Six-axis Robot Manipulator Numerical Control Programming and Motion Simulation

Chen-hua SHE* and Che-hsuan CHANG

Department of Mechanical and Computer Aided Engineering, National Formosa University
64 Wen-Hua Road, Huwei, Yunlin 632, TAIWAN

*Corresponding author

Keywords: Robot manipulator, D-H notation, Forward kinematics, Inverse kinematics.

Abstract. This study, using the KUKA KR 150-2 six-axis robot manipulator configuration, adopted the Denavit-Hartenberg notation to build relevant kinematic mathematical model so that subsequent forward and inverse kinematics could be derived. Borland C++ Builder was employed to program the simulation interface, transforming the tool path files generated from hyperMILL CAD/CAM software into readable formats by robot manipulators. Moreover, OpenGL was used to complete the kinematic simulation of robot manipulators. Finally, this study adopted the solid cutting simulation software VERICUT® for further verification and comparison. Result indicates the effectiveness of the proposed scheme.

Introduction

Robot is the science and technology in a variety of equipment consisting of mechanical, electronic, control, computers, sensors and artificial intelligence, and is one of the elements for the automation of production. Global industrial robotic market demands mainly come from the manufacturing sector, which articulated robot is the most widely used models and has very extensive application areas, such as welding, disassembly, transportation, machinery processing. Metal machinery industry, electronic industry and chemical industry are the manufacturing industry with large-scale output value. Metal product manufacturing industry belongs to 3K industry and the production process has the features to produce the same product repeatedly. This makes it suitable to use robotic manipulator for production. If the manufacturer receives orders smoothly and faces the problem of lack of work experience, the demand for robotic manipulator will arise.

A 6-axis robot manipulator, with its motion function capable of mimicking the human arms, has received global attention in recent years. As traditional motion paths of robot manipulators are mostly instruction-based, when a workpiece has complex and uneven curved surface, such moving path could not meet the demands. To control the end effector of the robot accurately, much research [1-3] has addressed the solutions of the inverse kinematics issue of robot manipulators.

Many manufacturers use CAD/CAM software and with CNC machine tools for product design and manufacturing of part, and their related technical ability has matured. The required CNC program should be converted by the expensive commercial software while importing the robotic manipulator. Although commercial software such as SprutCAM [4], Robotmaster® [5] could be used to arrange the moving and processing path of a robot manipulator and to generate the corresponding numerical control program, such software is costly. It will not only save considerable cost of inputs, but also integrate with existing CAD/CAM software if they can apply the existing CAD/CAM modules and theoretical basis for the field of robotics to carry out the path programming. Therefore, how to efficiently and accurately develop a path transformation interface for robot manipulators and how to use CAD/CAM software to generate corresponding paths are vital issues to be investigated.

This study develops a six-axis robot arm NC programming algorithms which integrate the traditional CAD/CAM module. Based on the Denavit-Hartenberg (D-H) notation matrix [6], the six robot joint angles are obtained from the forward/inverse kinematics. A window-based transformation
and simulation interface written by Borland C++ Builder and OpenGL is developed which can convert the tool path file generated from CAD/CAM system into robot NC program and simulate the robot motion.

**Robot Forward Kinematics**

In order to specify the pose of the robot end effector, the coordinate system should be established. Figure 1 shows the configuration and coordinate systems of the KUKA KR 150-2 robot [7]. From a kinematics perspective, a robot can be regarded as a system of $n$ links. To fully specify the pose of each link, the D-H notation is introduced. A set of D-H parameters is used to describe the spatial relationships between a joint axis and its two neighbor joint axes. The pose link $i$ with respect to link $i-1$ can be defined in terms of the link length $a_i$, joint distance $d_i$, link twist angle $\alpha_i$ and joint angle $\theta_i$ as

$$i^{-1}A_i = \text{Rot}(z, \theta_i)\text{Trans}(0, 0, d_i)\text{Trans}(a_i, 0, 0)\text{Rot}(x, \alpha_i)$$

(1)

where $\text{Rot}$ and $\text{Trans}$ are the rotation and translation matrix, respectively. The corresponding kinematics parameters of the robot are summarized in Table 1. Since the actual joint angles of KR 150-2 shown in Figure 1 are $(0, -90, 90, 0, 0, 0)$, the joint angles of D-H parameters should be modified as $(\theta_1, \theta_2, \theta_3 + 90, \theta_4, \theta_5, \theta_6)$.

The pose matrix of the link 6 frame with respect to the base frame, which is known as the robot forward kinematics and denoted by $^0A_6$, can be determined by

$$^0A_6 = A_1^0A_2^1A_3^2A_4^3A_5^4A_6^5$$

(2)

where

$$i_x = -s_3(c_4s_6 + s_4c_5c_6) + c_3[s_23(s_4s_6 - c_4c_5c_6) - c_23s_5c_6].$$

$$i_y = s_3[s_23(s_4s_6 - c_4c_5c_6) - c_23s_5c_6] + c_3(c_4s_6 + s_4c_5c_6).$$

$$i_z = s_23s_5c_6 + c_23(s_4s_6 - c_4c_5c_6).$$
\begin{align*}
j_x &= -s_i(c_x c_6 - s_x c_5 s_6) + c_i[s_{23}(s_4 c_6 + c_4 c_5 s_6) + c_{23} s_5 s_6]. \\
j_y &= s_i[s_{23}(s_4 c_6 + c_4 c_5 s_6) + c_{23} s_5 s_6] + c_i(c_x c_6 - s_x c_5 s_6).
\end{align*}

\begin{align*}
j_z &= -s_{23}s_5 s_6 + c_{23}(s_4 c_6 - c_4 c_5 s_6). \\
k_x &= -s_i s_4 s_5 - c_i(s_{23} c_4 s_5 - c_{23} c_5). \\
k_y &= -s_i(s_{23} c_4 s_5 - c_{23} c_5) + c_i s_4 s_5. \\
k_z &= -s_{23} c_5 - c_{23} c_4 s_5.
\end{align*}

\begin{align*}
l_x &= c_i(a_i + a_2 c_2 - a_3 s_{23} + d_4 c_{23}) + d_6[c_{23} c_5 - s_5 (c_1 c_4 s_{23} + s_4 s_2)]. \\
l_y &= s_i(a_i + a_2 c_2 - a_3 s_{23} + d_4 c_{23}) + d_6[c_{23} c_5 + s_5 (-c_4 s_{23} + c_1 s_2)]. \\
l_z &= -a_2 s_2 + d_1 - s_{23}(d_4 + c_5 d_6) - c_{23}(a_3 + c_4 s_5 d_6).
\end{align*}

\begin{align*}
s_i &= \sin \theta_i, \quad c_i = \cos \theta_i, \quad s_{ij} = \sin(\theta_i + \theta_j), \quad c_{ij} = \cos(\theta_i + \theta_j). 
\end{align*}

Table 1. Link parameters of the D-H notation for KUKA KR 150-2 robot.

<table>
<thead>
<tr>
<th>Link</th>
<th>(a_i (mm))</th>
<th>(d_i (mm))</th>
<th>(\alpha_i (°))</th>
<th>(\theta_i (°))</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>350</td>
<td>750</td>
<td>-90</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>1250</td>
<td>0</td>
<td>0</td>
<td>-90</td>
</tr>
<tr>
<td>3</td>
<td>55</td>
<td>0</td>
<td>90</td>
<td>180</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>1100</td>
<td>-90</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>0</td>
<td>90</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>230</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

**Determination of Cutter Location Matrix and Joint Angles**

Most current CAD/CAM systems can produce the cutter location data including cutter tip location \(p\) and tool axis \(a\) as shown in Figure 2. To represent the robot pose using the cutter location data, two mutually orthogonal unit vectors \(o\) and \(n\) should be determined. The following equations are obtained by the definition of orthogonal unit vector.

\begin{align*}
a_x a_x + a_y a_y + a_z a_z &= 0. \\
n_x^2 + n_y^2 + n_z^2 &= 1.
\end{align*}

Since there are only two equations to determine three unknowns \((n_x, n_y, n_z)\), infinite possible solutions can be obtained. For convenience, we can assume that \(n_x = a_x\). Therefore, \(n_y\) and \(n_z\) can be determined as follows:

\begin{align*}
n_y &= \frac{-a_x a_x - a_z a_z}{a_y}. 
\end{align*}
Once two vectors \( \mathbf{a} \) and \( \mathbf{n} \) are obtained, the vector \( \mathbf{o} \) can be derived by the right hand rule and the cutter location matrix represented by \( [\mathbf{n} \ \mathbf{o} \ \mathbf{a} \ \mathbf{p}] \) is determined. Therefore, the six joint angles \( (\theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6) \) of the robot can be obtained by equating the Eq. (2) and cutter location matrix, and using the inverse kinematics.

\[
\mathbf{n}_c = -a_y a_1 \pm \sqrt{a_y^2 + a_x^2 - a_y^2 a_1^2 - a_y^2 a_2^2 - a_y^2 a_3^2},
\]

\[
\frac{a_1^2 + a_2^2 + a_3^2 + a_4^2 + a_5^2 + a_6^2}{a_1^2 + a_2^2 + a_3^2 + a_4^2 + a_5^2 + a_6^2}.
\]

(6)

Virtual Robot Simulation Construction

A virtual robot simulation interface was constructed by using Borland C++ Builder and OpenGL programming languages in a Windows environment. The CAD model of the robot is created in the CAD system and, then, saved as the stereolithography (STL) format. A STL parser is developed to analyze the CAD model in STL format. The virtual robot can be built sequentially according to the relationship of structural elements of a robot configuration. Additionally, the robot motion is displayed using the OpenGL library including Graphics Library (GL), OpenGL Utility (GLU), OpenGL Auxiliary (GLAUX) and OpenGL Utility Toolkit (GLUT).

System Implementation and Verification

This work develops a window-based robot simulation interface to demonstrate the feasibility of the proposed scheme. A cutter location data is generated by the hyperMILL software and converted to the robot NC program by the developed interface. Figure 3 shows a screenshot of the system execution dialogue where the robot NC program is executed. The user can actuate the robot by the NC program or manual data input (MDI) function where the joint angles can be manually entered. The MDI function provides a quick and easy means of verifying that the machine component combination responds to NC code commands as expected. The generated NC program is further verified on the solid cutting simulation software VERICUT\textsuperscript{®} [8], which can construct the kinematics model of a robot and simulate the NC data. Figure 4 shows that the KUKA KR 150-2 robot is built in the software environment and the NC program is simulated. The results shown in Figure 3 and Figure 4 demonstrate the effectiveness of the developed robot programming algorithm and motion simulation interface.
Conclusions

This paper presents a robot numerical control programming algorithm and motion simulation interface for the KUKA KR 150-2 robot configuration. The six joint angles of the robot can be determined by the forward and inverse kinematics based on the D-H notation. A window-based robot simulation interface built by using Borland C++ Builder and OpenGL can simulate the robot motion by NC program or MDI function. Through verification by the solid cutting simulation software VERICUT®, the validity of the generated robot NC program can be confirmed.

![Figure 3. Execution dialogue of the developed interface.](image1)

![Figure 4. Solid cutting simulation software VERICUT® verification.](image2)

Acknowledgement

The financial support of the Ministry of Science and Technology of Taiwan under grant MOST 105-2221-E-150-028 is gratefully acknowledged.

References


