Seamless Manual-to-Autopilot Transition: An Intuitive Programming Approach to Robotic Welding*

Haruhiko Eto¹ and H. Harry Asada, Member, IEEE¹

Abstract—An intuitive on-site robot programming method for small-lot robotic welding is presented. In current robotic welding, a human operator has to input numerous parameters, including feedrate, swing width, and frequency, by using a teach pendant or a control panel before executing the task. This traditional approach is suitable for mass production, but requires tedious, time-consuming programming, which does not fit low-volume manufacturing, such as shipbuilding. In this paper, a method is developed for acquiring those parameters directly from an on-site human demonstration and seamlessly transitioning from manual operation to automatic control. With this method, a welding worker can directly execute a welding task, and the motion of a welding torch is observed, from which key parameters are identified and the machine performs the rest of the task autonomously. No tedious parameter input is required, but the worker can jump-start the task. The motion of a welding torch is represented as a combination of sinusoidal and linear functions. Discrete Fourier Transform (DFT) and Recursive Least Squares (RLS) estimates are used for identifying the parametric model in real time. Furthermore, an algorithm is developed for determining whether an appropriate estimation result has been obtained and when to switch from manual operation to autonomous control. The method is implemented on a virtual teleoperation system and seamless control transition is demonstrated.

I. INTRODUCTION

Welding is one of the most critical manufacturing processes in heavy industries. Shipbuilding, for example, requires a number of welding operations. Autonomous welding has been studied for a long time. Early works by D. Hardt and his collaborators aimed to regulate a welding process with closed-loop control and adaptive learning methods [1]–[5]. Complete autonomy, however, is difficult due to various factors. More recent approaches include a human in the loop. Teleoperated welding systems have been developed by several authors [6]–[8], and shared control approaches have been investigated [9]. Furthermore, human welding skills have been measured and analyzed offline [10], and used for autonomous control. However, autonomous welding machines are still impractical and difficult to use in shipyards due to the complexity of the environment and the variety of workpieces. On the other hand, portable semi-automatic welding carriages have been used as a practical solution for fillet welding and butt welding instead of expensive intelligent robots [11]. These carriages can track a welding joint at a constant speed with drive wheels and a mechanical guiding system (for instance, self-guiding rollers or guide rails). Thus, these carriages can perform continuous welding along a straight or gentle curved joint. Commercialized welding carriages usually have 1 or 2 active DOF: drive wheels, for moving along the welding joint line, and a torch swing mechanism, for making a specific rotating/swinging motion so-called weaving motion to obtain a desired bead shape and welding penetration. Robotic manipulators, which typically have more than 5 DOF, are another option for this work situation. These manipulators are installed on mobile bases or other mechanisms such as gantries and overhead cranes, and they have sensors for welding seam tracking [12].

In both welding carriages and welding manipulators, a human welder has to program initial welding parameters to perform the task autonomously. For example, in our target applications, Metal Inert Gas (MIG) welding and Metal Active Gas (MAG) welding, where shielding gas and an electrode wire are fed to the welding torch, the welder must program the gas flow rate, wire feeding speed, and arc current. Furthermore, parameters associated with torch motion, such as traveling speed, swing width, must also be specified. However, those parameter values must be adjusted depending on workpiece geometry and other factors. These motion parameters are usually set by manually turning dials and pushing buttons on the control panel of the devices or other type of programming tool. The traditional robot programming task is tedious and time-consuming [13], [14]. The parameter setting process also provides less flexibility for different workpieces and is not intuitive for human welders; therefore, they sometimes input wrong values and then have to modify them later.

Traditional programming is only suitable for simple tasks with simple geometry in general [15]. The walk-through programming method is more intuitive for humans than using teach pendants [16]. The human welder can physically guide the robotic torch and the robot records the trajectory for playback control. However, this approach is still tedious when the task or the geometry is complex. The Programming by Demonstration (PbD) approach using machine learning is an intuitive method which generalizes a task from observations of a human demonstration [17]. PbD has been used for various manipulation tasks such as a pick-and-place task using a virtual reality teaching interface [18], [19], peg-in-hole and door opening [20], complex household tasks [21]. However, these learning methods still require a significant training process.

This paper proposes a different type of programming method, where manual welding operation and autonomous programming...
control are smoothly switched in real time. Unlike conventional methods, the welding job seamlessly continues from the programming phase to the task execution phase. The proposed method eliminates tedious on-site parameter setting, and allows a robotic welder to execute the task as intended by the human welder. Once autonomous mode activates, the human welder is released and can work on other tasks. In the following sections, the seamless manual-to-autopilot transition is detailed, and algorithms are developed for analyzing human data and switching the control mode without stopping the process.

II. APPROACH: SEAMLESS MANUAL TO AUTOPILOT TRANSITION

We propose a programming method where manual and autonomous control are time-shared, focusing on a seamless transition from manual to autopilot mode. In this approach, a skilled human welder manually performs a welding operation for the first few segments. A real-time estimation algorithm will be developed to analyze the data and identify the intended operation of the human welder. As soon as all the parameters have been identified, the system takes over the welding operation and completes the task autonomously. This seamless transition from manual to autonomous operations has a few distinct features:

- A human welder can jump-start a welding operation without inputting any parameters. Optimal welding parameters are not explicitly known for human welders, but they can demonstrate best operations by manipulating the torch. This can make the human-robot interface intuitive and easy to use.
- As addressed previously, the parameter setting method of today’s welding carriages and welding robots is not flexible. The proposed method can realize a broader class of torch motions and patterns, so that a human welder can express subtle movements and skillful operations.
- The human demonstration for programming and the autonomous task execution are seamlessly connected. Once the welding starts, it never stops.
- The human welder is released from the welding operation, once the robot confirms that it has gained enough information from the human. The human can work on other welding robots to increase productivity.

Fig. 1(a) shows one of the realizations of the above method. The human welder holds a real welding torch and performs a welding operation. The welding torch is also attached to a robot, which is back-drivable by design, or controlled in zero-G mode, so that the human can freely move the torch. The movement of the torch is measured with the sensors on the robot (e.g., encoders of the joints). From the measured data, its trajectory and motion pattern are identified in real time. As a sufficient amount of data is obtained, the robot takes over the task to complete the welding. Fig. 1(b) shows another realization where the human welder uses a mock-up welding torch that is instrumented with position sensors so that the motion of the torch is measured. The measured human movement is transmitted to the welding robot that reproduces the human motion. As before, the robot takes over the task once it gains enough data. This realization is a type of teleoperation, requiring a virtual environment so that the human welder can observe the weld pool and the workpieces being joined. In both cases, the robot must be able to track the welding line autonomously with sensors that detect the welding line.

III. METHOD

In this section, we present a parametric model of welding torch motion and develop an algorithm for estimating the parameters from a human welder’s movements.

A. Welding Trajectory Model

The position and orientation of the welding torch are represented with a vector defined as

\[
\mathbf{x} = [x_1 \ x_2 \ x_3 \ x_4 \ x_5]^\top = [x \ y \ z \ \alpha \ \beta]^\top
\]

Since the nozzle of the welding torch is axisymmetric as shown in Fig. 2(a), the total number of DOF is five. We assume that the welding line is piecewise straight and uniform, so that the required torch motion does not change in each segment of straight line. Typically a human welder moves the torch at a constant traveling speed, or feedrate,
combined with an oscillatory motion: weaving. We thus model each torch’s position and orientation \( x_i(t) \) as a combination of a sinusoidal function and a linear function:

\[
x_i(t) = A_i \sin(2\pi f_i t + \psi_i) + a_i t + b_i
\]

(1)

where \( A_i \) is amplitude, \( f_i \) is frequency, \( \psi_i \) is phase angle, \( a_i \) is linear velocity, and \( b_i \) is a constant bias. We assume that the \( x \) axis and \( y \) axis correspond to the welding travel direction and the swing direction, respectively.

This proposed model \( x(t) \) can represent stringer bead paths (i.e., straight bead paths along a welding joint) and typical weave bead paths commonly used by welders, convex, concave, figure 8, circular, and so on [22]. Fig. 2(b) shows an example of three-dimensional torch paths, which can be represented by (1). Fig 3 shows examples of weaving patterns projected onto the \( xy \)-plane. The time profiles of \( x(t) \) and \( y(t) \) forming those patterns are also shown in the figure. To form these path curves, the following two conditions must be satisfied. First, \( y(t) \) must be symmetric with respect to its center line, which means \( a_2 = 0 \). Second, the ratio of frequencies \( f_x / f_y \) must exactly be \( f_x / f_y = 2 \) for patterns (a), (b), (c), and (d), and \( f_x / f_y = 1 \) for pattern (e). Pure sinusoidal curves, which are not involved in Fig. 3, are also an example of typical welding paths and its frequency ratio is \( f_x / f_y = 0 \). If the frequency ratio is slightly different from these standard values, the path shape will gradually collapse with time as shown in Fig. 4, which shows a convex path with the frequency ratio \( f_x / f_y = 2.02 \). This matter will be considered in the subsection III-B.

The trajectories (1) can be represented in a discrete-time form with a constant sampling interval \( \Delta t \) and a sample index \( k \) as follows:

\[
x_i(k) = A_i \sin(2\pi f_i \Delta t k + \psi_i) + a_i \Delta t k + b_i
\]

(2)

This trajectory function can be re-parameterized as:

\[
x_i(k) = A_i \sin(2\pi f_i \Delta t k + \psi_i) + a_i \Delta t k + b_i = \begin{bmatrix} \theta_{i,1} \\ \theta_{i,2} \\ \theta_{i,3} \\ \theta_{i,4} \end{bmatrix}
\]

(3)

where

\[
s_{i,k-1} = \sin(2\pi f_i \Delta t(k - 1))
\]

(4)

\[
c_{i,k-1} = \cos(2\pi f_i \Delta t(k - 1))
\]

(5)

\[
\theta_{i,1} = A_i \cos(2\pi f_i \Delta t) \cos \psi_i - \sin(2\pi f_i \Delta t) \sin \psi_i
\]

(6)

\[
\theta_{i,2} = A_i \sin(2\pi f_i \Delta t) \cos \psi_i + \cos(2\pi f_i \Delta t) \sin \psi_i
\]

(7)

\[
\theta_{i,3} = a_i \Delta t
\]

(8)

\[
\theta_{i,4} = a_i \Delta t + b_i
\]

(9)

\( x_i(k) \) corresponds to the torch tracking data stream obtained by a human demonstration. The problem is to estimate unknown parameters in (3), namely the frequency \( f_i \) and the parameter vector \( \theta_i \). Considering its nonlinearity, our algorithm estimates \( f_i \) using DFT prior to estimating \( \theta_i \) with RLS.

B. DFT and Frequency Correction

Since DFT needs a certain number of data points, the tracking data is required to be stored in a buffer. In our method, the data length for DFT is adaptively determined by detecting peaks of the motion tracking data stream. Our algorithm detects and counts only the peaks of \( y(t) \) to determine the data length since the frequency ratio \( f_x / f_y \) must exactly be 0, 1, or 2 as discussed in the previous subsection, which means that the number of waves of \( x(t) \) is always larger than or equal to that of \( y(t) \) within the...
Fig. 4. A convex path curve moving along the x direction with the frequency ratio $f_x/f_y = 2.02$. The small difference from the standard ratio collapses the curve shape.

Fig. 5. The left graph shows a synthetic test signal, which consists of a sinusoid and a linear component with random noise. The frequency is 3 Hz and the data length is predefined as 1 second. The right graph shows the frequency estimation result after zero padding. The result has two peaks in the graph. The peak around 0 Hz is because of the linear components, and the other peak, marked with a red dot, is the answer value.

data length (or there is no wave components in $x(t)$ when $f_x/f_y = 0$). The Hann window is applied to the stored dataset and zero padding is used to increase the resolution of estimated frequency. Because the tracking data have a linear component, the DFT result has two peaks as shown in Fig. 5. The peak around 0 Hz is because of the linear components, and then the other peak is the answer. The resultant peak value is identified as the estimated frequency. After $f_x$ and $f_y$ are estimated, their ratio $f_x/f_y$ is computed and classified to the closest standard value (0, 1, or 2) by thresholding. $f_x$ is then replaced as $f_x = 0$, $f_x = f_y$, or $f_x = 2f_y$. The estimated result of $f_y$ by DFT is usually more accurate than that of $f_x$ since $y(t)$ has no linear components; this is why we replace $f_x$ with a value defined by $f_y$. Thus we finally obtain estimated frequencies $f_x$ and $f_y$. $f_x$ and $f_y$ should be constant if a workpiece is uniform, but the frequencies of human’s torch operation can fluctuate or gradually shift. To update $f_x$ and $f_y$ at each sampling step, the sliding DFT algorithm [23] is used.

C. RLS-based Parameter Estimation

Once the estimated frequencies are obtained, the unknown parameter vector $\hat{\theta}_i$ can be estimated by the RLS. Prior to executing the RLS, initial estimates of the parameters are obtained from the data used for the DFT computation. The batch least squares algorithm is applied to the stored data, in which the frequency is replaced by the estimated frequency $\hat{f}_i$ obtained by DFT. This method allows us to provide initial values for the unknown parameters, i.e. $\hat{\theta}_i(0)$. After the initial estimated values are obtained, the sliding DFT updates $\hat{f}_i$ and feed it to the RLS algorithm. We apply the following RLS algorithm with forgetting factor to estimate $\hat{\theta}_i(k)$:

$$\hat{\theta}_i(k) = \hat{\theta}_i(k-1) + \frac{P_i(k-1)\hat{\varphi}_i(k)}{\rho_k + \hat{\varphi}_i^T(k)P_i(k-1)\hat{\varphi}_i(k)} [x_i(k) - \hat{\varphi}_i^T(k)\hat{\theta}_i(k-1)]$$  \hspace{1cm} (10)

$$P_i(k) = \frac{1}{\rho_k} [P_i(k-1) - \frac{P_i(k-1)\hat{\varphi}_i(k)\hat{\varphi}_i^T(k)P_i(k-1)}{\rho_k + \hat{\varphi}_i^T(k)P_i(k-1)\hat{\varphi}_i(k)}]$$  \hspace{1cm} (11)

where $P_i(k)$ is a covariance matrix, $\rho_k$ is a forgetting factor ($0 < \rho_k \leq 1$), and $P_i(0)$ is a positive definite matrix. Once we obtain $\hat{\theta}_i(k)$, the reference trajectory can be generated as:

$$\hat{x}_i(k \mid \hat{\theta}_i) = \varphi_i^T(k)\hat{\theta}_i.$$  \hspace{1cm} (12)

The reference trajectory $\hat{x}_i$ is fed to the controller of the robotic welding system so that the welding torch follows it. Because of the forgetting factor, the RLS algorithm outputs the parameters that gives more weight to recent data and outputs better fitting results than the old data. This method makes a smooth connection between the input trajectory and the reference trajectory.

There are several online algorithms for nonlinear parameter estimation, such as Extended Kalman Filter (EKF). As shown in Fig. 6(a) and (b), the EKF algorithm is able to compute the unknown parameters of a function, which consists of a sinusoid and a linear function. The EKF requires no batch processing, unlike the DFT combined with the least squares estimation, so the estimation can be completed quickly, in a few seconds. However, the performance of EKF is sensitive to the signal properties, as shown in 6(c) and (d). While the test signal in 6(c) is similar to the signal in (a), except for the phase, the estimated parameters does not converge well (Fig. 6(d)) and the curve fitting fails (Fig. 6(c)). To assure a stable estimation result, we used the combination of the DFT and the RLS in this paper.

D. Verification of the estimated result

The estimated trajectory $\hat{x}_i$ is required to have not only a smooth connection but also a global similarity to the input trajectory $x_i$ made by the human welder. In order to guarantee the smooth connection, we observe the squared errors of the trajectories, i.e. $(\hat{x}_i - x_i)^2$. This squared error should be smaller than a predefined threshold value $\epsilon_i$. For the global similarity, the estimated amplitude $\hat{A}_i$ is compared with the average peak value of the input trajectory $A_i$. The peak amplitudes of the input trajectory are obtained by the peak detection algorithm, and the average value $A_i$ is recursively computed. The estimated amplitude is computed by using the estimated parameters $\hat{\theta}_{i,1}$ and $\hat{\theta}_{i,2}$, as $\hat{A}_i = \sqrt{\hat{\theta}_{i,1}^2 + \hat{\theta}_{i,2}^2}$. The variance $\sigma_{A,i}^2$ is also recursively obtained with $\hat{A}_i$. If $(\hat{A}_i - A_i)^2 < \sigma_{A,i}^2$ is satisfied, it is considered that the global similarity is achieved. When both $(\hat{x}_i - x_i)^2 < \epsilon$ and $(\hat{A}_i - A_i)^2 < \sigma_{A,i}^2$ are satisfied simultaneously, it is considered that the estimation is reproducing the trajectory well. Note that the estimated parameters $\hat{\theta}_i$ do not usually converge because the human hand motion inevitably has
random fluctuations. That is why we use the instant values of \((\hat{x}_i - x_i)^2\) and \((A_i - \bar{A}_i)^2\) instead of the parameter convergence analysis.

IV. SYSTEM IMPLEMENTATION

A. Algorithm implementation

The algorithms described in the previous section are implemented as shown in the flowchart in Fig. 7. The main part of the software consists of five blocks: the frequency estimator block, the batch least squares (LS) block, the RLS block, the trajectory generator block, and a verification block. The motion of a human welder is first provided to the frequency estimator block. This block stores a sequence of measured motion \(x(t)\) in a buffer and estimates the frequency \(f_0\). The value of \(f_0\) is sent to the batch LS block. The batch LS block computes the initial values of the parameters \(\theta(0)\) by using \(x(t)\) in the buffer and \(f_0\). Next, \(f_0\) and \(\theta(0)\) are fed to the RLS block. After these initial values are computed, the RLS block recursively analyzes the motion tracking data \(x(t)\) streamed from a motion capture sensor. \(f(t)\) obtained by the sliding DFT is given to the RLS block at each sampling step. The result \(\hat{x}(t)\) is then evaluated at the verification block. The squared error \((\hat{x}_i - x_i)^2\) and the amplitude difference \(A_i - \bar{A}_i\) are evaluated, and the estimate is validated when these two values become lower than cutoff thresholds. The robot controller then outputs the control input \(u(t)\) to replace the human-generated motion command.

B. Virtual teleoperation user interface

The above computation and control software was installed on the experimental platform shown in Fig. 8. A human operator moves a mock-up welding torch, whose nozzle position is measured with Leap Motion [24] connected to a computer via USB. A virtual torch, which is a torch model in a three-dimensional virtual space, moves according to the mock-up torch movement as a master-slave teleoperation system. A 3D human hand model is also displayed as an indicator of manual operation mode. This virtual hand disappears when the autopilot mode enables, which signifies that the control input has been switched from the manual mode, with the mockup torch, to the autopilot mode, with the reference trajectory generated using the estimated parameters. In the autopilot mode, the virtual torch follows the reference trajectory by PID control. The user can observe the virtual torch movement from three different viewpoints displayed on a monitor. This multi-camera visual feedback can be realized in a physical robot system by attaching cameras onto different places on the robot body. The trajectories of the mock-up torch and the virtual torch are displayed on a window shown in the upper left corner of the screen, in order to give the user visual feedback of the torch movement and the estimation result. By watching this window, the user can monitor whether the transition is executed well, which means that the estimated trajectory is connected smoothly to the manual trajectory.

V. PRELIMINARY TEST RESULTS

In this section, we present preliminary test results to show the potential of the proposed method. In the following three cases, the number of signal peaks for determining the DFT window length is 8, the forgetting factor is 0.95, and \(\epsilon = 0.01 \cdot \bar{A}_i\).

A. Test with Synthetic Data

Prior to hand motion data, the developed algorithms were tested with a two-dimensional synthetic signal. The signal simulates a convex path, and consists of two functions with random noise: \(x(t) = 0.6 \sin(2\pi \cdot 2.0 \cdot t + \pi/2) + 3.0t + 1.0\) and \(y(t) = 18 \sin(2\pi \cdot 1.0 \cdot t + \pi)\). Fig. 9(a)-(c) show the estimation results of the synthetic signal. In (a), the trajectory in the travel direction \(x\) is shown. The trajectory in the swing direction \(y\) is shown in (b). The test trajectories \(x(t)\) and \(y(t)\) are plotted as blue solid lines, and the estimated trajectories \(\hat{x}(t)\) and \(\hat{y}(t)\) generated by the developed algorithm are plotted as red dashed lines in (a) and (b). The test signal was streamed to the algorithm and the data were stored in the buffer for the DFT until the number of the peaks of \(y(t)\) reached eight. After the initial frequency was computed by DFT, the batch LS estimated the other initial parameters. The RLS parameter estimation then started (at \(t = 3.80\))
Fig. 9. (a)-(c) Test results of a synthetic convex signal. (d)-(f) Test results of a concave signal made by human hand motion. (g)-(i) Test results of a circular signal made by human hand motion.

B. Test with Hand Motion Data

By using the developed virtual experimental platform, we demonstrated the real-time analysis of the motion tracking data and the seamless control transition. A concave path and a circular path were tested. In the virtual environment, we assumed a situation in which the task executed by the robot is flat welding, and the torch movement in the vertical direction (z direction) is constrained by a guiding system (e.g., a mechanical joint tracking mechanism or a sensor-based tracking system). Because of this constraint, the torch height is constant during the welding task, and the torch movement is considered as a two-dimensional motion. A human demonstrator in this test tried to keep a path uniform.

Fig. 9 (d)-(f) are the results of a concave path. Although the demonstrator intended to make a uniform path, the path shape has random distortions. However, our algorithm was successfully able to create the reference trajectories and the two-dimensional path (shown as red dashed curves) connecting smoothly with the human hand motion (the blue curves). It took 0.90 sec to complete the RLS process. Fig. 9 (g)-(i) show the results of a circular path. The generated trajectories and the resulting two-dimensional path are smoothly connected and similar to the input signals made by the human demonstrator. The RLS process time was 1.75 sec in this case.

Fig. 10(a) shows the virtual environment screen at an instant immediately after the control transition. The virtual torch trajectory \( y(t) \) seamlessly follows the reference trajectory.
tory $\hat{y}(t)$ by PID control. The manual mode indicator (the blue hand model) is disappearing in order to inform the user that the control transition has just been executed. After the transition, the virtual torch works alone automatically (Fig. 10(b)).

VI. CONCLUSIONS

In this paper, we presented a novel programming method for welding robots which allows a human operator to determine the torch motion pattern from manual teleoperation. The manual teleoperated welding task and the autonomous welding task are seamlessly connected and transitioned. The human welder can start the welding task manually by directly manipulating the torch or using the teleoperation system with a hand motion capture device. This manual operation process is a robot programming process as well. This means that the welder can define the robot torch movement intuitively using their motion. We modeled continuous welding movement as a combination of a sinusoidal function and a linear function, and demonstrated that this model can form typical welding movement patterns. The DFT and RLS algorithms were tested to estimate unknown parameters of the defined model in two-dimensional cases. We discussed that the ratio of the frequencies $f_x$ and $f_y$ have to satisfy the particular condition, otherwise, the welding path shape will collapse. Our RLS-based algorithm was tested with a synthetic path signal and actual human motion data, and it was confirmed that our algorithm successfully generated the paths. Also, the control transition can be achieved within a few seconds. Although the DFT window length and the RLS process time depend on a torch speed, a swing frequency, and a workpiece size, our proposed method can allow a welder to program the autonomous welding condition much faster than using a traditional programming tool. We also prototyped the virtual experimental platform and implemented the algorithm on it.

As the future direction of development, our algorithm will be implemented on a physical robotic welding system. The performance and the effectiveness of our proposed algorithm will be studied with various human subjects and different welding path patterns. A verification and approval process before the control transition based on the welder’s judgment about the quality of the parameter estimation should also be developed. Also, the control transition should be bidirectional between human and robot, in order for the welder to modify the robot’s movement when the robot needs help. Our approach could also be adapted for other manufacturing processes that consist of repetitive motion patterns of the end effector, such as painting, gouging, gluing, cutting, and grinding.

REFERENCES

[1] C. Doumanidis and D. Hardt, “A model for in-process control of thermal properties during welding,” Journal of dynamic systems, measurement, and control, vol. 111, no. 1, pp. 40–50, 1989.
[2] A. Tarn and D. Hardt, “Weld pool impedance for pool geometry measurement: stationary and non-stationary pools,” Journal of dynamic systems, measurement, and control, vol. 111, p. 545, 1989.
[3] C. Doumanidis and D. E. Hardt, “Multivariable adaptive control of thermal properties during welding,” Journal of dynamic systems, measurement, and control, vol. 113, no. 1, pp. 82–92, 1991.
[4] A. Suzuki, D. Hardt, and L. Valavanis, “Application of adaptive control theory to on-line gta weld geometry regulation,” Journal of Dynamic Systems, Measurement, and Control, vol. 113, no. 1, pp. 93–103, 1991.
[5] J. Song and D. Hardt, “Closed-loop control of weld pool depth using a thermally based depth estimator,” WELDING JOURNAL–NEW YORK, vol. 72, pp. 471–s, 1993.
[6] D. Ni, A. W. W. Yew, S. K. Ong, and A. V. C. Nee, “Haptic and visual augmented reality interface for programming welding robots,” Advances in Manufacturing, vol. 5, no. 3, pp. 191–198, Sep 2017.
[7] Y. Liu and Y. Zhang, “Toward welding robot with human knowledge: A remotely-controlled approach,” IEEE Transactions on Automation Science and Engineering, vol. 12, no. 2, pp. 769–774, April 2015.
[8] S. J. Chen, N. Huang, Y. K. Liu, and Y. M. Zhang, “Machine assisted manual torch operation: system design, response modeling, and speed control,” Journal of Intelligent Manufacturing, vol. 28, no. 6, pp. 1249–1258, Aug 2017.
[9] M. Ramacciotti, M. Milazzo, F. Leoni, S. Roccella, and C. Stefanini, “A novel shared control algorithm for industrial robots,” International Journal of Advanced Robotic Systems, vol. 13, no. 6, pp. 1–10, 2016.
[10] Y. Liu and Y. Zhang, “Human welder 3-d hand movement learning in virtualized GTAW: Theory and experiments,” in Transactions on Intelligent Welding Manufacturing, S. Chen, Y. Zhang, and Z. Feng, Eds. Singapore: Springer Singapore, 2019, pp. 3–25.
[11] D. Lee, “Robots in the shipbuilding industry,” Robotics and Computer-Integrated Manufacturing, vol. 30, no. 5, pp. 442 – 450, 2014.
[12] A. G. Dharmawan, A. A. Vibhute, S. Foong, G. S. Soh, and K. Otto, “A survey of platform designs for portable robotic welding in large scale structures,” in 2014 13th International Conference on Control Automation Robotics Vision (ICARCV), Dec 2014, pp. 1683–1688.
[13] S. V. Gray, J. R. Wilson, and C. S. Syan, “Human control of robot motion: orientation, perception and compatibility,” in Human-Robot Interaction. Taylor & Francis, 1992, pp. 48–64.
[14] E. C. Morley and C. S. Syan, “Teach pendants: how are they for you?” Industrial Robot: the international journal of robotics research and application, vol. 22, no. 4, pp. 18–22, 1995.
[15] V. Villani, F. Pini, F. Leali, C. Secchi, and C. Fantuzzi, “Survey on human–robot interaction for robot programming in industrial applications,” IFAC-PapersOnLine, vol. 51, no. 11, pp. 66 – 71, 2018, 16th IFAC Symposium on Information Control Problems in Manufacturing INCOM 2018.
[16] M. H. Ang, W. Lin, and S. Lim, “A walkthrough programmed robot for welding in shipyards,” Industrial Robot: the international journal of robotics research and application, vol. 26, no. 5, pp. 377–388, 1999.
[17] S. Mnch, J. Kreuziger, M. Kaiser, and R. Dillmann, “Robot programming by demonstration (pdd) – using machine learning and user interaction methods for the development of easy and comfortable robot programming systems,” in In Proceedings of the 24th International Symposium on Industrial Robots, 1994, pp. 685–693.
[18] J. Aleotti, S. Caselli, and M. Reggiani, “Leveraging on a virtual environment for robot programming by demonstration,” Robotics and Autonomous Systems, vol. 47, no. 2, pp. 153 – 161, 2004.
[19] J. Aleotti and S. Caselli, “Robust trajectory learning and approximation for robot programming by demonstration,” Robotics and Autonomous Systems, vol. 54, no. 5, pp. 409 – 413, 2006.
[20] M. Kaiser and R. Dillmann, “Building elementary robot skills from human demonstration,” in Proceedings of IEEE International Conference on Robotics and Automation, vol. 3, April 1996, pp. 2700–2705 vol.3.
[21] M. Padowitz, S. Knoop, R. Dillmann, and R. D. Zollner, “Incremental learning of tasks from user demonstrations, past experiences, and vocal comments,” IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), vol. 37, no. 2, pp. 322–332, April 2007.
[22] F. M. Marlow, Welding Know-How. Metal Arts Press, 2012.
[23] E. Jacobson and R. Lyons, “The sliding dft,” IEEE Signal Processing Magazine, vol. 20, no. 2, pp. 74–80, March 2003.
[24] Leap motion. Accessed: 2019-05-10. [Online]. Available: https://www.leapmotion.com
[25] Unity. Accessed: 2019-05-10. [Online]. Available: https://unity.com