A Method for Path Generation of Robot Automatic Polishing Based on Bounding Box

Lili Han\textsuperscript{1,2,3,a}, Qingwei Zhang\textsuperscript{1,2,3,b}, and Kai Jia\textsuperscript{1,3,c}

\textsuperscript{1}Shenyang Institute of Automation, Chinese Academy of Sciences, Shenyang, 110016, China
\textsuperscript{2}Graduate School of the Chinese Academy of Sciences, Beijing 100049, China
\textsuperscript{3}SIASUN Robot & Automation Co., Ltd, Shenyang 110168, China
\textsuperscript{a}savoy0924@163.com, \textsuperscript{b}Zhangqingwei@yahoo.cn, \textsuperscript{c}jiakai@siasun.com

Keywords: Path generation; Free-form surface; Bounding box; Direction-parallel machining

Abstract. In order to solve the problem of high intensity and low efficiency by manual labor during the polishing machining of mold free-form surface, this paper presents a method for path generation of robot automatic polishing based on bounding box. Firstly, ascertain the location and direction of the mold by constructing a box that is most close to the mold. Secondly, generate the tool path by introducing the improved direction-parallel machining. Finally, by using Visual C++6.0 development tool, the algorithm discussed above has been implemented and a software for trajectory generation has been developed. The results show that the method can generate the tool path better along the surface contour and is simple, effective. Besides, the software supplies a visible trajectory for robot polishing so that the unreasonable points can be modified before machining.

Introduction

During the process of mold manufacturing, the surface roughness of mold affects the product quality directly, and thus mold polishing is an absolute essential. Yet the traditional polishing mainly relies on manual labor which is a high intensity and low efficiency work. In recent years, with the widespread application of numerical control and CAD/CAM technology in mold industry, robot polishing method develops rapidly for its flexibility. For robotic machining system, in addition to the kinematics, dynamics and the transformation of the end gesture, we also need to consider trajectory planning for the execution tool.

At present, there isn’t a set of mature theory and method about the path generation for robot machining, although the domestic and foreign scholars have conducted extensive research. The algorithm of uniform scanning path proposed by Hon-yue Tam\textsuperscript{[1]} is just an improvement of the equal parameter lines method which is only suitable for these surfaces with uniform parameter lines. The direction-parallel machining and contour-parallel machining mentioned in reference \textsuperscript{[2]} are the most common method, while the limitations are obvious for ignoring the flexibility of the robot end-effector.

On the basis of direction-parallel machining method and the contour change of the free-form surface, summarizing the advantages and disadvantages of the previous method, this paper presents a method for path generation based on bounding box and develops a software for robot path generation.

Description of the Problem

Fig.1 shows a series of instantaneous scene during the process of robot machining workpiece. The advantage of multi-DOF guarantees that it can complete the whole machining from any position by any gesture. Then how to generate the trajectory for robot? In the previous robot polishing application as shown in Fig.2, the characteristics of regular shaped mold, simple path and the consistent terminal attitude make the robot lose its advantages. Therefore, what will be researched in this paper is just how to generate a series of track points for robot on the basis of the mold contour, especially for the free-form surface.
Path Generation for Robot Polishing

The Improved Method of Bounding Box. The basic idea of bounding box is to tightly wrap the object using simple shape geometry. It is widely used in collision detection, mold design, product packaging, image processing, pattern recognition and other fields. The traditional construction of box is much simple: First get separately the maximum and minimum of the object in $X$, $Y$ and $Z$ axis direction. Then draw 6 planes which respectively go through these values and are perpendicular to the corresponding axis. Last the cuboid consisted of intersecting planes is the bounding box. This method has the advantage of simple calculation but the disadvantage of not as close as possible to the object. Thus we improve it and present a new way as follows:

Step 1: Calculate the average normal vector of the surface.

For the surface $S$ shown in Fig.3 a), calculate the average normal vector with $M$ triangles. The calculation formula is defined as:

$$\bar{N} = \frac{\sum_{i=1}^{M} S_i \bar{n}_i}{\left| \sum_{i=1}^{M} S_i \bar{n}_i \right|}$$

Where, $\bar{N}$ is the average normal vector of the surface $S$; $\bar{n}_i$ and $S_i$ are the normal and the area of the $i$th triangle ($i = 1, ..., N$) respectively. There is a need to explain that the model of the mold surface is described by STL which is a file format consisted of an abundance of disordered triangle.

Step 2: Transform the coordinate to make the average normal vector $N$ and $Z$ axis parallel.

As shown in Fig.3 a), rotate the facet $S$ to make the average normal vector $N$ parallel with the $Z$ axis. Calculate the new coordinate value of the whole vertexes inside $S$ after rotating. At the same time record the value of $Z$-max and $Z$-min after the rotation.

Step 3: Project the vertexes of the surface to the $XOY$ plane and get the projective point set.

Project the transformed vertexes to the $XOY$ plane. Get the projective point set by removing the duplicate projection vertexes.

Step 4: generate the convex polygon and the minimum bounding rectangle of the projective point set. The specific method is various and the reference [3] is a viable one.

Step 5: Get the preliminary bounding box.

As illustrated in Fig.3 b), stretch the minimum bounding rectangle along the $Z$ axis direction also the rotated normal vector $N'$. During this process, the bounding box will be preliminary determined by the $Z$-max and $Z$-min obtained in step 2.

Step 6: Get the final bounding box by inverse transformation of the rotation.

Make the inverse transformation compared to step 2 and get the bounding box under the coordinate environment before transformation as shown in Fig.3 c). Based on this, the rest is to generate the path.
The improved direction-parallel machining. There is a method of trajectory generation commonly known as direction-parallel machining in surface milling for machine tool. The basic idea is shown in Fig.4 a). Under the coordinate system \( O, X, Y, Z \), when the plane \( ABCD \) paralleling to plane \( Y'O'Z' \) intersects with the mold, the exterior boundary line is just the space trajectory. This way is applicable to any shape especially for the free-form surface. For its mature application in machine tool, the shortcoming is limitation of cutting plane and taking no account of the contour trend. So taking idea of bounding box into consideration, we improve and introduce the direction-parallel method into robotic polishing machining.

According to the prior construction method of the bounding box, a cuboid is got to tightly wrap the mold and thus the information of the placement direction in the current environment is received. Then establish the coordinate for box, choose the average normal vector \( N \) as the \( Z' \) axis direction, and select the bottom plane of the box as the \( X'O'Y' \) plane shown in Fig.4 b). Next, we can choose a face of box as the cutting plane to create the path, and the choice here is the plane paralleling to the average normal vector \( N \) mentioned above. The steps of generating track points are as follows:

**Step 1:** Select the cutting plane and obtain the intersecting line.

Intersect the contour surface with a group of parallel planes paralleling the plane \( X'O'Z' \). The space between the planes is defined by process parameter \( L \). Fig.4 c) shows the projection of the intersecting lines in the bottom plane \( X'O'Y' \).

**Step 2:** Calculate the track points for each intersecting line.

The model of the mold is presented by STL form consisted of plenty of triangles. For each intersecting line, calculate the cross points shown in Fig.4 c).

**Step 3:** Connect all the points along the intersecting line and get the locus sequence.

**Programming and Results**

To validate the feasibility of the method, we realize it in the program and develop a software for trajectory generation by VC++6.0 development tool. The whole flow chart of path generation system for robot polishing is displayed in Fig.5: After obtaining the STL data, first divide the surface into
several patches according to the curvature of the contour, aiming to polish mold along the similar curvature surface. Then introducing the idea of bounding box, surround each patch with a box as small as possible. Last, on the basis of previous work, cut the box with improved direction-parallel machining and calculate the intersection points. Of course, the path generation module is just what we researched in this article. In addition, Fig.6 shows the basic relationships between the modules.

Fig.5 The whole flow chart of path generation system  Fig.6 The relationship map of the modules

To explain the method put forward here is applicable to any surface with any shape and curvature, we choose the mold with irregular shape and different curvature. The generated trajectory is shown in Fig.7. We can see that the density of the track points is variable with different surface curvature. When the surface is relatively flat, the track points are sparse. Yet the points are much dense when the surface is sharp. This just accords with the actual polishing process. That is, the polishing pace is large when polishing the flat surface, and should be adjusted to be short when machining the sharp part avoiding the interference between the end tool and mold surface.

Fig.7 The trajectory of the mold surface for robot polishing
Conclusions

Aiming at the problem of trajectory generation for robotic polishing, this article dexterously combines the improved idea of bounding box with the improved direction-parallel machining, then introduces them into the robotic machining application, last realizes the algorithm in Visual C++6.0 development environment and develops a software for path generation. The results shows that the method was simple, effective and the software supplies a visible trajectory for robot polishing so that the unreasonable points can be modified before machining.

References

[1] Hon-yue Tam, Osmond Chi-hang Liu, Alberet C.K. Mok. Robotic polishing of free-form surfaces using scanning path. Journal of Materials Processing Technology, 1999, 95(1999):191-200.

[2] Zhan Jianming, Zhan Jianbo, Zhou Xiaojin. Research on the path planning of robotic surface polishing[A]. In: 13th Conference of China University Society on Manufacturing Automation [C]. Harbin: 2008.

[3] Zhang Leitao, Hu Yuxia, Cao Xiuge, Yan Liqin. Study of the Minimum Bounding Rectangle Algorithm of Injection Parts [J]. Journal of Gansu Normal Colleges, 2005, 10(2):16-18.

[4] Wang Ping, Yang Yanping, Deng xiao. Study on the Motion Control of Mold Polishing Robot System [J]. Chinese Journal of Mechanical Engineering, 2007, 18(20):2422-2425.

[5] Cheng Pengfei, Yan Haowen, Han Zhenhui. An Algorithm for Computing the Minimum Area Bounding Rectangle of an Arbitrary Polygon [J]. Journal of Engineering Graphics, 2008(1):122-126.

[6] Márquez J J, Pérez J M, Ríos J, Vizán A. Process modeling for robotic polishing. Materials Processing Technology[J], 2005(159):69-82.

[7] Chih-ching Lo and Rong-shine Lin. An improved method for scheduling the tool paths for three axis surface machining. Int. J. Mach. Tool Manuf. 2001, 41(1):133-147.

[8] Ding S, Mannan M A, Poo A N. Oriented bounding box and octree based global interference detection in 5-axis machining of free-form surfaces[J]. Computer-Aided Design, 2004, 36(13):1281-1294.

[9] Chan C K, Tan S T. Determination of the minimum bounding box of an arbitrary solid: an iterative approach[J]. Computers and Structures, 2001, 79:1433-1449.