International Research Journal of Advanced Engineering and Science
ISSN (Online): 2455-9024

# Research on Integrating GRBL Firmware to Control an Industrial Welding Robot Arm Model with Programmed Trajectory

Ngo Xuan Luan[1*], Nguyen Van Toan[1], Dang Xuan Hiep[1], Cao Duc Tan[2]
[1]Department of Machine Manufacturing, Faculty of Mechanical Engineering, Le Quy Don Technical University
[2]Department of Electronics and Telecommunications, University of Engineering and Technology, VNU
Email address: *ngoluan.mta@gmail.com

***Abstract*—*This article presents the results of GRBL firmware integration for controlling an industrial welding robot arm with programed trajectories. The mathematical model of the 3DOF robot arm is built to determine the working space and solve the kinematics problem. The inverse kinematics problem is analyzed analytically and its results are used as input for the control problem. GRBL firmware is described and analyzed with connection, signal transmission and robot arm control features. The robot motion planning problem is considered with two basic trajectories including a straight line and a circular arc. A robot arm concept model is built in reality with full mechanical and control components to test integration with GRBL firmware. The integration results show the ability of the robot arm to respond effectively to previously planned trajectories.*

***Keywords*—*Robot arm, firmware GRBL, control, trajectory planning, integration.***

## I. INTRODUCTION

In recent years, robots are playing an increasingly important role in production, especially in the manufacturing industry. Most of them are used in welding processing [1]. Therefore, the role of robots is becoming increasingly important and not only limited to large enterprises but also gradually expanding to small and medium enterprises in Industry 4.0.

A welding robot is a pre-programmed robot that helps people fully automate the mechanical welding process by performing welds and processing details. The control system plays an important role in controlling the welding robot, and it is considered the "brain" of the robot. There are many different types of welding robots such as arc welding robots, spot welding, laser welding, plasma welding... [2]. Welding robots have important advantages such as increased productivity, high repeat accuracy, high weld quality and reduced processing costs. In the shipbuilding industry [3-5], gantry welding robot systems are strongly applied in combination with image sensors used to position workpieces and monitor welds. An improved RRT algorithm [3], [6] is proposed to program the weld path to avoid collisions with obstacles, as well as reduce time [7], [8] in finding the optimal trajectory. during welding. A new method is proposed in [9] for programming and optimizing robot machining paths in machining large and complex parts with small machining costs such as aircraft [10], turbine blades [ 11].

GRBL [12] is a completely free and open source control system that allows using Arduino to control motion and operate small CNC machine models [13]. It is written in C programming language for high performance and uses all the smart features of the AVR microcontroller for precise timing response. GRBL can maintain stable control pulses up to 30kHz and uses an integrated G-code interpreter according to the RS274D standard. In addition, GRBL is capable of controlling machines that use stepper motors and is often used in equipment such as milling machines, lathes, 3D printers, plotters, engravers and SCARA robots [14]. After installation, GRBL can receive G-code commands from computer software such as Universal G-code Sender (UGS) through the USB port of the Arduino chip. GRBL supports all standard G-codes and works well with UGS. The system can be connected to a laptop via a graphical user interface (GUI). System commands and functions can be controlled through the GUI, helping users perform tasks such as starting programs, controlling in different modes.

As is known, the robot's movement task is determined by determining a path along which the robot must move. The robot needs to follow the laws of motion while moving with the actuators generating force or torque and it does not violate the workspace limits along with meeting other factors such as vibration reduction or not. Vibration excitation of mechanical structures. Trajectory programming algorithms are developed to create a moving trajectory suitable for the robot arm. Accordingly, programming the welding robot's trajectory plays an important role in the welding process, especially when there are many welds that need to be performed. A properly programmed weld trajectory can help shorten overall time, increase production efficiency and reduce costs. The development of intelligent algorithms brings many effective methods to solve the problem of programming the trajectory of welding robots [15], [16], [17].

Accordingly, this article presents the results of welding trajectory programming for a 3DOF robot arm with the use of GRBL firmware to control the system. Two basic welding trajectories are considered to evaluate the feasibility of GRBL firmware integration problem.

## II. RESEARCH CONTENT

### *A. Kinematics modeling of industrial robot arm models*



Ngo Xuan Luan, Nguyen Van Toan, Dang Xuan Hiep, and Cao Duc Tan, "Research on integrating GRBL firmware to control an industrial welding robot arm model with programmed trajectory" *International Research Journal of Advanced Engineering and Science*, Volume 9, Issue 2, pp. 140-145, 2024.

International Research Journal of Advanced Engineering and Science
ISSN (Online): 2455-9024

Consider the industrial robot arm model depicted in Fig. 1.

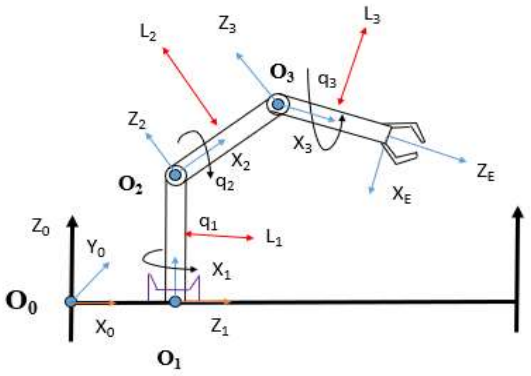


Fig. 1. 2D model of 3DOF robot arm

The robot consists of 3 joints corresponding to 3 degrees of freedom (3DOF). Joint 1 is rotating joint $q_1$, joint 2 is rotating joint $q_2$, joint 3 is rotating joint $q_3$. Point E is the end-effector point. The fixed coordinate system is $(Oxyz)_0$. Coordinate system $(Oxyz)_1$ is mounted at origin $O_1$. Coordinate system $(Oxyz)_2$ is mounted at origin $O_2$. Coordinate system $(Oxyz)_3$ is mounted at origin $O_3$. Coordinate system $(Oxyz)_E$ is mounted at the original position $O_E$. The structure of the robot arm is shown in Fig. 2.

Fig. 2. 3D model of 3DOF Robot arm

Based on D-H modeling theory [18], D-H parameters are set for the robot and are described in Tab. 1.

TABLE I. D-H PARAMETERS

| Link | $\theta$ | d | a | $\alpha$ |
|---|---|---|---|---|
| Link 1 | $\theta_1$ | $d_1 = L_1$ | 0 | $\frac{\pi}{2}$ |
| Link 2 | $\theta_2$ | 0 | $a_2 = L_2$ | 0 |
| Link 3 | $\theta_3$ | 0 | $a_3 = L_3$ | 0 |

Accordingly, the coordinates of the end-effector point location are determined as follows

$$\begin{aligned} xE &= \left(\left(L_3 * cos(\theta_3) + L_2\right) * cos(\theta_2) - L_3 * sin(\theta_2) * sin(\theta_3)\right) * cos(\theta_1); \\ yE &= sin(\theta_1) * \left(\left(L_3 * cos(\theta_3) + L_2\right) * cos(\theta_2) - L_3 * sin(\theta_2) * sin(\theta_3)\right); \\ zE &= \left(L_3 * cos(\theta_3) + L_2\right) * sin(\theta_2) + cos(\theta_2) * L_3 * sin(\theta_3) + L_1; \end{aligned} \quad (1)$$

Table 2 shows the geometric parameters and working limits of the joints.

The workspace of robot is calculated and shown in Fig. 3.

TABLE II. GEOMETRIC PARAMETERS AND JOINTS WORKING LIMIT

| Geometric parameters | Limit joint angles | |
|---|---|---|
| | Max | Min |
| $L_1 = 80mm;$<br>$L_2 = 128mm;$<br>$L_3 = 135mm;$ | $\theta_1\_max = 2\pi/3;$<br>$\theta_2\_max = 11\pi/18;$<br>$\theta_3\_max = -\pi/6;$ | $\theta_1\_min = -2\pi/3;$<br>$\theta_2\_min = 0;$<br>$\theta_3\_min = -\pi/2;$ |

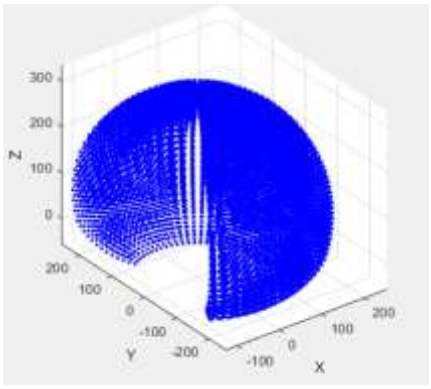


Fig. 3. Robot arm workspace

Based on the review of the results of building the kinematic equation and calculating the working space, the inverse kinematics problem is analyzed and solved by analytical methods. The calculation model is described in Fig. 4. Below are the steps to calculate the joint angles $\theta_1, \theta_2, \theta_3$ A of the 3DOF robot system in workspace with $x_E = P_x; y_E = P_y; z_E = P_z$ being the given coordinates and the value of the rotation angles must be converted from The $\arctan 2$ function ensures that the rotation angle values are accurate and unique.

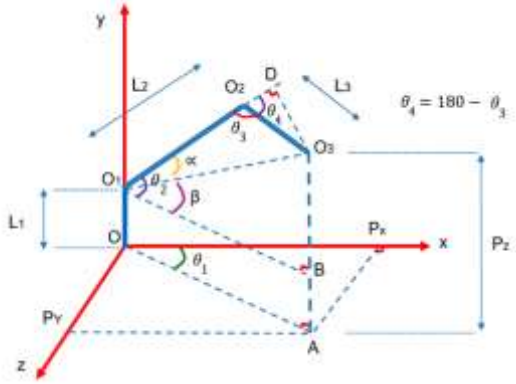


Fig. 4. Inverse kinematics calculation model

Joint $\theta_1$ is defined as follows

$$\theta_1 = arctan2\left(P_y, P_x\right) \quad (2)$$

Accordingly, joint $\theta_2$ is calculated as follows

$$\cos(\theta_3) = \frac{(L_2)^2 + (L_3)^2 - (P_X)^2 - (P_Y)^2 - (P_Z - L_1)^2}{2.L_2.L_3} \quad (3)$$

Through geometric transformation steps, joint $\theta_3$ is finally determined as follows



Ngo Xuan Luan, Nguyen Van Toan, Dang Xuan Hiep, and Cao Duc Tan, "Research on integrating GRBL firmware to control an industrial welding robot arm model with programmed trajectory" *International Research Journal of Advanced Engineering and Science*, Volume 9, Issue 2, pp. 140-145, 2024.

International Research Journal of Advanced Engineering and Science
ISSN (Online): 2455-9024

$$\theta_3 = arctan2\left(sin\left(\theta_3\right),\ cos\left(\theta_3\right)\right) \quad (4)$$

Joint $\theta_2$ is determined through the intermediate angle $\alpha$ and $\beta$ as follows

$$\alpha = arctan2\left(\frac{L_3.\mathrm{Sin}(\theta_3)}{L_2 - L_3.\mathrm{Cos}(\theta_3)}\right) \quad (5)$$

$$\beta = arctan2\left(\frac{P_Z - L_1}{\sqrt{(P_X)^2 + (P_Y)^2}}\right) \quad (6)$$

and,

$$\theta_2 = \alpha + \beta \quad (7)$$

### B. Firmware GRBL

GRBL is an open source CNC machine control software for Arduino boards. GRBL can be used in various industries such as manufacturing, manufacturing and construction. It is composed of main components including Arduino board, user interface, G-code protocol and expansion modules. GRBL firmware features include support for G-code protocols, support for expansion modules and it is open source for easy integration with different systems. The basic GRBL interface is shown in Fig. 5.

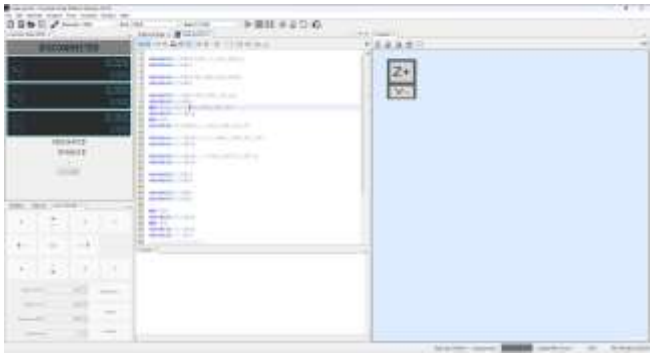

Fig. 5. User interface and some basic Gcode commands

### C. Trajectory planning for a robot arm

The trajectory planning problem is considered with two basic trajectories including straight lines and circular arcs.

The straight line trajectory is considered in space and is described in Fig. 6 with point A as the starting point and point B as the ending point.

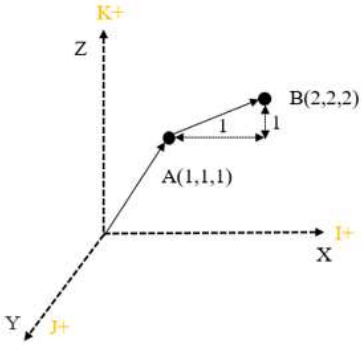


Fig. 6. Movement trajectory of E point in the workspace

The position vectors of points A and B are described as follows

$$\mathbf{X}_A = \begin{bmatrix} x_A \\ y_A \\ z_A \end{bmatrix}; \mathbf{X}_B = \begin{bmatrix} x_B \\ y_B \\ z_B \end{bmatrix} \quad (8)$$

Based on the inverse kinematics problem, the values of the joints corresponding to the positions of A and B are expressed as follows

$$\begin{aligned} \mathbf{\theta}_A &= \begin{bmatrix} \theta_{1A} & \theta_{2A} & \theta_{3A} \end{bmatrix}^T; \\ \mathbf{\theta}_B &= \begin{bmatrix} \theta_{1B} & \theta_{2B} & \theta_{3B} \end{bmatrix}^T; \end{aligned} \quad (9)$$

Because the inverse kinematics problem is solved by analytical methods, it is necessary to convention the direction of motion of the joints. Specifically described as follows

$$\begin{aligned} \theta_i &= \theta_{1A} - \theta_{1B} > 0 \rightarrow Forward\_motion \\ \theta_i &= \theta_{1A} - \theta_{1B} < 0 \rightarrow Inverse\_motion \\ \theta_i &= \theta_{1A} - \theta_{1B} = 0 \rightarrow Not\_motion; \end{aligned} \quad (10)$$

In case stepper motors are used to drive joints, the pulse signal that needs to be transmitted to the motors is calculated as follows

$$X_i = \frac{MicrosStep * TST * \theta_i}{StepSize} \quad (11)$$

In particular, the rotation angle of the joints to move from position A to B is $\theta_i (i = 1,2,3)$. The control mode of the motor is $MicrosStep$. The transmission ratio between the motor gear and the gear of the joint is $TST$. The step size used to Speed control is $StepSize$. Straight trajectory according to control code in GRBL is described in Tab. 3.

TABLE III. STRAIGHT TRAJECTORY CONTROL COMMAND USING G CODE.

| **Move to A position** | **Move to B position** |
|---|---|
| **G00** $X_A$ $Y_A$ $Z_A$ F100 | **G01** $X_B$ $Y_B$ $Z_B$ F100 |
| **G00:** The interpolation command moves straight from the initial position to point A | **G01:** command to interpolate linear motion from A to B |
| **F100:** Moving speed is 100 mm/min | |

The circular trajectory is considered with the starting point being A, the ending point being B, and the coordinates of the arc center being C. The relative distance from the center of C to A along the X axis is I, along the Y axis is J. The arc illustration can be considered as shown in Fig. 7.

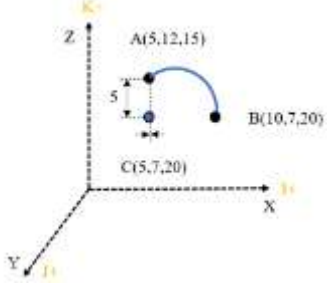


Fig. 7. Circular trajectory in the workspace



Ngo Xuan Luan, Nguyen Van Toan, Dang Xuan Hiep, and Cao Duc Tan, "Research on integrating GRBL firmware to control an industrial welding robot arm model with programmed trajectory" *International Research Journal of Advanced Engineering and Science*, Volume 9, Issue 2, pp. 140-145, 2024.

International Research Journal of Advanced Engineering and Science
ISSN (Online): 2455-9024

The coordinate vectors of points A and B and the center of arc C are expressed as follows

$$\mathbf{X}_A = \begin{bmatrix} x_A \\ y_A \\ z_A \end{bmatrix}; \mathbf{X}_B = \begin{bmatrix} x_B \\ y_B \\ z_B \end{bmatrix}; \mathbf{X}_C = \begin{bmatrix} x_C \\ y_C \\ z_C \end{bmatrix}. \quad (12)$$

The circular trajectory according to the control code in GRBL is described in Tab. 4.

TABLE IV. ARC CONTROL COMMANDS USING G CODE.

| **Move to A position** | **Move to B position** |
|---|---|
| G00 X5 Y12 Z20 F100 | G02 X5 Y-5 Z0 I0 J-5 F100 |
| **G02:** Circular interpolation clockwise | |
| **X5 Y12 Z20:** arc starting position; **X5 Y-5 Z0:** Arc ending position; **I0 J-5:** Relative distance from the center of the arc to point A; **F100:** Moving speed (mm/min) | |

*D. Integrated GRBL Firmware for robot arm*

*1. Connection steps*

To connect GRBL to the 3DOF robot model, the steps are as follows:

*Step 1:* Prepare the necessary components including 3DOF robot model, Arduino Uno circuit, Stepper motor transformer and jumper wires.

*Step 2:* Connect the components

Connect Arduino Uno to computer via USB port; Connect the stepper motor to the Stepper transformer; Connect Stepper transformer to Arduino Uno; Connect the jumper wires between the Arduino Uno and the 3DOF robot model.

*Step 3:* Install GRBL firmware for Arduino Uno.

*Step 4:* Open GRBL Console software on the computer.

*Step 5:* Configure GRBL parameters.

*Step 6:* Test connection.

*2. Set parameters of GRBL and transmit control signals*

After installing the GRBL firmware, the GRBL Console software is used to configure the GRBL parameters to suit the 3DOF robot model. Parameters to configure include: step speed (this is the speed at which the stepper motor will move); Acceleration and deceleration levels (these are the levels at which the stepper motor will speed up or decelerate) and linearity coefficient (this is the ratio between the motor steps and the distance moved by the end of the robot arm).

Regarding control signal transmission, the Universal Asynchronous Receive/Transmit communication standard or UART for short is used. This is an asynchronous data transmission and reception standard and is a popular communication standard, so it is often used in communication between microcontrollers with each other or with other devices. In GRBL, the UART communication standard is used to transmit G-code commands from the computer to the Arduino microcontroller. The Arduino microcontroller will process the G-code commands and send electrical pulses to the stepper motors to move them to the desired position. These electrical pulses are generated by STEP/DIR controllers.

The steps for transmitting control signals in GRBL are performed as follows

*Step 1:* The computer sends G-code commands to the Arduino Uno microcontroller via USB port.

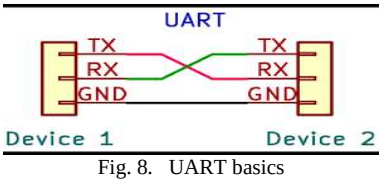


Fig. 8. UART basics

*Step 2:* Arduino Uno receives the G-code command and calculates the steps required to move the stepper motors.

*Step 3:* Arduino Uno sends electrical pulses to the stepper motors to move them to the desired position.

*Step 4:* The stepper motors move to the desired position and perform the movements specified by the G-code command.

Data transfer speed via USB port is an important factor to keep in mind when transmitting control signals in GRBL. The data transmission speed must be fast enough to transmit G-code commands without delay. The accuracy of the electrical pulses generated by the STEP/DIR controller is also an important factor to note. The electrical pulses must be highly precise to ensure that the stepper motors move to the desired position accurately.

*3. Control system components*

The Arduino Uno R3 microchip is used in this case. This is the block that has the function of receiving gcode control commands, processing and calculating inverse kinematics for the robot arm.

The Gcode command transmission and reception block is performed through the use of GcodeSender software. Universal Gcode Sender is software built in Java language, supporting users to control milling machines by sending G-code commands and instructions to CNC machines. Universal Gcode Sender software runs on Windows, OS X, Linux and Raspbian operating system platforms, etc.

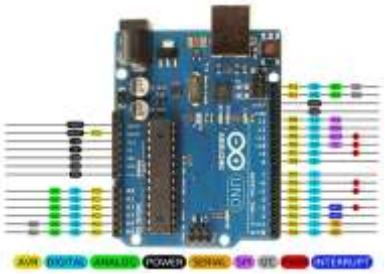

Fig. 9. Arduino Uno R3 pinout

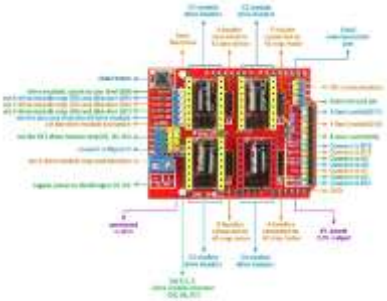

Fig. 10. CNC Shield V3 Board pinout

The motor's signal conversion block is performed using the Aruino CNC Shield V3 Board. This block has the function of



Ngo Xuan Luan, Nguyen Van Toan, Dang Xuan Hiep, and Cao Duc Tan, "Research on integrating GRBL firmware to control an industrial welding robot arm model with programmed trajectory" *International Research Journal of Advanced Engineering and Science*, Volume 9, Issue 2, pp. 140-145, 2024.

converting control signals into stepper motor control signals. Components of the block include CNC shield v3 and driver A4988 stepper motor.

Drive A4988 is an extremely small DMOS controller with converter and overcurrent protection. The A4988 can control bipolar stepper motors with current up to 2A per coil.

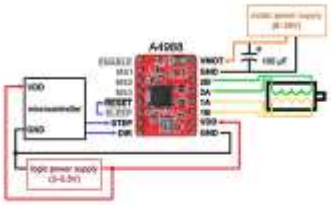

Fig. 11. Pin diagram of the A4988 driver controlling the stepper motor

After designing the principle diagram for each block and putting the parts together, the entire circuit diagram is described as follows.

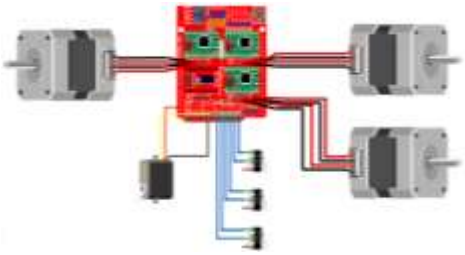

Fig. 12. Wiring diagram of the entire system

After assembling the 3D printed parts, stepper motors, servo motors, limit switches, the finished product is shown as shown in Fig. 13.

Fig. 13. Complete robot arm

Fig. 14 depicts the results of connecting the motor phase wire, power cord and endstop system to the CNC shield according to the principle diagram.

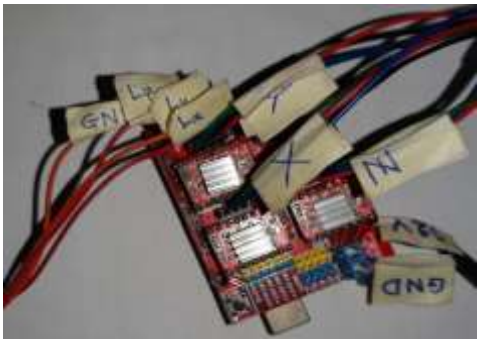

Fig. 14. Connecting the stepper motor and position sensor to the CNC Shield

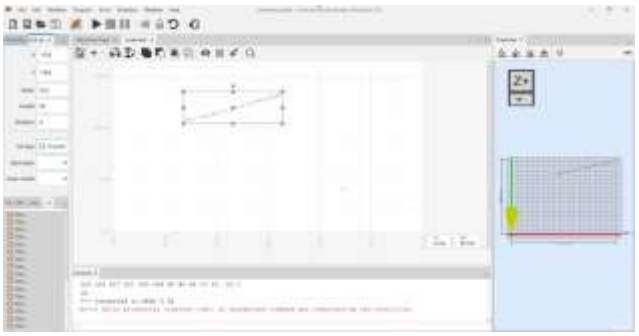

Fig. 15. Straight trajectory on GRBL

Fig. 16. Drawing a realistic straight trajectory

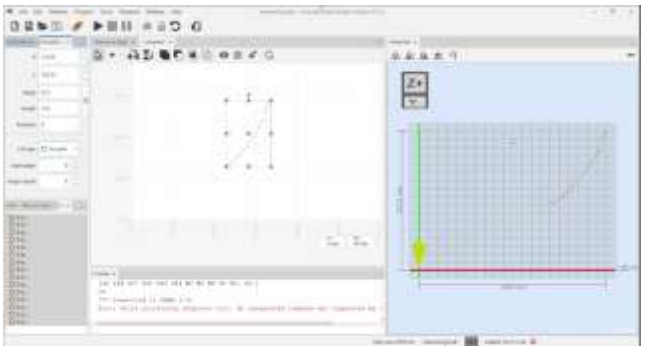

Fig. 17. Arc trajectory on GRBL

Fig. 18. Drawing the actual arc trajectory



Ngo Xuan Luan, Nguyen Van Toan, Dang Xuan Hiep, and Cao Duc Tan, “Research on integrating GRBL firmware to control an industrial welding robot arm model with programmed trajectory” *International Research Journal of Advanced Engineering and Science*, Volume 9, Issue 2, pp. 140-145, 2024.

## III. CONCLUSIONS

In summary, the article has presented the results of integrating GRBL firmware with a 3DOF robot arm to control the robot's movements according to programmed trajectories. Specifically, the robot kinematics problem is investigated and calculated accurately; The control system receives data and ensures the required movements; The robot arm receives control gcode commands and moves accurately to the desired position in 3-dimensional space; The endstop system with travel function works well, correctly determining the original working position of the arm; Furthermore, the gcode sender software effectively controls the robot arm model in modes such as: pressing button control (manual) or reading commands via gcode file (automatic).

However, some limitations that need to be considered include: poor load-bearing hardware, so it cannot integrate heavy applications for manipulation; The operating range of the robot arm is limited; Software that is completely synchronized with the system needs to be developed to avoid having to use open source alternative software.

Accordingly, the next development direction is determined to overcome the above limitations and expand applications for final operations such as adding 3D printing heads, laser engraving heads, CNC drilling heads, pick-and-drop clamps.

## REFERENCES

[1] https://www.robots.com/articles/welding-robots-improve-automotive-parts.

[2] J. Norberto. Pires, Altino. Loureiro, and Gunnar. Bolmsjö, Welding robots : technology, system issues and applications. Springer, 2006.

[3] X. Wang, J. Gao, X. Zhou, and X. Gu, "Path Planning for the Gantry Welding Robot System Based on Improved RRT*," Robot Comput Integr Manuf, vol. 85, Feb. 2024, doi: 10.1016/j.rcim.2023.102643.

[4] A. Zych, "Programming of Welding Robots in Shipbuilding," in Procedia CIRP, Elsevier B.V., 2021, pp. 478–483. doi: 10.1016/j.procir.2021.03.107.

[5] Y. Shen, Y. Gao, M. Yuan, H. Sun, and Z. Guo, "Multi-Objective Immune Optimization of Path Planning for Ship Welding Robot," Electronics (Switzerland), vol. 12, no. 9, May 2023, doi: 10.3390/electronics12092040.

[6] H. Zhang, Y. Wang, J. Zheng, and J. Yu, "Path planning of industrial robot based on improved RRT algorithm in complex environments," IEEE Access, vol. 6, pp. 53296–53306, 2018, doi: 10.1109/ACCESS.2018.2871222.

[7] X. Wang, B. Tang, Y. Yan, and X. Gu, "Time-Optimal Path Planning for Dual-Welding Robots Based on Intelligent Optimization Strategy," in Transactions on Intelligent Welding Manufacturing, Springer, 2018, pp. 47–59. doi: 10.1007/978-981-10-7043-3_3.

[8] A. Rout, M. Dileep, G. B. Mohanta, B. Deepak, and B. Biswal, "Optimal time-jerk trajectory planning of 6 axis welding robot using TLBO method," in Procedia Computer Science, Elsevier B.V., 2018, pp. 537–544. doi: 10.1016/j.procs.2018.07.067.

[9] G. Wang et al., "Trajectory Planning and Optimization for Robotic Machining Based on Measured Point Cloud," IEEE Transactions on Robotics, vol. 38, no. 3, pp. 1621–1637, Jun. 2022, doi: 10.1109/TRO.2021.3108506.

[10] Y. Liu, W. Yi, Z. Feng, J. Yao, and Y. Zhao, "Design and motion planning of a 7-DOF assembly robot with heavy load in spacecraft module," Robot Comput Integr Manuf, vol. 86, Apr. 2024, doi: 10.1016/j.rcim.2023.102645.

[11] W. Fang, L. Ding, X. Tian, and F. Zheng, "A robot welding path planning and automatic programming method for open impeller," International Journal of Advanced Manufacturing Technology, vol. 124, no. 5–6, pp. 1639–1650, Jan. 2023, doi: 10.1007/s00170-022-10415-9.

[12] B. Khan Altamash and K. Faizan Ahmed, "'CNC plasma cutting machine", Department of Mechanical Engineering In Partial Fulfillment of the Requirement for the Award of Bachelor's Degree In Mechanical Engineering."

[13] N. Parajuli, S. S. Thakuri, S. R. Shah, S. Bista, A. Shrestha, M. K. Guragai, Modeling and Fabrication of Low Cost 3-Axis Computer Numerical Control (CNC) Machine, International Journal of Advanced Engineering, Vol.04, No.01, pp.43-54

[14] Z. Miljković, N. Slavković, B. Momčilović, D. Milićević, Development of a Domestic 4-axis SCARA Robot, XI International Conference "Heavy Machinery-HM 2023", Vrnjačka Banja, 21–24 June 2023, P.9-16.

[15] F. Xu, Z. Hou, R. Xiao, Y. Xu, Q. Wang, and H. Zhang, "A novel welding path generation method for robotic multi-layer multi-pass welding based on weld seam feature point," Measurement (Lond), vol. 216, Jul. 2023, doi: 10.1016/j.measurement.2023.112910.

[16] C. Zheng et al., "Hybrid offline programming method for robotic welding systems," Robot Comput Integr Manuf, vol. 73, Feb. 2022, doi: 10.1016/j.rcim.2021.102238.

[17] Y. Geng, Y. Zhang, X. Tian, and L. Zhou, "A novel 3D vision-based robotic welding path extraction method for complex intersection curves," Robot Comput Integr Manuf, vol. 87, Jun. 2024, doi: 10.1016/j.rcim.2023.102702.

[18] M. W. Spong, S. Hutchinson, M. Vidyasagar, Robot modeling and Control, First edition, New York, USA, 2001.