varying rate, different initial values (ranging from 0.1 to 0.5) and decay rates (ranging from 0.01 to 0.005 per second) had been tried and achieved satisfactory performance for most experimental trials. But in some trials an overcorrection would occur, especially when the network predicted a large bead width. This resulted in the addition of a dampening value for high predicted widths, to avoid overcorrection. The finally employed gradient descent rate begins with an η of 0.25 and reduces 0.01 every second. If the predicted width is above the resulting change is halved to prevent an overshoot. The results are shown in Fig. 12.

The controller was tried with a slow starting speed, 5 mm/s and a high starting speed, 7 mm/s. Even though the controlled bead widths had larger variation in the real experiments than in the simulation, the controller still performed admirably and was able to generate relatively stabilized beads within two to four seconds. Visual inspection of the back bead width shows that both speeds produced desired linear weld, but were more stable and accurate with slower starting speeds. This is likely due to the nature of higher speed welding trials were less consistent than slower starting speeds. A notable result did occur portrayed in the bottom right. There was an early unexpected spike in the weld accompanied by unusual audible and visual signs during the process. The controller was still able to adjust for this variation and

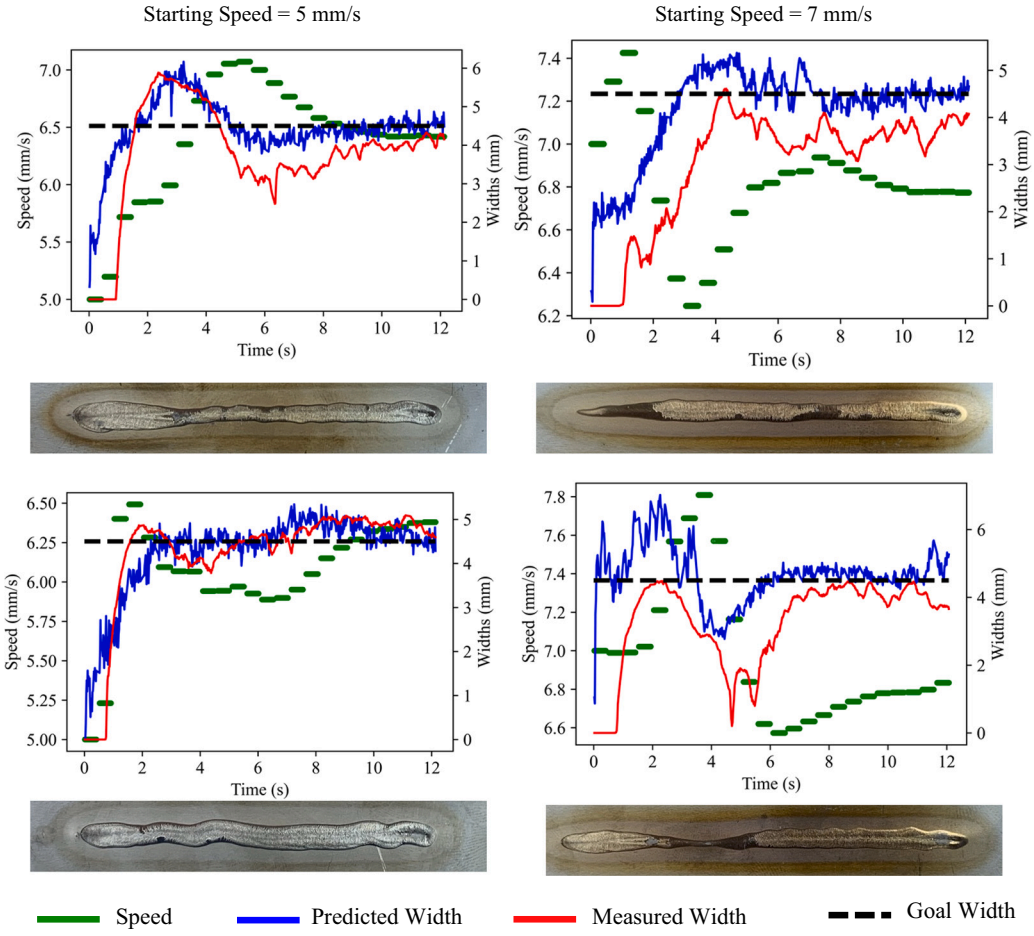

Fig. 12. Results of experimental control tests under two starting speeds: v = 5 mm/s (left) and v = 7 mm/s (right), as well as photographs of completed welds (bottom).

accomplish the goal width.

4. Conclusions

This research presents a hybrid ML framework for real-time bead width prediction and adaptive welding speed adjustment to achieve desired bead width in the robotic GTAW welding process. The ML framework contains a CNN for real-time processing of weld pool images and estimation of bead width, an MLP for correlating welding speed and time to bead width, and an MLP-based gradience descent controller to adaptively adjust speed towards desired bead width. Experimental results confirm the strong agreement between actual bead width values and those predicted by the CNN and MLP. Simulation and experimental studies also demonstrated the effectiveness of the gradient descent controller in adjusting speeds. Different rates of speed adjustment were evaluated and discussed. Future studies will expand the experimental evaluation of the controller by trying different starting speeds and a longer welding time. This configuration on welding speed adjustment will also be extended to the current, path, and orientation changes that are more organic, in order to apply adaptive robotic control solutions to the most difficult welding scenarios.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

This study is supported by National Science Foundation under Grant No. 2024614.

References

[1] Sprovieri J. New technology for robotic welding. In: Assembly. 59; 2016. p. 36–9.
[2] Robotic welding market. Mark Mark; 2021. https://www.marketsandmarkets.com/Market-Reports/robotic-welding-market-248099543.html.
[3] National welding month: time to let everyone in on the secret : resources. Am Weld Soc 2018. https://www.aws.org/resources/detail/national-welding-month-time-to-let-everyone-in-on-the-secret.
[4] Maharjan R. Semi-automatic robot pipe welding system. Häme University of Applied Sciences; 2019.
[5] Lauer S, Wiese P, Dryba S, Flügge W. Data-driven approach for robot-assisted multi-pass-welding thick sheet metal connections. Procedia Manuf 2020;52:95–100. https://doi.org/10.1016/j.promfg.2020.11.018.
[6] Lee D. Robots in the shipbuilding industry. Robot Comput Integr Manuf 2014;30:442–50. https://doi.org/10.1016/j.rcim.2014.02.002.
[7] Shi L, Tian X. Automation of main pipe-rotating welding scheme for intersecting pipes. Int J Adv Manuf Technol 2015;77:955–64. https://doi.org/10.1007/s00170-014-6526-8.
[8] Wu Y, Go JZM, Ahmed SM, Lu WF, Chew CM, Pang CK. Automated bead layout methodology for robotic multi-pass welding. IEEE Int Conf Emerg Technol Fact Autom ETFA October, 2015;vol. 2015. https://doi.org/10.1109/ETFA.2015.7301590.
[9] Micallef K, Fang G, Dinham M. Automatic seam detection and path planning in robotic welding. In: Lect. Notes Electr. Eng. 88. LNEE; 2011. p. 23–32. https://doi.org/10.1007/978-3-642-19959-2_3.
[10] Shen H. A study of welding robot path planning application based on genetic ant colony hybrid algorithm. In: Proc 2016 IEEE Adv Inf Manag Commun Electron Autom Control Conf IMCEC 2016; 2017. p. 1743–6. https://doi.org/10.1109/IMCEC.2016.7867517.
[11] Zhang BC, Wu C, Pang ZX, Li Y, Wang RK. Hybrid global optimum beetle antennae search - genetic algorithm based welding robot path planning. In: 9th IEEE Int. Conf. CYBER Technol. Autom. Control Intell. Syst. CYBER 2019; 2019. p. 1520–4. https://doi.org/10.1109/CYBER46603.2019.9066742.
[12] Fang H, Ong S, Nee A. Robot path planning optimization for welding complex joints. Int J Adv Manuf Technol 2017;90:3829–39. https://doi.org/10.1007/s00170-016-9684-z.
[13] Cho DW, Na SJ, Cho MH, Lee JS. A study on V-groove GMAW for various welding positions. J Mater Process Technol 2013;213:1640–52. https://doi.org/10.1016/j.jmatprotec.2013.02.015.
[14] Zhou L, Xia YJ, Shen Y, Haselhuhn AS, Wegner DM, Li YB, et al. Comparative study on resistance and displacement based adaptive output tracking control strategies for resistance spot welding. J Manuf Process 2021;63:98–108. https://doi.org/10.1016/j.jmapro.2020.03.061.
[15] Yu J. Adaptive resistance spot welding process that reduces the shunting effect for automotive high-strength steels. Metals (Basel) 2018;8. https://doi.org/10.3390/met8100775.
[16] Papacharalampopoulos A, Stavropoulos P, Stavridis J. Adaptive control of thermal processes: laser welding and additive manufacturing paradigms. Procedia CIRP 2018;67:233–7. https://doi.org/10.1016/j.procir.2017.12.205.
[17] Zhang K, Li D, Gui H, Li Z. Adaptive control for laser welding with filler wire of marine high strength steel with tight butt joints for large structures. J Manuf Process 2018;36:434–41. https://doi.org/10.1016/j.jmapro.2018.10.042.
[18] Liu YK, Zhang YM. Model-based predictive control of weld penetration in gas tungsten arc welding. IEEE Trans Control Syst Technol 2014;22:955–66. https://doi.org/10.1109/TCST.2013.2266662.
[19] Masinelli G, Le-Quang T, Zanoli S, Wasmer K, Shevchik SA. Adaptive laser welding control: a reinforcement learning approach. IEEE Access 2020;8:103803–14. https://doi.org/10.1109/ACCESS.2020.2998052.
[20] Wang P, Gao RX, Yan R. A deep learning-based approach to material removal rate prediction in polishing. In: CIRP Ann - Manuf Technol. 66; 2017. p. 429–32. https://doi.org/10.1016/j.cirp.2017.04.013.
[21] Krizhevsky A, Sutskever I, Hinton GE. ImageNet classification with deep convolutional neural networks. Commun ACM 2017;60:84–90. https://doi.org/10.1145/3065386.
[22] Wang Q, Jiao W, Wang P, Zhang YM. A tutorial on deep learning-based data analytics in manufacturing through a welding case study. J Manuf Process 2020:1–12. https://doi.org/10.1016/j.jmapro.2020.04.044.
[23] Kingma DP, Ba JL. Adam: a method for stochastic optimization. 3rd Int. Conf. Learn. Represent. ICLR 2015 - Conf. Track Proc., 2015.
[24] Hastie Trevor, Tibshirani Robert, Friedman J. The elements of statistical learning the elements of statistical Learning data mining, inference, and prediction. 2nd ed. 2009.
[25] Chang Y, Yue J, Guo R, Liu W, Li L. Penetration quality prediction of asymmetrical fillet root welding based on optimized BP neural network. J Manuf Process 2020;50:247–54. https://doi.org/10.1016/j.jmapro.2019.12.022.
[26] Rumelhart DE, Hinton GE, Williams RJ, Rumelhart DE. Learning internal representations by error propagation. In: McClelland JL, et al., editors. Parallel distributed processing: explorations in the microstructure of cognition. Cambridge, MA: MIT Press; 1986.