ABB Robotic Arm Offline Programming System

Zhe Xu\textsuperscript{1,a} and Yuanjiang Liu\textsuperscript{1,b}

\textsuperscript{1}Beijing University of Technology, Beijing, China
\textsuperscript{a}xuzhe@bjut.edu.cn; \textsuperscript{b}411914912@qq.com

Abstract: The offline programming technology of industrial robots is more and more widely used because it can effectively improve work efficiency. A DXF drawing file is a file generated by CAD that can serve as a bridge for data exchange with other software. Through experiments, this paper combines CAD-generated DXF drawing files with ABB robotic arm to realize data extraction of CAD drawings, and automatic generation and storage of robotic arm running programs, which are imported into the robot arm, so that the robot arm can be made according to CAD. The trajectory on the drawing performs the welding function. The simulation of path planning was carried out under the Robot studio software developed by ABB Robotics to meet the experimental requirements. The realization of this off-line programming system achieves the following two purposes. One is to assist non-control professional teaching, to help understand the simulation and actual production and processing of industrial robot arm; the second is to improve the efficiency of industrial robot arm programming and realize full automation programming.

1. Introduction

With the development of economic globalization and the intensification of market competition, the demand for industrial robot technology and automation equipment in China’s manufacturing industry has grown rapidly. Among them, the application of industrial robots in welding is the most common\cite{1}. The laboratory has a multi-disciplinary robotic arm project that combines materials, electromechanical, environmental energy and control. The industrial robot (ABB robot arm) is used as a tool to weld or laser weld some metal materials.

With the increase of robot control speed and precision, the application range of robotic arc welding is becoming more and more extensive\cite{2}. At present, most of the domestic welding robots are working in the teaching and reproduction mode. The main technical problems in the actual production and application of the teaching and reproducing robots are \cite{3}: (1) The online teaching programming process of the robot is cumbersome and inefficient. (2) The accuracy of teaching is determined entirely by the experience of the teacher, and it is difficult to achieve a satisfactory teaching effect for complex paths. (3) For some applications that need to make real-time decisions based on external information, they cannot be made at any time. The teaching reproduction requires the production to stop the online programming of the welding robot. The contradiction between this production method and programming will be more and more, and the effective way to solve the contradiction is to use offline programming. Therefore, the research and application of arc welding robot offline programming technology \cite{4} is an inevitable trend to improve the level of welding automation.

2. DXF file
The system takes CAD drawings as input and processes it to obtain the executable file of the robot arm, which is then imported into the robot arm, so that the robot arm works according to the trajectory on the CAD drawing. In CAD, there are three formats for file saving: DWG, DWS, DWT, DXF. DXF (Drawing Exchange Format) is a CAD data file format developed by Auto-desk to exchange CAD data between Auto-CAD and other software [5]. DXF is an open vector data format that can be divided into two categories: ASCII format and binary format; ASCII has good readability, but it occupies a large space; binary format occupies less space and reads faster. Since Auto-CAD is now the most popular CAD system, DXF is also widely used as a de facto standard. Most CAD systems can read or output DXF files [6]. Therefore, research and analysis of DXF file structure has important significance.

2.1. Introduction of the DXF file
The complete DXF file consists of six segments and an end marker, each segment starting with a group code of 0 and a corresponding group value SECTION, followed by a group code and a group value representing the segment name, each piece of content being a group of elements The code and the group value are composed. The corresponding group code 0 of the group value ENDSEC indicates the end of the segment, and the entire DXF file end flag is marked with the group code 0 and the group value EOF[7]. The names, uses and contents of the six segments are shown in Figure 1.

![Figure 1. DXF file structure](image)

1) HEADER SECTION: with more than six authors: This section contains basic information about the graph and records the settings of the various variables associated with the graph. It consists of a database version number and a number of system variables, each of which includes a variable name and its group value.

2) CLASSES SECTION: This section contains information about the application-defined classes, and instances of these classes are contained in the database for segments, segments, and segments. Class definitions are fixed in the hierarchy of the class.

3) TABLES SECTION: This section contains definitions for the following symbol tables: application identifier, block reference table, dimension style sheet, layer table, line style table, text style sheet, user coordinate system table, view table, and viewport configuration table.

4) BLOCKS SECTION: This section contains the block definition and the graphic entity information that makes up each block reference in the drawing.

5) ENTITIES SECTION: This section contains entity (primitive) information that makes up the graph, including block references.

6) OBJECTS SECTION: This section contains non-graphic objects in the graph, and all those non-symbolic, symbolic, and non-primitive objects are stored in this area.

2.2. Processing of DXF files
Set the position of the target data according to the key field in the DXF file, ENTITY is the entity field, all the track data is stored in the entity, and the C language technology can be used to extract the data value of straight lines, polylines, arcs, splines, etc..
2.3. Extract the line coordinates in the DXF file

```cpp
fscanf(stream,"%d",&code);
fscanf(stream,"%s",codevalue);
if(code == 0 && strcmp(codevalue,"LINE")==0)
{ entity1 = (EntityData *)malloc(DATASIZE);
  strcpy(entity1->id,codevalue);
  fscanf(stream,"%d",&code);
  while(code)
  { //Save coordinate values }
}
```

2.4. Extract the polyline coordinates in the DXF file

```cpp
fscanf(stream,"%d",&code);
fscanf(stream,"%s",codevalue);
if(code == 0 && strcmp(codevalue,"POLYLINE")==0)
{ entity1 = (EntityData *)malloc(DATASIZE);
  strcpy(entity1->id,codevalue);
  fscanf(stream,"%d",&code);
  while(code)
  { //Save coordinate values }
}
```

2.5. Extract the arc coordinates in the DXF file

```cpp
fscanf(stream,"%d",&code);
fscanf(stream,"%s",codevalue);
if(code == 0 && strcmp(codevalue,"ARC")==0)
{ entity1 = (EntityData *)malloc(DATASIZE);
  strcpy(entity1->id,codevalue);
  fscanf(stream,"%d",&code);
  while(code)
  { //Save coordinate values }
}
2.6. Extract the B-spline curve in the DXF file

```c
fscanf(stream, "%d", &code);
fscanf(stream, "%s", codevalue);
if (code == 0 && strcmp(codevalue, "SPLINE") == 0)
{
    entity1 = (EntityData *)malloc(DATASIZE);
    strcpy(entity1->id, codevalue);
    fscanf(stream, "%d", &code);
    while (code)
    {
        //Save coordinate values
    }
}
```

3. Implementation of offline programming system

The robot offline programming system is based on the MFC framework in Visual Studio. MFC is the abbreviation of Microsoft Foundation Class Library. It is a C++ class library implemented by Microsoft Corporation. It mainly encapsulates most of the Windows API functions and contains an application framework to reduce the workload of application developers.

This offline programming system acts as an intermediate bridge connecting the work path to the ABB arm of the work tool, as shown in Figure 2.

![Figure 2. System structure](image)

Figure 3 shows the flow chart of the entire system.

![Figure 3. System flow chart](image)

3.1. Conversion algorithm of arc coordinate data

The arc information is stored in ENTITIES, and after the program scans to “ARC”, all useful information related to the arc is saved. In the DXF file, the position information of the arc is saved in the file in the format specified by the CAD software. The DXF file records the coordinates of the center point of the arc, the starting arc of the arc, and the arc of the end point of the arc. The parameters that need to be passed in the MOVC command that the robot arm can recognize are the coordinates of the three points on the arc. Value, these three points include the starting point coordinates of the arc, the midpoint coordinates of the arc, and the end point coordinates of the arc. So in this process, the geometry of the arc is involved. The geometry corresponding to the starting point, end point, center and radius of the arc is shown in Figure 4.
The arc is stored in the DXF file in the form of a center \((x, y, z)\), a starting angle \(\alpha\), and an ending angle \(\beta\). According to the geometric relationship of the arc, the following transformation can be performed: the arc is stored in the DXF file in the form of a center \((x, y, z)\), a starting angle \(\alpha\), and an ending angle \(\beta\). According to the geometric relationship of the arc, you can make the following conversion:

\[
\begin{align*}
x_1 &= x + R \cos \alpha, \quad y_1 = y + R \sin \alpha, \quad z_1 = z \\
x_2 &= x + R \sin \beta, \quad y_2 = y + R \sin \beta, \quad z_2 = z \\
x_3 &= x + R \cos((\alpha + \beta)/2), \quad y_3 = y + R \sin((\alpha + \beta)/2), \quad z_3 = z
\end{align*}
\]

This algorithm calculates the circular arc information extracted from the DXF file (i.e., the \(x\), \(y\), and \(z\) coordinates of the center point of the arc and the starting arc of the arc, and the ending arc) according to the corresponding relationship in the geometry. The three-point coordinate value[8] on the arc (that is, the \(x\), \(y\), and \(z\) coordinate values of the starting point, midpoint, and end point of the arc).

3.2. Conversion and Discrete Algorithm of B-Spline Curve Coordinate Data

The B-spline method has the powerful function of expressing and designing free-form curves and surfaces, and is one of the mainstream methods of shape mathematical description. In addition, the B-spline method is the current international standard for geometric definition of industrial products—the rational B-spline method (NURBS). The B-spline method combines all the advantages of the Bezier method, including geometric invariance, affine invariance, etc., while overcoming the shortcomings of the Bezier method due to the fact that the overall representation does not have local properties[9] (moving a control vertex will affect the entire curve).

In the DXF file, the information of the B-spline curve is stored in the physical segment SPLINE. To get a B-spline curve, the curve must be discretized. According to the \(x\), \(y\), and \(z\) coordinates of the control points in the DXF file, the new \(x\), \(y\), and \(z\) coordinate values of the discrete points can be calculated. Taking four control points \(P_1, P_2, P_3,\) and \(P_4\) and setting an equal-point \(u\), the new discrete point coordinate values can be calculated according to the following formula.

\[
\begin{align*}
n_1 &= (-u^3 + 3u^2 - 3u + 1)/6; \\
n_2 &= (3u^3 - 6u^2 + 4)/6; \\
n_3 &= (-3u^3 + 3u^2 + 3u + 1)/6; \\
n_4 &= (u^3)/6;
\end{align*}
\]

\[
\begin{align*}
cx &= n_1 p_1(x) + n_2 p_2(x) + n_3 p_3(x) + n_4 p_4(x); \\
cy &= n_1 p_1(y) + n_2 p_2(y) + n_3 p_3(y) + n_4 p_4(y); \\
cz &= n_1 p_1(z) + n_2 p_2(yz) + n_3 p_3(z) + n_4 p_4(z);
\end{align*}
\]
3.3. System implementation
The Figure 5 shows a functional interface implemented for this system.

![System function interface](image)

**Figure 5.** System function interface

1) Click on the “Open” control and select the file path to open via the dialog.
2) Click on the “Read” control to display the text information in the DXF file in the dialog box.
3) Click the “Extract Coordinates” control to extract and display the coordinates of the trajectory data in the DXF file in the dialog box.
4) q1-q4 is the attitude information, f1-f4 is the axis configuration, the workpiece coordinate system is used to determine the coordinate system, and the powder feeding waiting time to the powder input part is used to determine the value.
5) Click on the “Generate Executable” control to generate a command program that automatically moves into the arm and welds.
6) Enter the file save path and click the “Save As” control to save the generated program to the specified path.

4. Software simulation and experiment
The Robot Studio software is a set of software developed by ABB Robotics specifically for simulation with the ABB robotic arm. It is unique in that it has no translation phase in the process of downloading to the actual controller.

Using DXF files as a bridge for data exchange, the target data in the DXF file can be extracted and saved to an array for use by reading and writing files in the C language and the DXF file format. Finally, the data is written into the program in a certain format and saved. An executable file of the robot arm can be generated by the offline programming system, and the generated executable file is imported into the robot studio for simulation.

Figure 6 is a planar path drawn in CAD, saved in DXF format. Figure 7 is a path simulation performed in Robot Studio. The DXF file generated in Figure 1 can be used to generate an executable program of the robot arm through the robot offline programming system. Then the generated program is simulated in the virtual environment, and the same path as the DXF file can be obtained. Figure 8 is a verification of the motion control program generated by the system with the actual robotic arm.

![Figure 6. Plane path in CAD](image)

![Figure 7. Virtual simulation in Robot Studio](image)

![Figure 8. Experimental verification](image)
5. Conclusion
The simulation and experimental verification of Robot Studio software, the feasibility of this off-line programming system is proved. The established trajectory drawn by the CAD drawing can be simulated and applied to the actual robot arm welding. Input DXF drawing files, through the ABB robotic arm offline programming system, you can directly generate the execution program of the robot arm operation, and then import the executable program file into the laboratory ABB IRB 1600 robot arm, the arm can be welded according to the expected path. The actual experiment proved to be feasible.

In the realization process of the robot arm offline programming system, the key point is the extraction of the trajectory coordinate data in the DXF file. The partial trajectory coordinate data needs to be converted and discrete according to the geometric relationship, such as the trajectory of the circular trajectory and the B-spline curve. Coordinate data requires coordinate transformation and dispersion.

Acknowledgment
This work was supported by the “National Key R&D Program of the Ministry of Science and Technology (2018YFC1900801)”.

References
[1] Ilana Nisky, Ferdinando A, Amir Karniel. Analytical study of perceptual and motor transparency in bilateral teleoperation[J]. IEEE Transactions on Human-Machine Systems. 2013, 43(6): 570–582.
[2] Song Yixu, Yang Hongjun, Lv Hongbo. Intelligent control for a robot belt grinding system[J]. IEEE Transactions on Control Systems Technology. 2013, 21(3): 716–724.
[3] H.V. Brussel. About CIM, Concurrent Engineering and Mechatronics[J]. Advanced Manufacturing Technology. 1994(9): 23–30.
[4] Matin Kortus. An Alternative Approach to off-line Programming. Industrial Robot, 1993, 20(4): 17–20.
[5] Bi jan Shirinzadeh. Task Planning and off-line Programming. Industrial Robot, 1996, 23(5): 4–5.
[6] Xie Yuanyuan, Wang Chongchang. Realization of Digital Topographic Map Coordinate System Transformation in the CAD[J]. Geomatics & Spatial Information Technology. 2018, 41(6): 192-197.
[7] Lin Junjian, Zhang Xiaojin, Lin Yueke. Design of Offline-programming and Simulation System for Industrial Robot Based on XNA[J]. Machine & Hydraulics, 2017, 5(3): 24-27.
[8] Wang Xiaoyu. Coordinate conversion of DXF graphic files[J]. Geospatial Information, 2018, 16(6): 108-111.
[9] Wang Guitang, Xiong Feng, Chen Yongbin. Automatic path generation in 2-D detection based on DXF file[C]. 2017 IEEE 13th International Conference on Electronic Measurement & Instruments, 2017.