A Point Cloud-Based Method for Automatic Groove Detection and Trajectory Generation of Robotic Arc Welding Tasks

Rui Peng, David Navarro-Alarcon, Victor Wu and Wen Yang

Abstract— **In this paper, in order to pursue high-efficiency robotic arc welding tasks, we propose a method based on point cloud acquired by an RGB-D sensor. The method consists of two parts: welding groove detection and 3D welding trajectory generation. The actual welding scene could be displayed in 3D point cloud format. Focusing on the geometric feature of the welding groove, the detection algorithm is capable of adapting well to different welding workpieces with a V-type welding groove. Meanwhile, a 3D welding trajectory involving 6-DOF poses of the welding groove for robotic manipulator motion is generated. With an acceptable error in trajectory generation, the robotic manipulator could drive the welding torch to follow the trajectory and execute welding tasks. In this paper, details of the integrated robotic system are also presented. Experimental results prove application value of the presented welding robotic system.**

I. INTRODUCTION

Nowadays, industrial robotic manipulators are used extensively in many factories around the world, with the teach-playback method dominating the robotic welding field. More specifically, in order to manipulate a robotic arm for welding tasks, a human operator needs to set every path point with precise 3D positioning and 3D orientation in advance. However, although CAD-based approaches [1] can generate rather accurate welding trajectories, they still require a great deal of human intervention, as with teach-playback method. These conventional robotic welding applications are already unable to cope with the growing welding demand of the construction industry.

To improve robotic welding efficiency, sensors are considered to assist manipulators in automatically locating the welding groove. Prevailing sensors involve vision sensors [2]–[4], RGB-D sensors [5]–[7], infrared sensors and laser sensors. But laser sensors are much more expensive than vision sensors. To control cost in research, vision sensors are more appealing. During actual robotic welding tasks, it is essential to plan a high-precise trajectory [8] of the welding groove between two workpieces to achieve acceptable welding quality in complex and unpredictable situations. Recently, 2D imaging-based methods have become useful in assisting robotic welding in various industrial environments [9]. But with respect to common welding tasks, 2D vision sensors could capture only one frame image of the workpieces. An image acquisition system using a CCD video camera, is established by L. Nele et al. [10] for real-time weld seam tracking.

However, 2D imaging processing algorithms, which rely on color information, cannot deal with dramatic environmental brightness variation, particularly in the welding fields. Researchers prefer 3D vision sensors which provide abundant information of the welding environment. Furthermore, with advancement of the Point Cloud Library (PCL) [11]- specifically designed for 3D point cloud processing, it is possible to extract and locate the welding seam region in the surface point cloud of welding workpieces. The stereo camera as a 3D sensor, is used to realize welding seam reconstruction and tracking [12]. The tracking and planning accuracy is good for welding requirements. An unorganized point cloud-based edge and corner recognition approach [13] is proved for applicable robotic welding. Also the algorithm has accuracy beyond some of relevant algorithms on 3D point cloud processing. Some methods which combine depth data and RGB images have been proposed [14]. These methods implement low-cost 3D sensors (e.g. Intel RealSense) [15]. Through integrating an RGB-D sensor into the robotic system [16], the controller is able to cope with the dynamic welding environment. Li et al. [17] proposed a welding groove detection approach by an RGB-D sensor. The approach employs RGB images to recognize the weld groove and a point cloud to acquire the pose of the targeted weld groove.

One of the main problems in the aforementioned research works based on 3D sensors is that their experimental results lack sufficient testing of different types of welding workpieces; moreover, the targeted welding groove is simple. Inspired by [18], the proposed method in this paper focuses on welding groove detection and 3D motion trajectory generation. Experimental results involve three aspects: runtime efficiency, groove detection accuracy, and trajectory execution. The resulting performance of the system on four types of workpieces with V-type grooves proves feasibility in actual welding applications.

The main contributions of this paper are as follows:

1) Develop an integrated intelligent robotic system to automatically execute welding tasks without much human intervention.
2) Propose a point cloud based welding groove detection algorithm for unpredictable workpieces.
3) Implement the automatic 3D welding trajectory generation method on a 6-DOF robotic arm.

This work is supported in part by the Chinese National Engineering Research Centre for Steel Construction (Hong Kong Branch) at The Hong Kong Polytechnic University under grant BBV8, in part by the Research Grants Council of Hong Kong under grant 142039/17E and in part by The Hong Kong Polytechnic University under grant G-YBYT.

RP, DNA, and VW are with The Hong Kong Polytechnic University, Kowloon, Hong Kong. Corresponding author: dna@ieee.org

WY is with the Wuhan University, Hubei, People's Republic of China.

II. ROBOT SYSTEM IMPLEMENTATION

The integrated robotic system contains a robotic manipulator (Universal Robot 3), an RGB-D camera (Intel Realsense D415), and a welding torch. The end-effector and whole experimental platform are shown in Fig. 1. The welding torch is tightly installed on the end-wrist of the robotic manipulator by a supportive metal structure aimed at stable welding execution. In addition, the RGB-D camera is physically connected to the welding torch instead of at a fixed position within the welding platform. The RGB-D camera moves with the welding torch. This design enables the system to flexibly cope with unpredictable welding situations. It is also convenient for hand-eye calibration to obtain the accurate transformation matrix from the camera coordinate to the UR3 base coordinate.

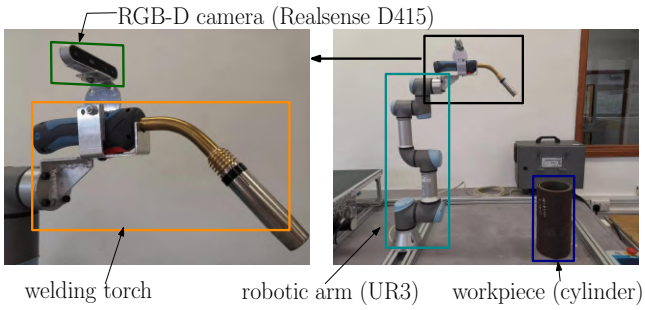

Fig. 1. (Left) End-effector of the proposed welding robotic system comprises an RGB-D camera and welding torch. The camera is situated above the welding torch. (Right) Experimental welding platform.

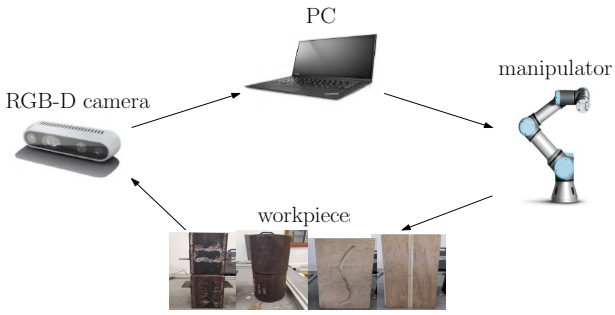

Fig. 2. Workflow of the proposed robotic system: (1) the RGB-D camera captures the surface point cloud of the workpiece; (2) the welding groove detection algorithm locates the welding groove region in the input point cloud; (3) the trajectory generation method processes the groove point set and outputs a 3D welding trajectory; (4) the manipulator automatically executes the welding motion while tracking the generated trajectory.

The system running procedure is shown in Fig. 2 in which the PC with ubuntu 16.04 is a Lenovo-Thinkpad whose CPU uses Intel i5. ROS [19], An open-source robot operating system is used to build a software framework that makes it convenient to develop algorithm modules. The welding groove detection algorithm completely relies on PCL to process the point cloud and extract the geometric feature. The Moveit [20] package, as one part of the software system, is used to control the UR3 robotic arm with motion planning and collision avoidance.

III. WELDING GROOVE DETECTION

The welding groove detection algorithm extracts the groove region by computing the geometric feature of the input point cloud, which represents the surface profile of the welding workpiece. The geometric feature is defined as the surface variation. In other words, the extent of surface slope change. Compared with flat and smooth regions, the groove region has higher surface variation (see Fig. 3).

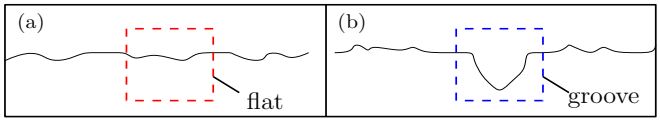

Fig. 3. Geometric feature diagram of surface. (a) Flat region. (b) Groove region.

In order to mathematically describe the surface variation, an efficient 3D feature histogram in the algorithm was designed. Furthermore, a surface variation descriptor is computed for each point of the input point cloud. By setting a threshold value for the descriptor, the groove region can be separated from other regions.

A. Input Point Cloud Preprocessing

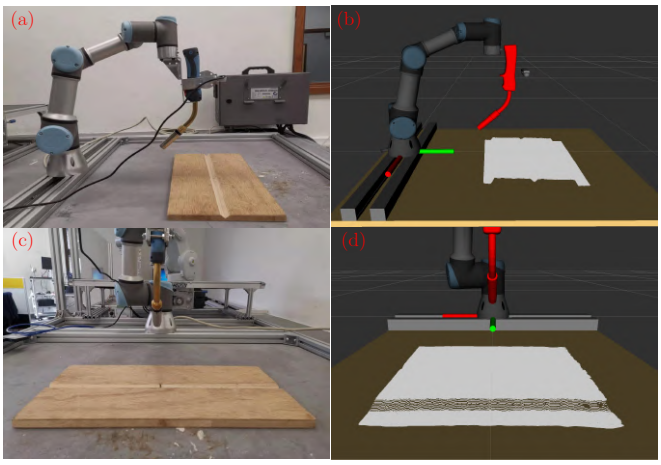

Fig. 4. The RGB-D camera captures the surface point cloud of the workpiece with a straight-line welding groove. (a) A robotic manipulator moves so that the camera can capture the complete point cloud. (b) The surface point cloud is shown in the simulation (Rviz). (c) The actual workpiece is viewed from the side. (d) The raw point cloud is organized.

First, the RGB-D camera attached to the robotic arm moves to an appropriate position and captures one frame of a raw point cloud which covers the whole surface of the workpiece, as shown as Fig. 4. Then the raw point cloud is taken as an input of the welding groove detection algorithm.

In general, there is noise data in the raw point cloud owing to camera hardware factors. Thus, the point cloud needs smoothing before proceeding to the next step. PCL provides a Moving Least Squares (MLS) surface reconstruction method to smooth the point cloud surface and reduce noisy data. Fig. 5 shows the before and after results of the smoothing process for the point cloud of testing the welding workpieces

in Fig. 4. To some extent, it demonstrates that the point cloud surface becomes more even and gentle after smoothing.

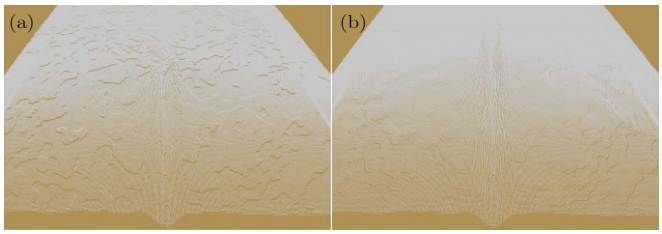

Fig. 5. Raw point cloud smoothing. (a) Point cloud before smoothing has too many noisy regions. (b) Point cloud after smoothing has more of an even surface.

Surface normal computation is an important fundamental part of welding groove detection. Also, PCL offers a mathematical method to estimate the surface normal of each point. Theoretically, given a point cloud cluster, computing one point normal is actually a problem of estimating a normal of a plane tangent to the point cloud surface. Of course, the plane must pass through this point. Simply put, one point's neighbor (a sphere with a constant radius and the point being the sphere's center) is a small cluster of a point cloud which can fit onto a plane. Therefore, normal of this plane is regarded as normal of the point.

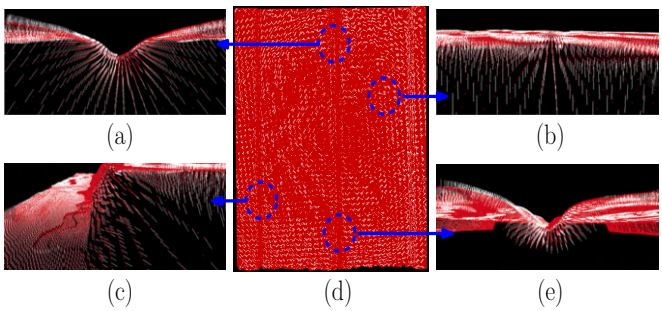

Fig. 6. Surface normal map. Each white arrow represents the normal of each point. (a) Groove region. (b) Flat region. (c) Edge region. (d) Surface point cloud. (e) Groove region.

Then, a normalized map of the surface normal (shown in Fig. 6) for the after-smoothing point cloud is obtained by iterating every point with least-square plane fitting. Moreover, each point normal $\vec{u_i}$ shown as a white arrow in Fig. 6 is a unit vector defined as:

$$\vec{u_i} = [x_i, y_i, z_i]^T, \quad \sqrt{(x_i)^2 + (y_i)^2 + (z_i)^2} = 1 \qquad (1)$$

where i represents the i-th point in the organized point cloud.

Inspired by [21] regarding object recognition, a descriptive method called a groove feature histogram (GFH) was designed to quantify surface variation extent. There are two types of GFH considered for each point in a point cloud: local GFH and global GFH.

B. Local GFH

The neighbor of each point is defined as a sphere (the point is its neighbor's central point) with a constant radius r. Within this neighbor, the kdTree search method is used to find other neighbor points (the Euclidean distance to the central point is not more than r). Neighbor points are regarded as an individual point set where the normal of each point can be obtained from the surface normal map (Fig. 6). Then the central point is paired with every other neighbor point and every pair has two unit vectors which form one included angle. Fig. 7 shows one point's neighbor with one pair.

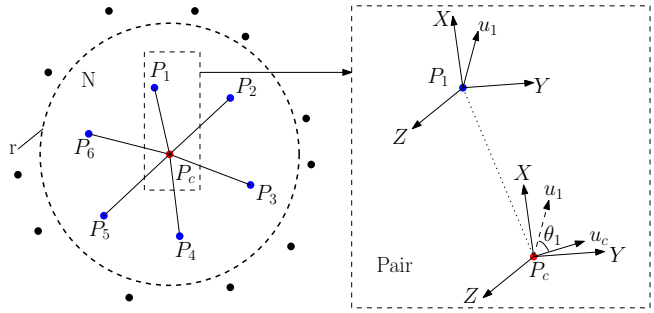

Fig. 7. (Left) The neighbor N of one point P is defined as a sphere whose radius is r. Within the neighbor N, the red point is the central point P_c of the sphere and the blue points are neighbor points. (Right) The central point P_c with its normal u_c is paired to one neighbor point P_1 with its normal u_1. θ_1 is the included angle between u_c and u_1.

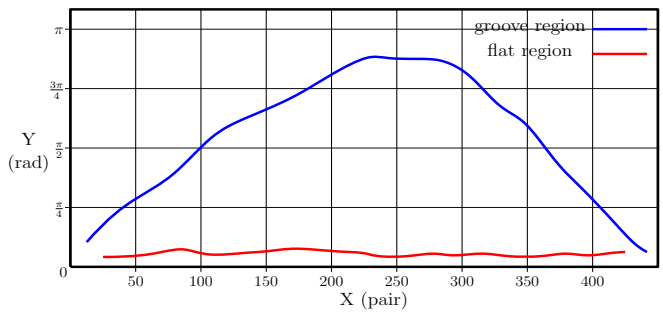

Fig. 8. Local groove feature histogram. The groove region has higher variation extent than the flat region. X-axis shows the number of pairs in one point's neighbor. Y-axis shows the included angle of one pair.

Generally, since the 3D position of every point is relevant to the manipulator base coordinate system through rigid transformation, all the point normals are based on the same coordinate XYZ (see Fig. 7 [Right]). Furthermore, the included angle θ_j of the j-th pair within the center point's neighbor is computed as:

$$\theta_j = \arccos \frac{\vec{u_c} \cdot \vec{u_j}}{|\vec{u_c}| \cdot |\vec{u_j}|} \qquad (2)$$

where $\vec{u_c}$ is normal of the center point and $\vec{u_j}$ is normal of the j-th neighbor point paired to the center point. By using Equation (2) for iterating every pair of the neighbor, all the computation results are arranged as a set ($\Theta = \{\theta_1, \theta_2, ..., \theta_n\}$, n is number of the neighbor points), which builds up the local GFH of the center point. Fig. 8 shows the local GFH of one point from the groove region and one point from the flat region.

C. Global GFH

For global GFH, the whole point cloud needs to be taken into account. Therefore, the unit benchmark normal $\vec{u_b}$ representing the main direction of the whole point cloud is defined as:

$$\vec{U_b} = \sum_{i=0}^{n} \vec{u_i}, \quad \vec{u_b} = \frac{\vec{U_b}}{\left|\vec{U_b}\right|} \tag{3}$$

where $\vec{u_i}$ is the i-th point normal of the point cloud.

Back to the neighbor of the previous point (the central point P_c in Fig. 7), each point normal in the neighbor is paired to the benchmark normal $\vec{u_b}$ (Fig. 9).

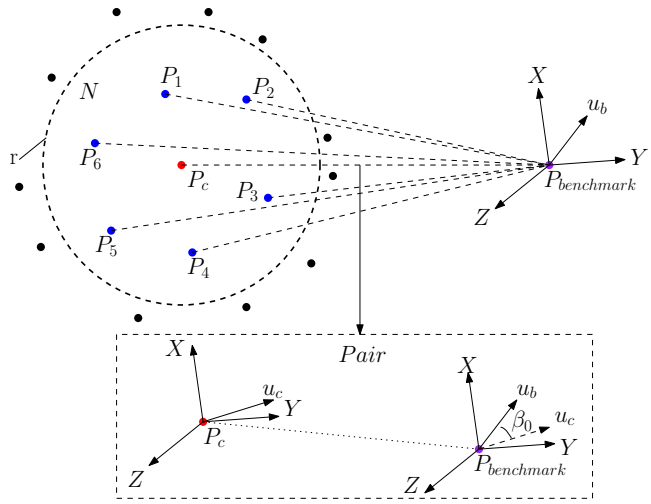

Fig. 9. Every point normal of the neighbor is paired to the benchmark normal. For example of one pair (the central point normal $\vec{u_c}$ and the benchmark normal $\vec{u_b}$), β_0 is the included angle between $\vec{u_c}$ and $\vec{u_b}$.

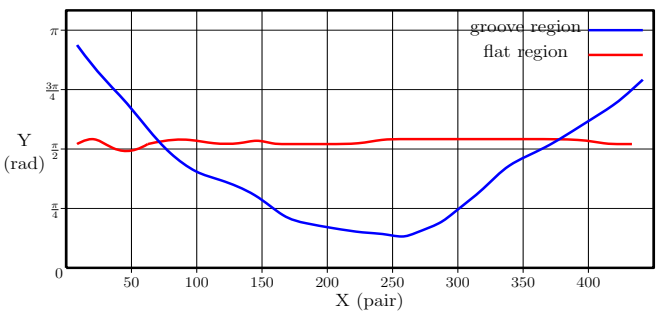

Fig. 10. Global groove feature histogram. The groove region has a higher variation extent than the flat region. X-axis shows the number of pairs in one point's neighbor. Y-axis shows the included angle of one pair.

Under the same principle as local GFH, the k-th included angle β_k of the neighbor is computed as:

$$\beta_k = \arccos \frac{\vec{u_b} \cdot \vec{u_k}}{|\vec{u_b}| \cdot |\vec{u_k}|} \tag{4}$$

Therefore all the resultant included angles are put into one set ($B = \{\beta_1, \beta_2, ..., \beta_n\}$) which builds up the global GFH of the central point. Fig. 10 shows the global GFH of one point from the groove region and one point from the flat region.

D. Surface Variation Descriptor

According to analysis of the local GFH III-B and global GFH III-C, the variations for both histograms of one point are defined respectively as:

$$\sigma^l = \frac{\sum_{j=0}^{n} (\theta_j - \overline{\theta})^2}{n}, \quad \sigma^g = \frac{\sum_{k=0}^{n} (\beta_k - \overline{\beta})^2}{n} \tag{5}$$

where $\overline{\theta}$ is average value of the local GFH and $\overline{\beta}$ is average value of the global GFH. n is the number of pairs in the point's neighbor. Then the surface variation descriptor D_i for i-th point P_i of the point cloud is defined as:

$$D_i = \sqrt{(\sigma_i^l)^2 + (\sigma_i^g)^2} \tag{6}$$

where σ_i^l is variation of local GFH of P_i and σ_i^g is variation of global GFH of P_i.

Through computing D_i of every point, a map which represents the surface variation extent of the whole point cloud is obtained and shown as Fig. 11. It intuitively shows the extent of surface variation on the whole point cloud, although there are noisy places. The blue regions, such as the groove region and the edges, have high variation, while the white regions are almost flat.

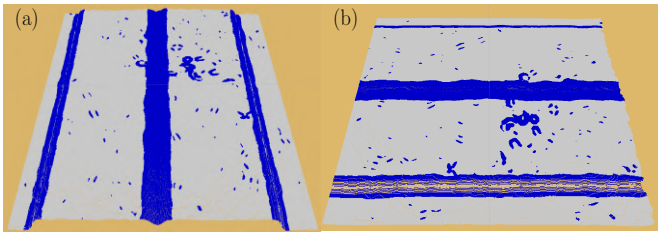

Fig. 11. Surface Variation Map. (a) Front view. (b) Side view.

By analyzing the surface variation map, points of the groove region with descriptor values more than threshold: 4.5-5 are centralized and almost tightly connected. In terms of the descriptor threshold, all the unqualified points are deleted. Consequently, only the groove point set remains. The groove detection results (blue region) are shown in Fig. 12. In the next step, the groove point set is used to generate the 3D welding trajectory for the robotic arm.

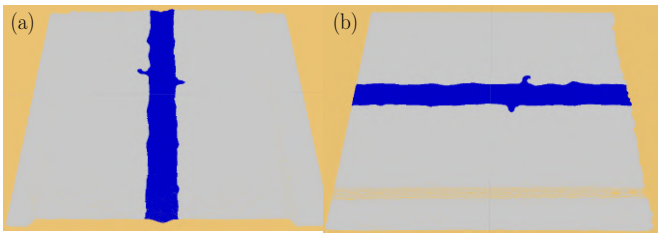

Fig. 12. Groove Detection Result. (a) Front view. (b) Side view

Compared with the PFH descriptor (computational complexity is $\mathcal{O}(\nu\mu^2)$) [22], the surface variation descriptor solely considers angular values instead of combining angular values and the distance of a single pair. One of the two

components of the surface variation descriptor, global GFH, is related to all the point normals, meaning that it can adapt to different types of workpieces. Since PFH focuses on local geometric features, it is not as adaptive as global GFH. Moreover, the other component, local GFH, is simplified SFPH [23] without involving the distance information. Therefore, the proposed surface variation descriptor reduces the computational complexity to $\mathcal{O}(2\nu\mu)$, where the ν is the number of points in the point cloud and μ is the number of pairs in each point's neighbor. In an actual experiment, the proposed algorithm could run in real-time.

IV. WELDING TRAJECTORY GENERATION

Based on the groove point set (marked in blue) obtained by the groove detection algorithm (see Fig. 12), the 3D welding trajectory is generated accordingly. The welding motion direction is closely dependent on the layout of the groove point set. Then, along the direction, the point set is evenly segmented into 50-60 consecutive regions of the same width (blue, green, and red for distinction), as shown in Fig. 13. The width of each segmented region is related to the total length of the welding groove; and each segmented region generates a single way point. Together, all the way points form the final 3D welding trajectory.

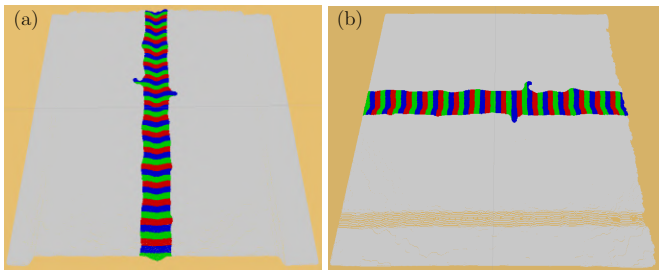

Fig. 13. Groove point set segmentation. (a) Front view of segmented groove region. (b) Side view of segmented groove region.

The sum of the distance from the way point of each segmented region to every point of the same segmented region should be as short as possible. The generation accuracy of the way points is dependent on groove detection results. If one resultant segmented region does not perfectly match the actual groove, its way point will deviate from the central position. The issue can be defined as a function for each segmented region:

$$\vec{I_i^s} = P^w - P_i^s, \quad f = argmin \sum_{i=0}^{m} \left| \vec{I_i^s} \right| \tag{7}$$

where P^w is the unknown way point and P_i^s is the i-th point of one segmented region. When f gets to its minimum, P^w is obtained as the way point. During experimentation, the gradient descent method is used to optimize the function. Specifically, the value 1e-4 is set as the threshold for the aborting iteration and the maximum of the iteration loop is set as 1000.

After iterating every segmented region of the groove point using Equation (7), the final welding trajectory generation result is shown as Fig. 14. The 3D orientation of each way

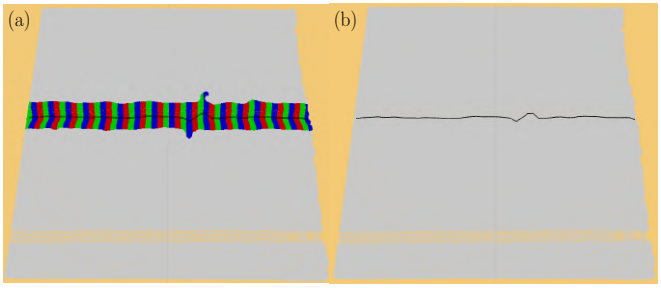

Fig. 14. Welding trajectory generation. (a) Generated welding trajectory inside the segmented groove point set. (b) The trajectory inside the raw point cloud.

point of the generated trajectory is represented as a unit vector $\vec{o_w}$, defined as:

$$\vec{O_w} = \sum_{i=0}^{m} \vec{u_i^s}, \quad \vec{o_w} = \frac{\vec{O_w}}{\left|\vec{O_w}\right|} \tag{8}$$

where $\vec{u_i^s}$ is the i-th point normal of the segmented region. Then the unit vector is transformed as three Euler angles by the Eigen library. Ultimately, a 3D welding trajectory with both 3D positions and orientations for the welding groove in workpiece is generated. The trajectory will then be sent to the manipulator control for executing the actual tracking motion.

V. RESULTS

An open welding environment (Fig. 1) with four types of welding workpieces was prepared for experiments. Fig. 15 shows the experimental welding workpieces which are straight-line, curve-line, box and cylinder respectively. The reason why the wood workpieces are regarded as experimental objects, is due to their convenience to simulate real welding groove like straight type or curve type. The material of welding workpieces has no effects on the proposed algorithm, because the inputting 3D point cloud contains no optical properties.

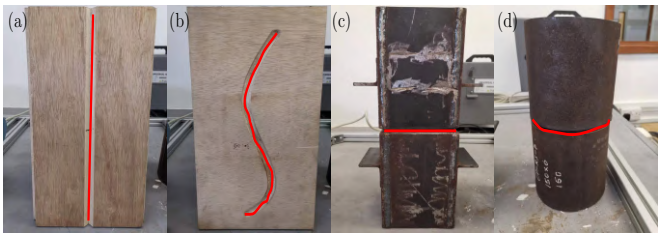

Fig. 15. Experimental welding workpieces with their welding groove marked as red regions. According to the shape of the welding groove, the workpiece is defined as: (a) straight-line; (b) curve-line; (c) box; (d) cylinder.

In order to objectively evaluate the performance of the proposed method, three vital elements are considered:

1) The whole processing runtime from the raw point cloud to generating the motion trajectory.

2) The overlapping rate of the detected groove region and the actual groove region.
3) The disparity between the generated trajectory and standard welding trajectory.

The processing runtime as an essential factor could evaluate the efficiency of the proposed robotic system for automatic welding. The overlapping rate is used to illustrate the accuracy of the welding groove detection algorithm. The disparity is presented by actual automatic welding execution. Thus, by focusing on the aforementioned three elements, each workpiece shown in Fig. 15 was tested. In fact, although the RGB-D camera is well calibrated, an error in the measurement of depth still exists.

A. Processing Runtime Results

To measure the processing runtime of the proposed method for the surface point cloud of one workpiece, the function "clock()" of C++ is used to capture the system start time and end time of the method. Due to the running performance of the PC, each workpiece was tested ten times and each runtime from inputting the raw point cloud to generating the motion trajectory was recorded (Table I).

TABLE I
RESULTS OF PROCESSING RUNTIME FOR EACH WORKPIECE

Types	P_{num}	t_{mean}(s)
Straight-line	265800	14.09
Curve-line	266497	14.08
Box	127031	7.51
Cylinder	121429	6.05

P_{num} = Number of points. t_{mean} = Average time of runtime.

B. Groove Detection Results

To evaluate the accuracy of the welding groove detection algorithm, the results of the algorithm need to be compared with the ground truth (actual groove region defined by the authors). Inspired by the concept of IoU (intersection over union) in 2D image processing, the 3D overlapping rate λ of the detected groove region and actual groove region is introduced:

$$\lambda = \frac{n_{overlap}}{n_d + n_{gt} - n_{overlap}} \tag{9}$$

where n_d is the number of points of the detected groove region, n_{gt} is the number of points of the actual groove region and $n_{overlap}$ is number of points of overlapping region between the detected groove region and the actual groove region.

The welding groove detection process for each workpiece in Fig. 15 is shown in Fig. 16, and the detection accuracy (defined by 3D overlapping rate) results are presented in Table II.

For types of straight-line, curve-line and box, the proposed welding groove detection approach has quite high detection accuracy. But for type of cylinder, the performance of the detection approach is not acceptable, because of poor robustness to model surface which is not almost plane.

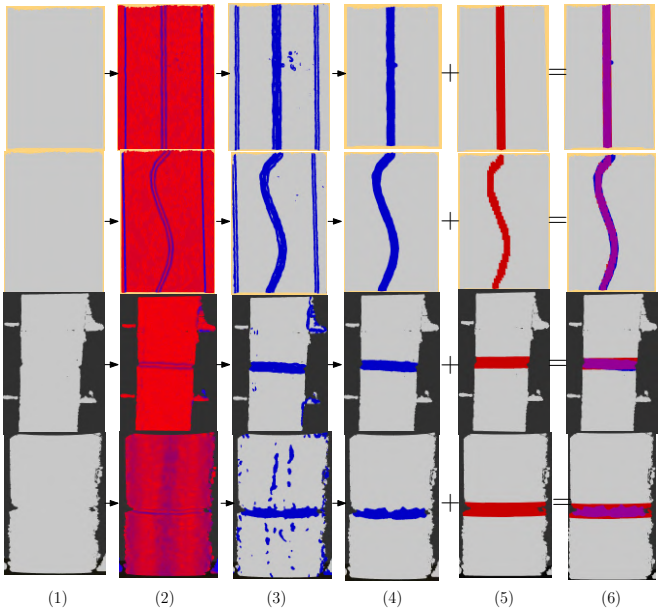

Fig. 16. Welding groove detection workflow. (1) Input raw point cloud of each workpiece. (2) Global normal map. (3) Surface variation map by groove feature histogram (GFH). (4) Filtering results (detected groove region). (5) Ground truth of groove region in raw point cloud. (6) Overlapping map by adding detected groove region and ground truth of groove region. Eventually the overlapping map is used to compute the 3D overlapping rate to evaluate the accuracy of the groove detection algorithm.

TABLE II
RESULTS OF GROOVE DETECTION ACCURACY FOR EACH WORKPIECE

Types	λ_1	λ_2	λ_3	λ_4	λ_5	λ_{mean}
Straight-line	92.74	93.01	92.81	92.49	91.35	92.48
Curve-line	81.24	82.58	82.13	81.87	82.29	82.02
Box	81.48	82.65	82.02	80.49	81.97	81.72
Cylinder	63.27	61.79	64.69	67.57	65.75	64.61

λ_1-λ_5 = Accuracy (%) of each test. λ_{mean} = Average accuracy.

C. Motion Execution Results

As discussed in Section IV, the motion trajectory generation is based on the point set of the detected groove region (see Fig. 16). Therefore, the robotic manipulator drives the welding torch which follows the motion trajectory to execute the welding tasks. Fig. 17 presents the actual motion execution of the robotic manipulator. As shown in Fig. 17, the motion of the manipulator fits the generated trajectory tightly without terrible mismatch. Also, by using cartesian path planning of the Moveit [20] package, the welding torch is able to complete welding tasks well (smoothly following the trajectory of the groove). However, it is difficult to evaluate the welding torch's motion accuracy through external devices such as Vicon [24]. In this case, evaluating the actual welding quality is the most reasonable course of action.

According to the running performance (runtime and detection accuracy) of the proposed method, the robotic system has the capability to realize automatic welding tasks efficiently, as compared to the conventional teach-playback method.

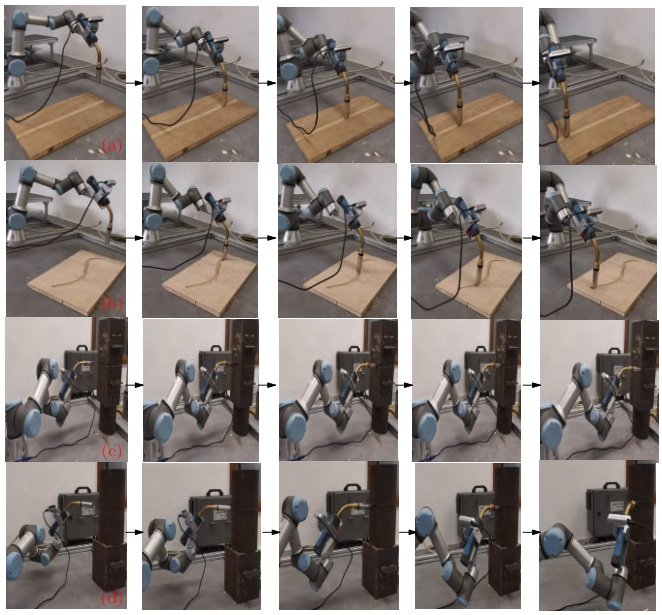

Fig. 17. Actual motion execution of the robotic manipulator. (a)-(b) represents type of straight-line, curve-line, metal box and metal cylinder workpiece respectively. And their motion order is from left side to right side.

VI. Conclusions

This paper presents an integrated robotic system for industrial welding with an automatic groove detection and trajectory generation method. The system is composed of a robotic manipulator (Universal Robot 3), an RGB-D camera (Realsense D415), and a welding torch. The system also has good flexibility when facing different welding situations. The software framework is built on ROS, with 3D point cloud processing being key. Four types of general welding workpieces were tested. After evaluating the accuracy between the welding trajectory generated by the proposed method and the ground truth of the welding groove, the motion execution performance proves good feasibility of the designed robotic system. However, a problem with the proposed method is that it cannot cope with a larger 3D point cloud of welding workpiece surfaces due to the more complex geometric regions and noise. In future work, a neural network instead of a geometric feature-based method could be introduced to improve robustness and accuracy of the welding groove detection algorithm. Also, we may incorporate additional functionalities to the robot, e.g. the capability to interact with the workpice [25].

References

[1] J. Norberto Pires, T. Godinho, and P. Ferreira, "Cad interface for automatic robot welding programming," *Industrial Robot: An Int. J.*, vol. 31, no. 1, pp. 71–76, 2004.

[2] Y. Li, Y. F. Li, Q. L. Wang, D. Xu, and M. Tan, "Measurement and defect detection of the weld bead based on online vision inspection," *IEEE Trans. Instrumentation and Measurement*, vol. 59, no. 7, pp. 1841–1849, 2009.

[3] Y. Liu and Y. Zhang, "Iterative local anfis-based human welder intelligence modeling and control in pipe gtaw process: A data-driven approach," *IEEE/ASME Trans. Mechatronics*, vol. 20, no. 3, pp. 1079–1088, 2014.

[4] C. Diao, J. Ding, S. Williams, Y. Zhao, *et al.*, "A passive imaging system for geometry measurement for the plasma arc welding process," *IEEE Trans. Industrial Electronics*, vol. 64, no. 9, pp. 7201–7209, 2017.

[5] J. Li, F. Jing, and E. Li, "A new teaching system for arc welding robots with auxiliary path point generation module," in *35th Chinese Control Conf. (CCC)*, 2016, pp. 6217–6221.

[6] M. Rodríguez-Martín, P. Rodríguez-Gonzálvez, D. Gonzalez-Aguilera, and J. Fernandez-Hernandez, "Feasibility study of a structured light system applied to welding inspection based on articulated coordinate measure machine data," *IEEE Sensors J.*, vol. 17, no. 13, pp. 4217–4224, 2017.

[7] S. M. Ahmed, Y. Z. Tan, G. H. Lee, C. M. Chew, and C. K. Pang, "Object detection and motion planning for automated welding of tubular joints," in *IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS)*, 2016, pp. 2610–2615.

[8] H. Ma, S. Wei, Z. Sheng, T. Lin, and S. Chen, "Robot welding seam tracking method based on passive vision for thin plate closed-gap butt welding," *The Int. J. Advanced Manufacturing Technology*, vol. 48, no. 9-12, pp. 945–953, 2010.

[9] S. H. Rao, V. Kalaichelvi, and R. Karthikeyan, "Real-time tracing of a weld line using artificial neural networks," in *IEEE/ACIS 17th Int. Conf. Computer and Information Science (ICIS)*, 2018, pp. 275–280.

[10] L. Nele, E. Sarno, and A. Keshari, "An image acquisition system for real-time seam tracking," *The Int. J. Advanced Manufacturing Technology*, vol. 69, no. 9-12, pp. 2099–2110, 2013.

[11] R. B. Rusu and S. Cousins, "3d is here: Point cloud library (pcl)," in *IEEE Int. Conf. Robotics and Automation*, 2011, pp. 1–4.

[12] Y. Xu, N. Lv, G. Fang, S. Du, W. Zhao, Z. Ye, and S. Chen, "Welding seam tracking in robotic gas metal arc welding," *J. Materials Processing Technology*, vol. 248, pp. 18–30, 2017.

[13] S. M. Ahmed, Y. Z. Tan, C. M. Chew, A. Al Mamun, and F. S. Wong, "Edge and corner detection for unorganized 3d point clouds with application to robotic welding," in *IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS)*, 2018, pp. 7350–7355.

[14] S. Song and J. Xiao, "Sliding shapes for 3d object detection in depth images," in *European Conf. computer vision*, 2014, pp. 634–651.

[15] B. Yang, W. Luo, and R. Urtasun, "Pixor: Real-time 3d object detection from point clouds," in *Proceedings of the IEEE Conf. Computer Vision and Pattern Recognition*, 2018, pp. 7652–7660.

[16] P. Maiolino, R. Woolley, D. Branson, P. Benardos, A. Popov, and S. Ratchev, "Flexible robot sealant dispensing cell using rgb-d sensor and off-line programming," *Robotics and Computer-Integrated Manufacturing*, vol. 48, pp. 188–195, 2017.

[17] L. Jing, J. Fengshui, and L. En, "Rgb-d sensor-based auto path generation method for arc welding robot," in *Chinese control and decision Conf. (CCDC)*, 2016, pp. 4390–4395.

[18] V. Patil, I. Patil, V. Kalaichelvi, and R. Karthikeyan, "Extraction of weld seam in 3d point clouds for real time welding using 5 dof robotic arm," in *5th Int. Conf. Control, Automation and Robotics (ICCAR)*, 2019, pp. 727–733.

[19] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "Ros: an open-source robot operating system," in *ICRA workshop on open source software*, vol. 3, no. 3.2, 2009, p. 5.

[20] S. Chitta, I. Sucan, and S. Cousins, "Moveit![ros topics]," *IEEE Robotics & Automation Magazine*, vol. 19, no. 1, pp. 18–19, 2012.

[21] R. B. Rusu, G. Bradski, R. Thibaux, and J. Hsu, "Fast 3d recognition and pose using the viewpoint feature histogram," in *IEEE/RSJ Int. Conf. Intelligent Robots and Systems*, 2010, pp. 2155–2162.

[22] L. A. Alexandre, "3d descriptors for object and category recognition: a comparative evaluation," in *Workshop on Color-Depth Camera Fusion in Robotics at the IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS), Vilamoura, Portugal*, vol. 1, no. 3, 2012, p. 7.

[23] R. B. Rusu, N. Blodow, and M. Beetz, "Fast point feature histograms (fpfh) for 3d registration," in *IEEE Int. Conf. Robotics and Automation*, 2009, pp. 3212–3217.

[24] O. Clarisse and S.-K. Chang, "Vicon," in *Visual languages*, 1986, pp. 151–190.

[25] D. Navarro-Alarcon, Y.-H. Liu, J. G. Romero, and P. Li, "Energy shaping methods for asymptotic force regulation of compliant mechanical systems," *IEEE Trans. Control Syst. Technol.*, vol. 22, no. 6, pp. 2376–2383, 2014.