Hindawi
Advances in Materials Science and Engineering
Volume 2018, Article ID 8697056, 11 pages
https://doi.org/10.1155/2018/8697056

Research Article

Automatic Robot Trajectory for Thermal-Sprayed Complex Surfaces

Dandan Fang,[1,2] You Zheng ,[2,3] Botao Zhang,[4] Xiangbo Li,[5] Pengfei Ju,[6] Hua Li ,[4] and Cunnian Zeng[2]

[1]*Aux Group, Ningbo 315201, China*
[2]*Wuhan University of Technology, Wuhan 430000, China*
[3]*SELF Electronics Co., Ningbo 315201, China*
[4]*Key Laboratory of Marine Materials and Related Technologies, Key Laboratory of Marine Materials and Protective Technologies of Zhejiang Province, Ningbo Institute of Materials Technology and Engineering, Chinese Academy of Sciences, Ningbo 315201, China*
[5]*State Key Laboratory for Marine Corrosion and Protection, Luoyang Ship Material Research Institute (LSMRI), Qingdao 266101, China*
[6]*Shanghai Aerospace Equipment Manufacture, Shanghai 200245, China*

Correspondence should be addressed to You Zheng; zhengyou@nbut.edu.cn and Hua Li; lihua@nimte.ac.cn

Received 15 January 2018; Revised 27 March 2018; Accepted 22 April 2018; Published 8 July 2018

Academic Editor: Shuo Yin

Automatic trajectory generation for thermal spray application is highly desirable for today's automotive manufacturing. Automatic robot trajectory for free-form surfaces to satisfy the coating uniform is still highly challenging due to the complex geometry of free-form surfaces. The purpose of this study is to present and implement a method for automatic generation of robot trajectory according to the given spray parameters on polygon profile and complex curved free-form surfaces, such as torch speed, spray distance, spray angle, and so on. This software development foundation is an Add-In programme of RobotStudio, which is off-line programming and simulation software of ABB Company. The experimental results show that the robot trajectory can be generated rapidly, accurately, and automatically on the complex geometries by this method.

1. Introduction

Robot manipulators are programmable mechanical systems designed to execute many tasks in industry. The advanced robot system has been widely used in thermal spray application in recent years [1–3]. When a robot is used, trajectory generation represents an important part of the work, because the uniformity of spray thickness can significantly influence the quality of coating, and the robot trajectory has a great effect on the uniformity of coating [4–6].

Robot trajectory tells the robot how to move around. It is composed of a series of curves which are defined by the spray parameters in thermal spray operations, and there are several targets which are represented by orientation and Cartesian position on each curve. The generation of robot trajectory on plane surface in thermal spraying is traditionally made by such methods as point-by-point or teaching-playback. This method is tedious and time-consuming. The demand for coating on curved surface has increased over the last few years. It requires hundreds of points in the trajectory and different orientation for each point, so it is very difficult to create such a trajectory by manual point-by-point programming method [7–9]. With the development of the robotics, the robot manufacturers provide software for off-line programming, such as RobotStudio, which is a simulation and off-line programming software of ABB Company [10]. A virtual workshop can be elaborated, and the robot program can be prepared and simulated with RobotStudio.

The complete robot trajectory can be generated manually point by point, which is a very laborious process and not precise in RobotStudio. Furthermore, the functions of RobotStudio are too narrow to process some complex workpieces for thermal spray applications since the fact that spraying trajectory is composed of many lines or curves with certain offset. Consequently, it is necessary to develop a software tool to generate robot trajectory for free-form surfaces according to the given spray parameters.

Most previous researchers typically focused on the mesh method for automated trajectory generation since such method is capable of generating trajectories on all types of complex workpieces [8, 11]. In the mesh method, the mesh of the CAD model is generated at first, then the trajectory from the mesh points and normal vectors to the surface are found out according to graphics theories. Unfortunately, the off-line programming software does not provide any application programming interface (API) for the mesh method. Most of the off-line programming software such as RobotStudio uses the bounding box method to represent 3D models since the bounding box method is efficient in real-time collision detection algorithms. In other words, developers need to develop various functions and programs of the off-line programming software independently. For example, developers must transfer the coordinates of points to the robot system by geometrical transformations. Although some mathematical software such as MATLAB could help realize the functions, too much heavy labor is still involved in software developing and porting.

To blend into the off-line programming software, the cutting method was proposed and applied to generate robot trajectories in thermal spraying. With this method, every point of curves can be obtained by some special APIs to realize object Boolean operations, and the trajectories data also can be calculated and simulated in the off-line programming software directly. It means that the entire development process with the orthogonal cutting method can be simpler and faster than the process with the method of mesh generation. Thermal Spray Toolkit (TST) is one such application, which was developed based on the RobotStudio software for thermal spraying to provide solutions to generate automatic trajectories on various workpieces according to the right spray parameters and trajectory parameters. Some details of TST for the simple workpiece were introduced in [12]. In this paper, we try to answer the question how to generate robot trajectory for complex surfaces automatically.

2. Development Environment

2.1. RobotStudio Software. Off-line programming is the best way to generate robot trajectory on complex geometry. ABB's simulation and off-line programming software, RobotStudio, allow robot programming to be done on a PC in the office without shutting down production. It also enables robot programs to be prepared in advance, which increases overall productivity. RobotStudio provides the tools to increase the profitability of a robot system by performing tasks such as training, programming, and optimization without disturbing production. This provides numerous benefits including risk reduction, quicker start-up, shorter changeover, and increased productivity [10].

RobotStudio is built on the ABB Virtual Controller, an exact copy of the real software that runs the robots in production. It thus allows very realistic simulations to be performed, using real robot programs and configuration files identical to those used on the shop floor. RobotStudio allows creating Add-In programs that enable users to customize and extend its functionality. Thermal Spray Toolkit is an Add-In program to extend RobotStudio's functions.

Currently, there are two different methods for Add-In development. One is creating an Add-In in Visual Studio Tool for Applications (VSTA), and the other is creating an Add-In based on RobotStudio API in Visual Studio.

2.2. RobotStudio API. An Application Programming Interface (API) is an interface implemented by a software program, which enables it to interact with other software. It is similar to the interaction of user interface which facilitates communication between humans and computers. An API is implemented by applications, libraries, and operating systems to determine their vocabularies and calling conventions and used to access their services. It may include specifications for routines, data structures, object classes, and protocols used to create communications between the consumer and the implementer of the API.

RobotStudio API shows the RobotStudio Object Model and explains the methods, properties, and events for each object. In the development of Thermal Spray Toolkit, the RobotStudio API is used.

2.3. Create an Add-In with VSTA. Add-Ins are programs that add optional commands and features to RobotStudio. Before using an Add-In, RobotStudio must be installed on the computer and then loaded in the memory. Loading an Add-In program makes the feature available in RobotStudio and adds any associated commands to the appropriate menus.

Visual Studio Tool for Applications (VSTA) is included in RobotStudio and can allow create Add-Ins in a simple way, but there are some limitations when using VSTA to create the Add-Ins which are caused by Microsoft Proxy Generator. These things will not work:

(i) A method that has an array type as parameter.
(ii) A method or property that references the System. Drawing or System. Windows. Forms namespaces.
(iii) A method or property that references the type.
(iv) An event that is declared static.

Since Add-Ins created with Visual Studio do not use the proxy, these limitations do not apply.

2.4. Create an Add-In in Visual Studio. To be able to use the entire RobotStudio API without limitations, Visual Studio can be used to create an Add-In. All the codes and interface can be developed in Visual Studio (C# or VB.NET), and RobotSutdio can load the Add-In program automatically when it is opened. Thermal Spray Toolkit was developed in environment C#, which is more powerful than VB.NET. The specific steps for creating an Add-In in Visual Studio are listed in Appendix.

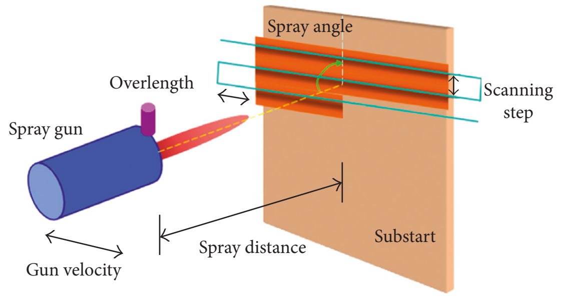


FIGURE 1: The kinematic parameters of cold spray process.

3. Autogeneration of Trajectory

3.1. 3D Model. Computer-aided design (CAD) is a well-known computer-based tool that assists engineers, architects, and other design professionals in their design activities. Model-based object representations are known from computer graphics and CAD. RobotStudio is based on a 3D model. The model of workpiece can be imported to RobotStudio, and trajectory can be generated directly based on the CAD model in RobotStudio [13]. But RobotStudio meets the challenges in thermal spray application. The function of RobotStudio is not powerful enough to deal with the Boolean operation of a 3D model, especially when confronted with the complex surface. It is difficult to get a series of curves on the complex surface according to the spray parameters, such as the robot velocity, the spray distance, the spray angle, the step between each pass, and the over-length. The development of Thermal Spray Toolkit must be based on the API of RobotStudio. As a result, Thermal Spray Toolkit has a great flexibility to deal with complex surfaces.

3.2. Parameters of Cold Spray Process. The kinematic parameters of the cold spray process are demonstrated in Figure 1. Spray distance is the distance from the nozzle exit of spray gun to the substrate surface; scanning step is the distance between two neighbor scanning passes; spray angle is the angle between the centerline of the spray gun and the substrate surface; gun velocity is the relative velocity between spray gun and substrate [14–16].

3.3. The Methodology of Generation of Trajectory. As mentioned above, the methodology of generation of trajectory proposed in this paper is based on the cutting method. Known by the characteristics of thermal spray, the uniform scanning step could keep a stable TCP velocity so that the uniformity of coating can usually be guaranteed. That is, when designing with cutting method, the uniformity of the distance of two neighbor scanning passes should be considered.

With a polygonal substrate, the key of algorithm is to create a plane orthogonal with the polygon profile to equally divide the substrate. In this algorithm, to create an initial plane Π, a point on the polygon should be selected at first. Using the theory of differential geometry [17], the equation of the polygon is described as

$$\begin{aligned} x &= x(t), \\ y &= y(t), \\ z &= z(t), \quad \alpha \le t \le \beta, \end{aligned} \tag{1}$$

$P_0(x(t_0), y(t_0), z(t_0)) \in L$ is the selected point.

Based on the differential geometry [17], the tangent vector $\vec{\tau}$ at this point should be

$$\vec{\tau} = \{x'(t_0), y'(t_0), z'(t_0)\}. \tag{2}$$

And the unit normal vector of the polygon can be generated easily by the vector product of two noncollinear vectors on the polygon.

In the RobotStudio, the scroll bar tools can indicate all the points around the surface for selection, the API function Edge.GetTangent(·) is able to get the tangent vector on point P_0 and the API function Faces.GetNormalToSurface(·) can get the unit normal vector of a polygon.

Particularly, when one edge of the polygon is a straight line, the direction vector on the line is just right to be the tangent vector of the point on the line. Simply connect the two end-points of that line, and the tangent vector can be generated.

When the tangent vector on the point of that curve and the unit normal vector is available, the normal vector of initial orthogonal plane Π is supposed to be

$$\vec{\beta} = \vec{\tau} \times \vec{n} = \{a, b, c\}. \tag{3}$$

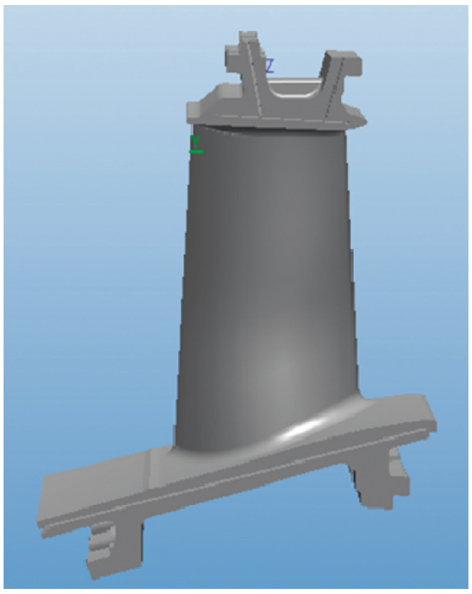
Thus, the initial orthogonal plane equation involving $P_0(x(t_0), y(t_0), z(t_0)) \in L$ on the curve should be

$$a(x(t) - x(t_0)) + b(y(t) - y(t_0)) + c(z(t) - z(t_0)) = 0. \tag{4}$$

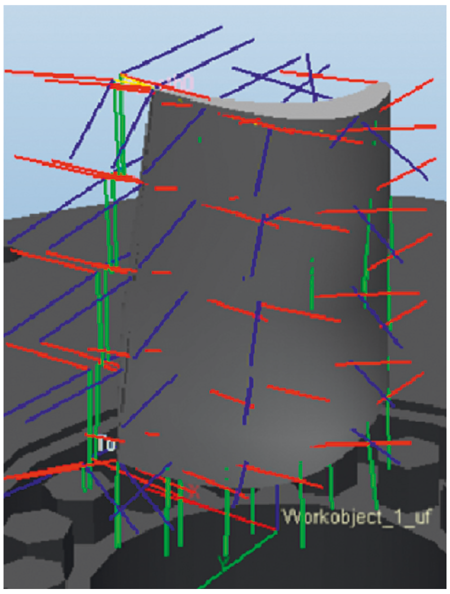
After that, just move the plane up or down with equal distance, and the trajectory with equal scanning step can be generated.

For plane surface, the robot trajectory is simple because the unit normal vector on every point of the surface is the same. But for curved surface, there are many targets on each curve for keeping the same spray distance. Some curved surfaces are very complex and unsymmetrical. For example, as shown in Figure 2, the product is a propeller blade, and its spraying area consists of small surfaces which have different curvatures (the blue lines represent the normal direction of small surfaces, and the red lines represent the tangent direction of small surfaces).

The unit normal vectors on surface vary with the surface curvature, which brings a great challenge to cutting methods. To keep the uniformity of scanning step as much as possible, first of all, select two borders on the substrate, and get some equidistant points on these two borders. Then choose a curve with the largest curvature from the surface on the directions of the two edges, and divide it equally to get some equidistant points. The cutting plane can be created by the three corresponding points on these three lines. Since on a smooth surface, the longest curve is usually the curve with the largest curvature. So, the curves on the same direction of the longest curve can be divided as equally as possible if the longest curve is divided. Also, it is difficult to choose



(a)



(b)

Figure 2: Complex curved surface model.

the curve with the largest curvature. In previous research, the manual selection method was widely used. However, this method is not convenient or accurate enough. Based on the contour line theory in GIS, a method to choose a curve with the largest curvature automatically with the help of the density of contour lines is presented in this paper. According to the contour line theory, the denser the contour, the steeper the slope in the area, and the thinner the contour, the smaller the gradient of the terrain. As shown in Figure 3, from C1 height to C9 height, the contour density along the direction of L1 is higher than that along the direction of L2 and L3, and the path along the direction of L1 is also the shortest. In other words, the denser the contour along a certain direction, the shorter the path along this direction.

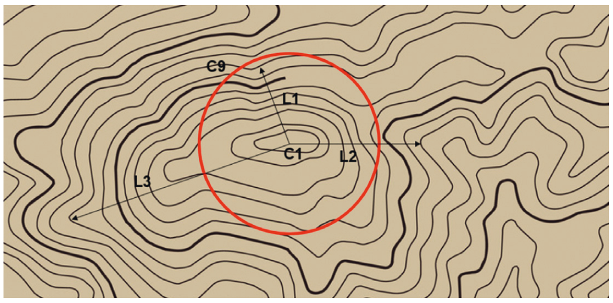


Figure 3: The contour line map.

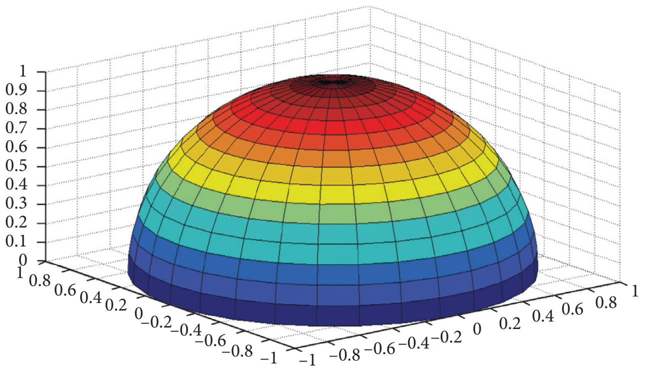


Figure 4: A hemisphere model with a radius of 1 m.

In the following section, the density analysis method is replaced with the path analysis method to find the line with the greatest curvature.

Consequently, starting from the contour line on the top, the path to the edge with the denser is the projection of the curve with the largest curvature. For instance, there is a hemisphere model with a radius of 1 m in Figure 4. And a plane which is parallel to the substrate is used to cut the sphere in every 0.1 m interval, so 11 contour lines can be generated. The distribution of contour lines is as shown in Figure 5.

The result of calculation shows that the length of the shortest path starting from the contour line a is $d_{\min} = n1 + n2 + \cdots + n11$ along with the radius direction of the sphere. It is because the shortest path between two contour lines is along the direction of radius. As shown in Figure 5, starting from point b, the length of $n2$ on the radius direction is shorter than any other path (e.g., path $m2$). Using the shortest path and the unit normal vector of plane at the bottom, a new plane or surface can be created. The intersection line of the plane or the surface is the curve with the largest curvature. As shown in Figure 6, the curve n is the largest curvature selected on the hemisphere in Figure 4.

By cutting planes (polygon cases) or objects (surface cases), a series of lines could be generated. Then, find out the endpoints of each line, and calculate the three-dimensional coordinate (x, y, z) and quaternion-vector ($q1$, $q2$, $q3$, $q4$) of these endpoints in the world coordinate system by some functions. The quaternion-vector ($q1$, $q2$, $q3$, $q4$) is used to represent the orientation of endpoint in the three-dimensional space.

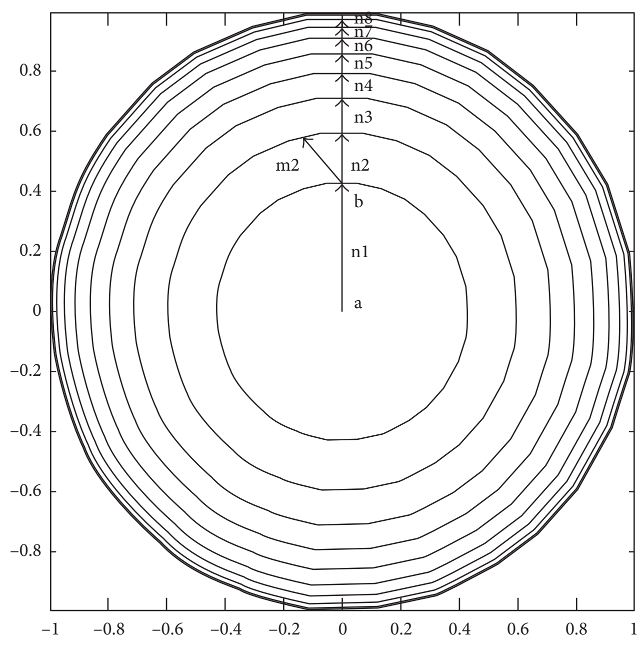


FIGURE 5: The contour lines of the hemisphere model.

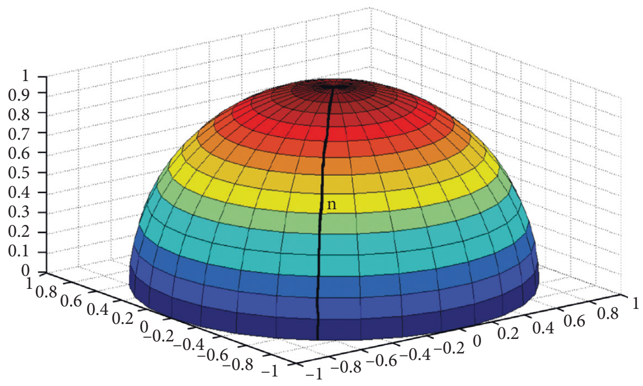


FIGURE 6: The largest curvature n selected on the hemisphere.

Finally, the spray parameter over-length is defined. Theoretically, the over-length is equal to the distance during which the robot accelerates from a halt to a necessary spray speed. In such a way, the scanning speed on the workpiece could be constant and the overheating on the edge could be avoided. Furthermore, it means obviously a great amount of power savings. Thus, the best solution is to keep a constant over-length of each point on the edge of the workpiece. Since different types of robot have different kinematic parameters, the over-length depends on a specific robot type. It is very difficult to find the exact position of these over-lengths on real workpiece in the work cell, even in off-line programming under RobotStudio.

For clarity of description of the algorithm above, two examples using an arbitrary polygon and a curved surface to generate trajectory are used for illustration, respectively.

3.4. Automatic Generation on a Polygon Profile. The method used for automatic generation on a curved-edge polygon profile is as follows, and the sketch map is shown in Figure 7:

(1) Click on the surface of CAD model workpiece in the graphics window.
(2) Pull the scroll bar which indicates all the points around the surface, the direction of tangent of this point is the direction of trajectory, and then choose the trajectory starting from left or right.
(3) Create a large plane involving two points to intersect the spraying surface and ensure the large plane and the spraying surface are orthogonal. Create the first intersection line.
(4) According to the value of step, move down the large plane and create a series of lines.
(5) Create targets on both the ends of series of lines according to the interlength. Calculate the robot velocity and orientation for each target.
(6) Add all the targets to trajectory in order.

This function is not only suitable to round and ellipse, but also suitable to many irregular curved-edge polygon surfaces. With the straight-edge polygon profile, the straight edge is selected instead of the point on the edge. In this way, autogeneration of trajectory could be simpler. The method used for automatic generation on a straight-edge polygon profile is as follows, and the sketch map is shown in Figure 8:

(1) Click on the surface of the CAD model workpiece in the graphics window, all the edges of the surface will be displayed in the list.
(2) Select one edge as the direction of trajectory and choose the trajectory starting from left or right.
(3) Create a large plane on the edge to intersect the spraying surface, and ensure that the large plane and the spraying surface are orthogonal. Create the first intersection line.
(4) According to the value of step, move down the large plane. Create a series of lines.
(5) Create targets on both the ends of series of lines according to the interlength. Calculate the robot velocity and orientation for each target.
(6) Add all the targets to trajectory in order.

3.5. AutoGeneration for Curved Surface. The following presents the method of autogeneration for a curved surface, and the sketch map is shown in Figure 9.

(1) Create two edges of the complex 3D coating surface.
(2) The two edges are divided into equal parts according to the spray parameters.
(3) Find the line with the greatest curvature of the complex 3D coating surface on the directions of the two edges, and divide this line equally to get some equidistant points. Create a large plane through the corresponding points on the edges and the line with

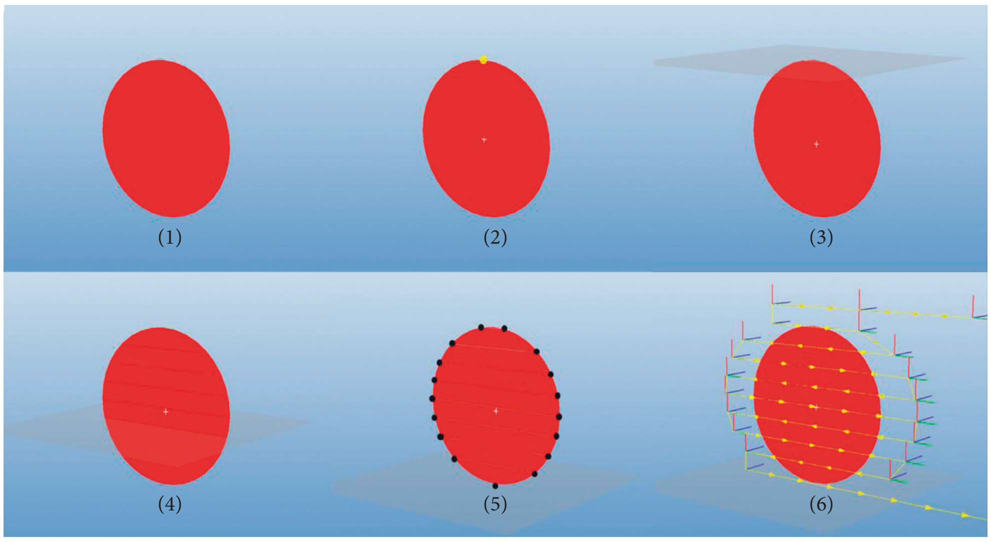


Figure 7: Autogeneration on a curved-edge polygon profile.

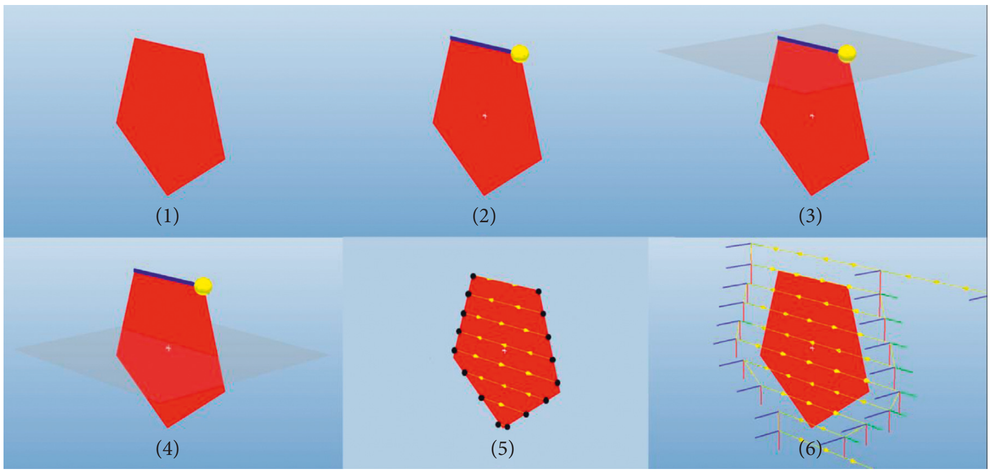


Figure 8: Autogeneration on a straight-edge polygon profile.

the greatest curvature to intersect this 3D model. Create the first closed intersection curve. This step is suitable for a smooth curved surface. For an irregular curved surface, manual adjustment is necessary to ensure that these closed intersection curves are equidistant.

(4) Offset the large plane to the next corresponding points on the edges and the line with the greatest curvature to create a series of closed intersection curves.

(5) Choose the part of closed intersection curves which are located on the spraying area.

(6) Create targets on the series of selected curves according to the interlength. Calculate the robot velocity and orientation for each target. Add all the targets to trajectory in order.

3.6. Calculation of Velocity. Robot velocity plays a significant role in coating quality. A robot usually has a continual motion and robot velocity is planned as constant as possible to obtain a uniform thickness if the step is constant. But with the methodology used in thermal spray toolkit, the step is varied, so the robot velocity should change with the step. In theory, the robot velocity is inversely proportional to the step: $v2/v1 = d1/d2$. Figure 10 shows the meaning of character $v1$, $v2$, $d1$ and $d2$. The robot velocities are calculated in the Add-In program and assigned to corresponding targets in order to keep the uniformity.

4. Program Implementation and Experimentations

To validate the methodology and visualize the trajectories, an Add-In programme with interface has been developed in the C# environment, and it can be loaded automatically when opening the RobotStudio. The function of the program has been tested using a different 2D model and 3D model. Furthermore, the real experiment has been implemented to evaluate the effect of autogeneration of trajectory.

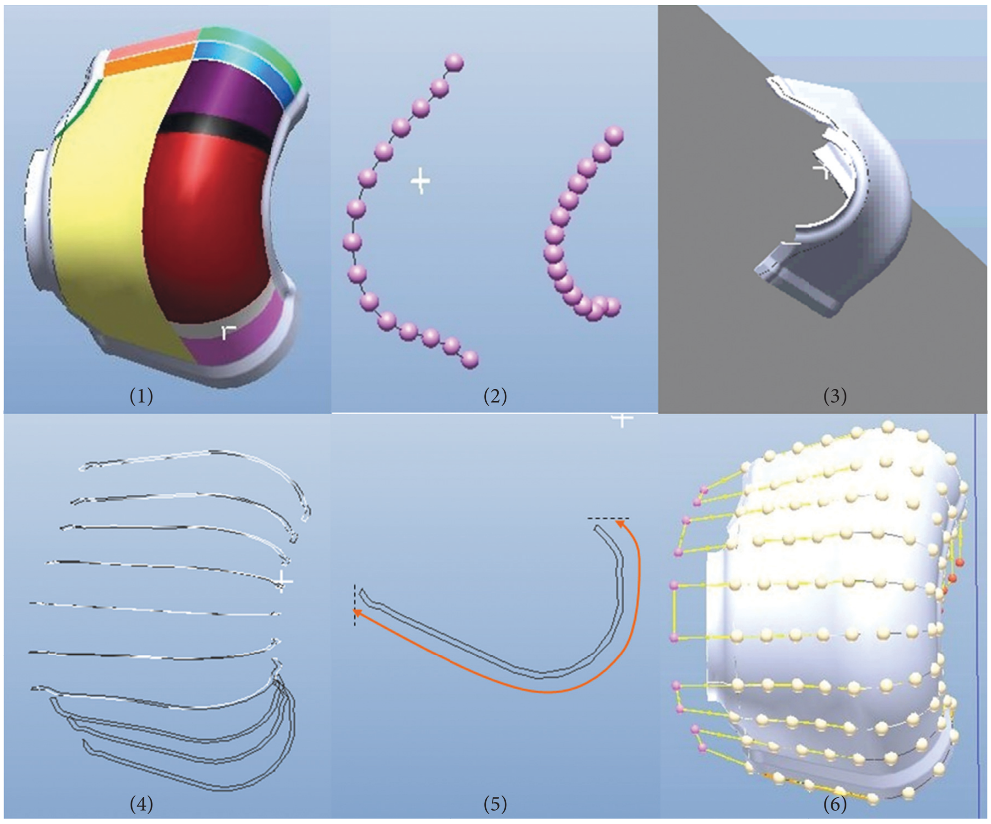


Figure 9: Autogeneration for complex curved surface.

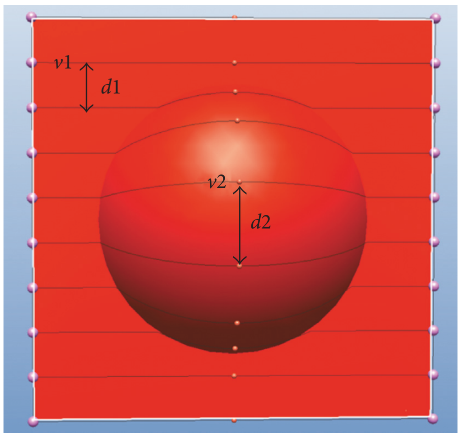


Figure 10: Meaning of character $v1$, $v2$, $d1$, and $d2$.

4.1. Implementation on a Straight-Edge Polygon Profile. The following figures provide some typical examples to illustrate the function of autogeneration of trajectory on a polygon profile. According to the thermal spray characteristics, it can be clearly seen that the first line and the last line of trajectory are located outside the workpiece in order to ensure the uniformity of coating. Figure 11 shows the autogenerated trajectory on a type of regular polygons, and Figure 12 shows the autogenerated trajectory on a type of irregular polygons.

4.2. Implementation on a Curved-Edge Polygon Profile. As seen from Figure 13, the autogenerated trajectory on different kinds of curved-edge polygons can be realized by TST. The beginning part and the end part of the trajectory are prolonged to a certain distance as the start and end of torch, respectively.

4.3. Implementation of on Free-Form Surfaces. Figure 14 shows the program implementation of autogeneration of trajectory on free-form surfaces. As shown in the figure, the function can process various complex 3D workpieces, even propeller blades.

4.4. Experiments and Analysis. It is well known that the scanning speed is the most important parameter for the coating homogeneity during the deposition. So, the experiment is implemented to analyse the TCP velocity according to the selected robot trajectory. As a professional robot off-line programming platform, RobotStudio is able to supply the kinematics data in simulations, which are highly close with the data generated in actual motions of robots. The simulation process of RobotStudio enables users to obtain the real-time value of TCP position, speed, and the rotation angle of robot axis and to implement the collision detection between robot, torch, and workpiece. A virtual experimental environment which is completely consistent with the real experimental environment is

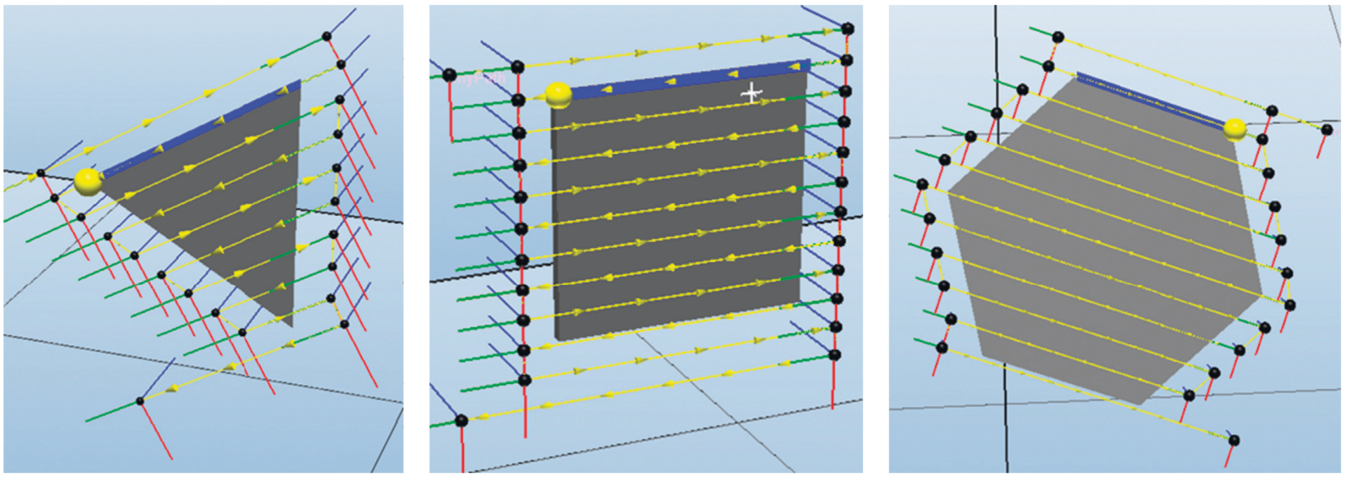

(a) (b) (c)

Figure 11: Autogenerated trajectory on a type of regular straight-edge polygons.

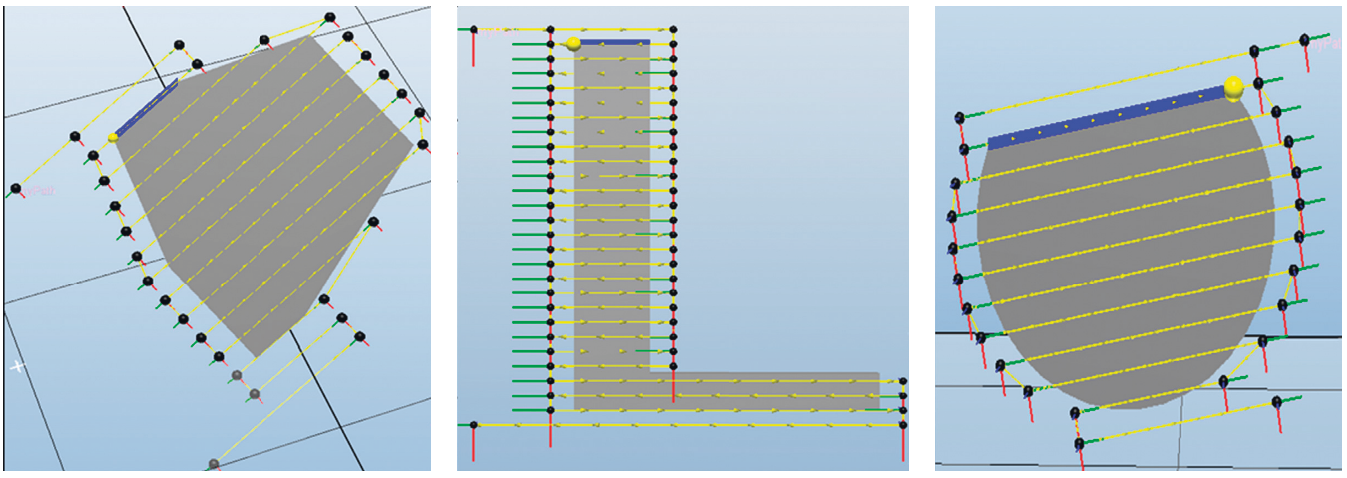

(a) (b) (c)

Figure 12: Autogenerated trajectory on a type of irregular straight-edge polygons.

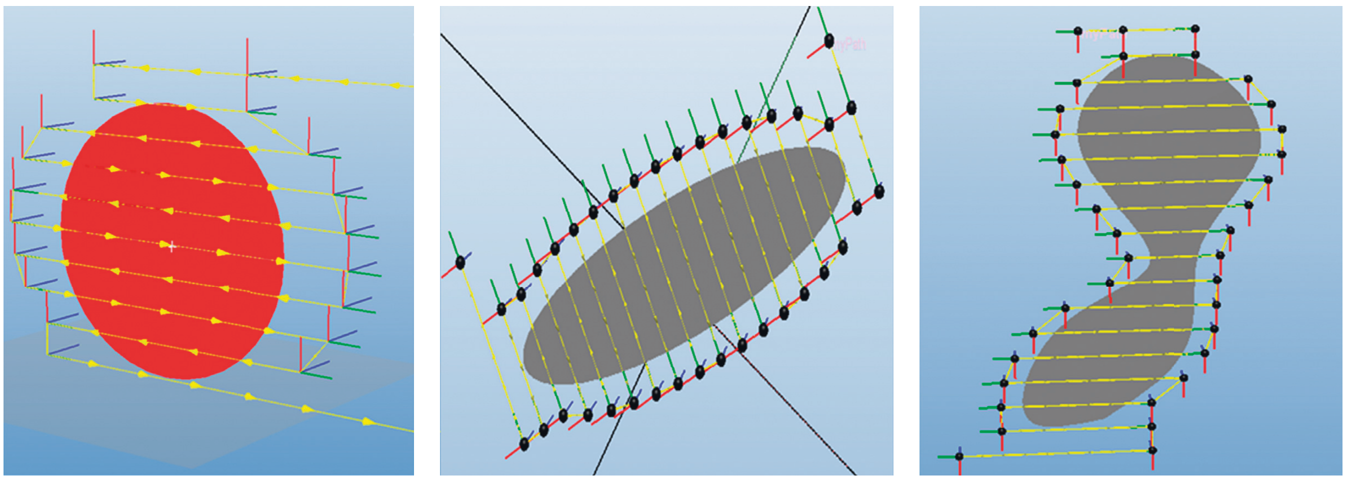

(a) (b) (c)

Figure 13: Autogenerated trajectory on a type of curved-edge polygons.

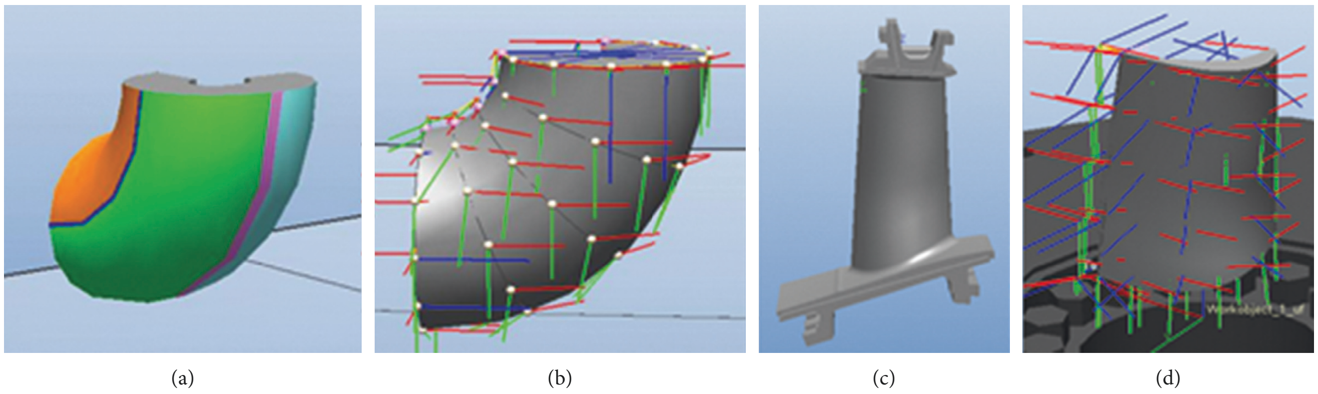


Figure 14: Autogeneration of trajectory on free-form surfaces.

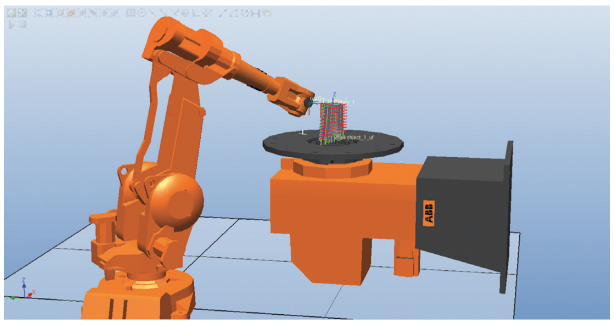


Figure 15: The virtual experimental environment.

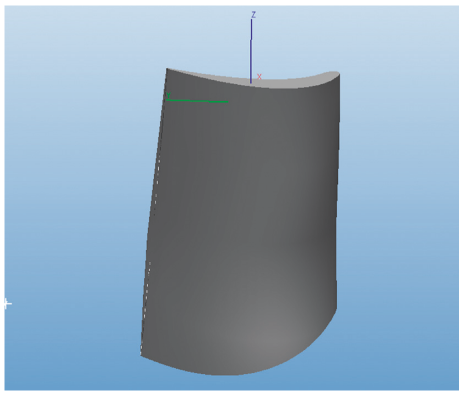


Figure 16: The shape of the substrate surface.

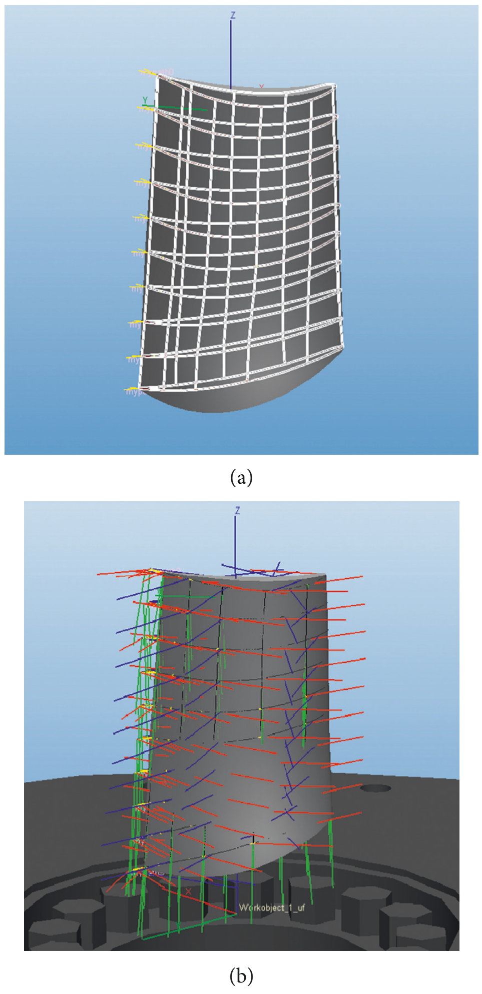


Figure 17: A series of curves (a) and several targets (b) in the robot trajectories.

created as shown in Figure 15, which simulated the spray process to get the record of TCP velocity. The shape of the substrate surface is described in Figure 16. A series of curves and several targets in the robot trajectories generated by TST are shown in Figures 17(a) and 17(b), respectively.

AF4-MB type torch is guided by a robot of IRB4400. The scanning step is 0.022 m and the over-length is 0.03 m. During the simulation process, the real-time TCP velocity and position are recorded on each trajectory point. The simulated TCP velocity and the calculated TCP velocity for robot trajectory are shown in Figure 18, respectively.

When passing through the complex curved surface, each pass has 5 samples, and the starting point is on the left of the model. As mentioned in the calculation of velocity section, the TCP velocity on first pass is the reference velocity which equals to 300 mm/s in this experiment. Theoretically, the

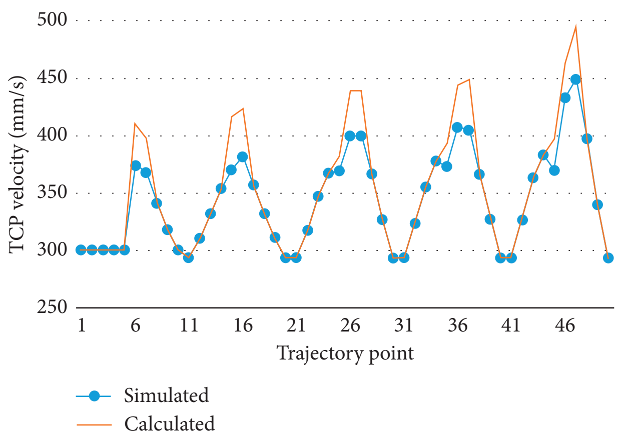


Figure 18: The simulated TCP velocity and the calculated TCP velocity for robot trajectory.

robot velocity is inversely proportional to the scanning step. Since the left border of the model is wider than the right border of the model, the TCP velocity of the pass from right to left should be increasing. In the same way, the TCP velocity of the pass from left to right should be decreasing. The simulated TCP velocity (blue line) in Figure 18 is basically close to the calculated TCP velocity (red line), which is inversely proportional to the scanning step. It means that the method for autogeneration trajectory works well for thermal spray application. The simulated TCP velocity decreased by about 10% less than the calculated TCP velocity that corresponds to the passage of torch at the rounded area of the workpiece where the rotation is the most difficult.

5. Conclusion

Autogeneration trajectory is needed to operate the spray process on a free-form surface, but no off-line programming system has yet been developed to meet this need. In this paper, Thermal Spray Toolkit was developed based on RobotStudio, it is flexible and robust enough to deal with any complex 2D and 3D model, and the robotic trajectory can be generated automatically according to the spray parameters with high efficiency.

Appendix

(1) Create a new project with Class Library as your template.

(2) To use the RobotStudio API, you need to reference the ABB.Robotics.*.dll assemblies. These are as follows:

ABB.Robotics.Math—Vector and matrix math.
ABB.Robotics.RobotStudio—General and top-level (e.g., nonstation specific) classes.
ABB.Robotics.RobotStudio.Environment—For manipulating the GUI, for example, adding menus, buttons and so on.
ABB.Robotics.RobotStudio.Stations—For manipulating stations and their contents.
ABB.Robotics.RobotStudio.Stations.Forms—GUI controls used by RobotStudio.
You can add a new reference to your project by right click on References in the Solution Explorer and choose Add Reference....
Go to the Browse tab and browse to your RobotStudio installation directory. This is where you will find the dll.

(3) Add the following lines to the top of your code:

```
CopyC#
using ABB.Robotics.Math;
using ABB.Robotics.RobotStudio;
using ABB.Robotics.RobotStudio.Environment;
using ABB.Robotics.RobotStudio.Stations;
using ABB.Robotics.RobotStudio.Stations.Forms;
```

(4) Add the following code to the class:

```
CopyC#
public static void AddinMain()
{
  //This is where you write your code.
}
```

The AddinMain()-method is the entry point for your Add-In.

(5) Go to the properties of your project (right click on your project in the Solution Explorer and select Properties) and select the Build Events. Add the following line to the Post-built event command line:

```
XCopy/y "$(TargetPath)" "C:\Program Files\ABB Industrial IT\Robotics IT\RobotStudio 5.12\Bin \Addins\"
```

Assuming that this is the directory for your RobotStudio installation.
This will copy your Add-In dll-file to the RobotStudio Add-In directory when building your Add-In.

(6) You could also add the path to RobotStudio.exe to Start external program under Debug in Properties, so RobotStudio will start every time you want to debug your Add-In.

(7) Write your code (or copy and paste one of the examples into AddinMain()).

(8) Select Start Debugging from Debug menu (or press F5). This will start RobotStudio and autoload your Add-In. Since your Add-In will load and execute directly when RobotStudio starts (even though there is not active station) it will probably throw an exception. It might be a good idea to add a menu button to start your code. Have a look at the Add Menus and Buttons Example for how to create menus and buttons.

(9) If you go to the Add-Ins menu you can find your Add-In under the General folder. If you right click on your Add-In you can select whether or not it should autoload, and you can load it if it is not already loaded.

RobotStudio Add-Ins can be created in VB.NET or C#.

Data Availability

The data used to support the findings of this study are not available from the corresponding author upon request due to the commercial interests in the research.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this article.

Acknowledgments

This work was supported by the National Natural Science Foundation of China (31500772, 21705158, 51401127, and 51771122), Key Research and Development Program of Zhejiang Province (2015C01036 and 2107C01003), Zhejiang Provincial Natural Science Foundation of China (LQ14F010001, LY15F010006, LY16F10014, LQ15F020004, and LY18F020025), Key Scientific and Technological Projects of Ningbo (2014B92001), Postdoctoral Science Foundation of Zhejiang Province (BSH02020), Ningbo Natural Science Foundation (2017A610109 and 2015A610132), Research Fund of State Key Laboratory for Marine Corrosion and Protection of Luoyang Ship Material Research Institute (LSMRI) under the Contract no. KF160409, International Scientific and Technological Cooperation Project of Ningbo, and Science and Technology Commission of Shanghai Municipality (Grant nos. 16QB1401100 and 15QB1401500).

References

[1] R. Gadow and M. Floristán, "Manufacturing engineering in thermal spraying by advanced robot systems and process kinematics," in *Future Development of Thermal Spray Coatings*, pp. 259–280, Elsevier, New York, NY, USA, 2015.

[2] N. Espallargas, *Future Development of Thermal Spray Coatings: Types, Designs, Manufacture and Applications*, Elsevier, New York, NY, USA, 2015.

[3] S. Deng and C. Chen, "Generation of robot trajectory for thermal spray," in *Robot Kinematics and Motion Planning*, Nova Science Publishers, Hauppauge, NY, USA, 2015.

[4] M. Floristán, J. A. Montesinos, J. A. García-Marín, A. Killinger, and R. Gadow, *Robot Trajectory Planning for High Quality Thermal Spray Coating Processes on Complex Shaped Components*, ASM International, Geauga County, OH, USA, 2012.

[5] D. Fang, S. Deng, H. Liao, and C. Coddet, "The effect of robot kinematics on the coating thickness uniformity," *Journal of Thermal Spray Technology*, vol. 19, no. 4, pp. 796–804, 2010.

[6] R. Gadow, A. Candel, and M. Floristán, "Optimized robot trajectory generation for thermal spraying operations and high quality coatings on free-form surfaces," *Surface and Coatings Technology*, vol. 205, no. 4, pp. 1074–1079, 2010.

[7] B. Zhou, X. Zhang, Z. Meng, and X. Dai, "Off-line programming system of industrial robot for spraying manufacturing optimization," in *Proceedings of the 33th Chinese Control Conference*, pp. 8495–8500, Nanjing, China, July 2014.

[8] A. Candel and R. Gadow, "Trajectory generation and coupled numerical simulation for thermal spraying applications on complex geometries," *Journal of Thermal Spray Technology*, vol. 18, no. 5-6, pp. 981–987, 2009.

[9] D. Hegels, T. Wiederkehr, and H. Müller, "Simulation based iterative post-optimization of paths of robot guided thermal spraying," *Robotics and Computer-Integrated Manufacturing*, vol. 35, pp. 1–15, 2015.

[10] ABB Robotics, *RobotStudio™ Users Guide*, ABB Robotic Products, Zürich, Switzerland, 2015.

[11] P. N. Atkar, A. Greenfield, D. C. Conner, H. Choset, and A. A. Rizzi, "Uniform coverage of automotive surface patches," *International Journal of Robotics Research*, vol. 24, no. 11, pp. 883–898, 2005.

[12] S. Deng, H. Liao, and C. Coddet, "Robotic trajectory autogeneration in thermal spraying," in *Proceedings of the Thermal Spray Connects: Explore Its Surfacing Potential*, Basel, Switzerland, May 2005.

[13] R. Holubek, D. R. Delgado Sobrino, P. Košťál, and R. Ružarovský, "Offline programming of an ABB robot using imported CAD models in the RobotStudio software environment," *Applied Mechanics and Materials*, vol. 693, pp. 62–67, 2014.

[14] K. Wen, M. Liu, K. Zhou et al., "The influence of anode inner contour on atmospheric DC plasma spraying process," *Advances in Materials Science and Engineering*, vol. 2017, Article ID 2084363, 12 pages, 2017.

[15] P. Ctibor, M. Kašparová, J. Bellin, E. Le Guen, and F. Zahálka, "Plasma spraying and characterization of tungsten carbide-cobalt coatings by the water-stabilized system WSP," *Advances in Materials Science and Engineering*, vol. 2009, Article ID 254848, 11 pages, 2009.

[16] L. Pawlowski, *The Science and Engineering of Thermal Spray Coatings*, John Wiley & Sons, Hoboken, NY, USA, 2nd edition, 2008.

[17] V. Y. Rovenski, *Differential Geometry of Curves and Surfaces*, Prentice-Hall Inc., Upper Saddle River, NY, USA, 2004.