Research Article

INTERNATIONAL JOURNAL OF ADVANCED ROBOTIC SYSTEMS

International Journal of Advanced Robotic Systems
November-December 2021: 1–11

DOI: 10.1177/17298814211064767
journals.sagepub.com/home/arx
SAGE

Novel surface optimization for trajectory reconstruction in industrial robot tasks

Miguel Angel Funes-Lora[1], Eduardo Vega-Alvarado[2], Raúl Rivera-Blas[2], María Barbara Calva-Yáñez[2] and Gabriel Sepúlveda-Cervantes[2]

Abstract

This study presents a novel algorithm implementation that optimizes manually recorded toolpaths with the use of a 3D-workpiece model to reduce manual error induced. The novel algorithm has three steps: workpiece declaration, manual toolpath declaration, and toolpath optimization using steepest descent algorithm. Steepest descent finds the surface route wherein the manually recorded toolpaths traverse over a 3D-workpiece surface. The optimized toolpaths were simulated and tested with an industrial robot showing minimal error compared to the desired optimized toolpaths. The results obtained from the presented implementation on three different trajectories demonstrate that the proposed methodology can reduce the manual error induced using as a reference the CAD-workpiece surface.

Keywords

Free-form surface, industrial robot, optimization algorithm, point cloud data

Date received: 31 March 2021; accepted: 18 November 2021

Topic: Robot Manipulation and Control
Topic Editor: Marco Ceccarelli
Associate Editor: Kenji Hashimoto

Introduction

The industrial robot–workpiece interaction has an important role to perform tasks over complex surfaces such as painting[1–3] milling,[4] grinding,[5] laser cladding,[6] nondestructive examinations,[7,8] and robotic automated fiber placement applications.[9,10] The path planning can be classified into online and offline programming. In online programming, the main disadvantages are the time spent on manual robot programming and the manual error induced (error due to operator skills-dependency). On another side, the performance of off-line path planning methods is faster and often model-based 3D CAD.[7,11–13] Some robotic applications take advantage of 3D-workpiece for diverse applications, for example, in Xingguo and Li,[14] a robotic model-based offline programming method made the projection of a 2D-plane onto the surface of a CAD model for machining processes, taking into consideration the spin angle of the tool. If a CAD model is not available, it is possible to take advantage of reverse engineering techniques using scanned data[15,16] offering discrete 3D-workpiece models where special algorithms generate trajectories on point clouds and meshes.[8,13] In an earlier study,[3] the authors propose an incremental approach for automated trajectory generation from CAD model, the

[1] The University of Michigan, Ann Arbor, MI, USA
[2] Instituto Politécnico Nacional, Ciudad de México, Mexico

Corresponding author:
Eduardo Vega-Alvarado, Instituto Politécnico Nacional - México CIDETEC - Group of Research and Innovation in Mechatronics, Ciudad de México, MX-MEX 07738, Mexico.
Email: evega@ipn.mx

whole surface is meshed and divided into surface sections (SS) based on spray radius, and these SS are painted by spray gun located at a standoff distance from the surface section point (SSP) along the surface section normal (SSN). To calculate the SSP, the weighted average of all the centroids of the triangle inside the SS is used and the weighted average of the normals of the triangles in the SS determines the SSN. Subsequently, a new SS is generated with an incremental distance along the x axis, steps are repeated till the end of the surface is reached, and the incremental distance in y axis depends on the optimal overlap distance. To generate the optimal overlap distance and the velocity for a given paint pass (SS), a genetic algorithm is used and the process ends when the whole surface is reached. Li et al.[6] propose a path planning method whose key lies in the topological reconstruction of the artificial joint surface. From an STL file, the information to create the relations among points, edges, and facets is extracted, facilitating the path planning method. Via equidistant parallel planes that intersect the CAD model, a series of intersections point were generated (slicing algorithm) to obtain transversal lines that use the topological relation of the edges instead of using a sorting algorithm to reduce the number of intersection calculations. The height error method is used for the robot interpolation points and the normal vectors of each point can calculate the position and pose of the robot tool center point (TCP) to ensure the perpendicularity between the laser beam and the surface. Finally, by connecting the interpolation points sequentially, the motion path is obtained.

Other works have addressed the problem by mesh following techniques using a triangular mesh of a surface as a guide for path-planning algorithms; Mineo et al.[7,8] present a flexible off-line trajectory planning for inspection of complex curved surfaces on nondestructive testing systems. The toolpath is generated to follow the contour of a triangular meshed CAD surface without the need for an approximating analytical surface. To generate the path planning, it is necessary to create a curve contour of the surface, as the surface is a triangular mesh, the surface edge is a segment with extremities of two adjacent triangles, then always is possible to find the intersections points between the plane perpendicular to the segment and the edges of the triangles of the surface mesh. The curvilinear distance is calculated by cumulating the segment of the intersection points from the reference edge that is the distance along the surface contour where every point of the curve must be normal to the relative triangle.

This article presents a novel algorithm implementation that combines the skills of the operator and the advantages of the CAD model to optimize manually recorded toolpaths with the use of 3D-workpiece models to reduce manual error induced avoiding exhaustive touch up reprogramming when considering large, workpiece surfaces. The procedure was tested both with a simulation and experimentation on a real industrial robot arm; model ABB IRB 1600-7/1.45 type A. The results show a high-quality trajectory reproduction, allowing more accurate adjustments on manually recorded toolpaths.

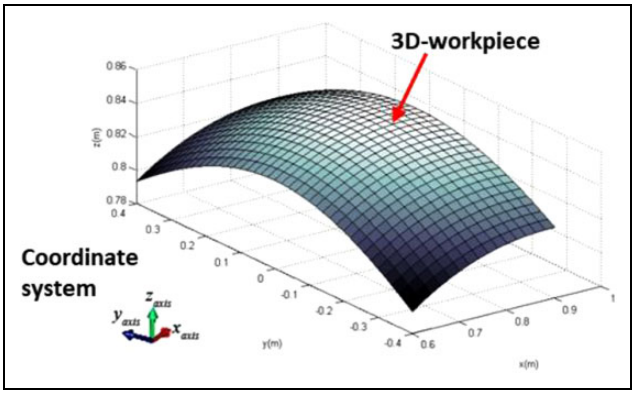

Figure 1. Point cloud data representing the 3D-workpiece surface displayed with the surf instruction in MATLAB.

The organization of this article is divided into four sections: Methodology presents a generalized explanation of the proposed algorithm for optimizing manually recorded trajectories, Steepest Descent algorithm selects the number of nodes in the objective function for reconstructing the surface route on the 3D-workpiece surface. Experimentation and Results were carried out at ABB industrial robot using three different trajectories where the mean absolute error was calculated by comparing the robot TCP trajectory and the desired optimized trajectory. Finally, the Conclusions and Future Work are presented.

Methodology

The presented novel algorithm implementation combines the manually recorded toolpaths stored in a text file with the use of a 3D-workpiece-model to optimize robotic toolpaths and reduce manual error induced. The steepest descent algorithm finds the surface route wherein the manually recorded toolpaths traverse over the CAD-workpiece. The three-step procedure of the novel algorithm implementation is presented as follows:

Step 1: Workpiece declaration

The first step is the workpiece declaration, Figure 1 shows a curved surface that represents a 3D-workpiece using point cloud data. This figure includes 656 points, covering an area of 0.24 m^2. The points were uniformly distributed over the surface but can also be randomly distributed. The application of the presented methodology focuses directly on surjective CAD surfaces on 3D as in previous works,[8,13] where every z coordinate in the codomain Z from the workpiece surface has the corresponding xy-couple coordinate such that $z = f(x, y)$. The shape of the CAD-workpiece surface and its coordinate reference frame determine if the workpiece is surjective.

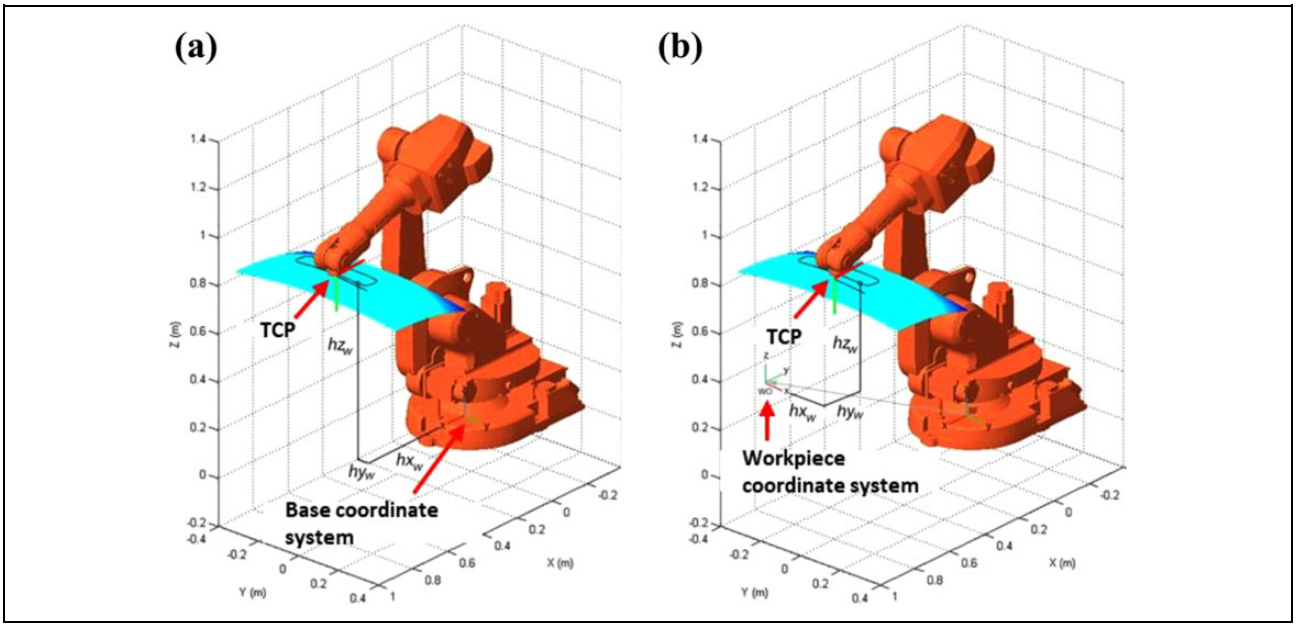

Figure 2. (a) Reference of a 3D-workpiece point DBW^i with respect to the base coordinate system of the robot. (b) Reference of a 3D-workpiece point DBW^i with respect to the workpiece coordinate system.

The manually recorded toolpaths and the point cloud data that conform to the CAD-workpiece can be referenced to the robot base coordinate system or any other specific coordinate system (see Figure 2). This characteristic allows identifying the distancing hx_w, hy_w, and hz_w from any coordinate system with respect to any point from the CAD surface, wherein any point can be defined as (DBW^i) for $i =$ number of points that conforms the 3D-workpiece. The presented methodology also considers a TCP in the flange of the robot or a tool attached to it.

Step 2: Manual toolpath declaration

The second step is the declaration of manually recorded toolpaths. These manually recorded toolpaths conventionally present variable standoff distancing between TCP and CAD-workpiece (represented as point cloud data), due to manually induced error. Every manually recorded toolpath can be traversed using at least two poses sequentially, and the result can be a straight toolpath that is subdivided into tiny segments.[17] As an example, Figure 3 shows five poses $(P_1, P_2, P_3, P_4, P_5)$ to create four straight toolpaths.[17] Every pose can be represented as shown in equation (1), where hx_f^i, hy_f^i, and hz_f^i correspond to the TCP of the robot, and E_z^i, E_y^i, and E_x^i represent the orientation in Euler angles, such that $\text{Euler}_{ZYX}(\alpha^i, \beta^i, \gamma^i)$, where α^i, β^i, and γ^i are the rotation about Z, Y, and X axes, respectively.[18] Two successive poses defined as P_{gpp}^i and $P_{gpp}^{(i+1)}$ can be expressed as in equations (1) and (2). For an "n" number sequenced toolpaths, it is possible to define P_{gpp} as a matrix of M_{nx6}

$$P_{gpp}^i = [\,hx_f^i \quad hy_f^i \quad hz_f^i \quad E_z^i \quad E_y^i \quad E_x^i\,] \tag{1}$$

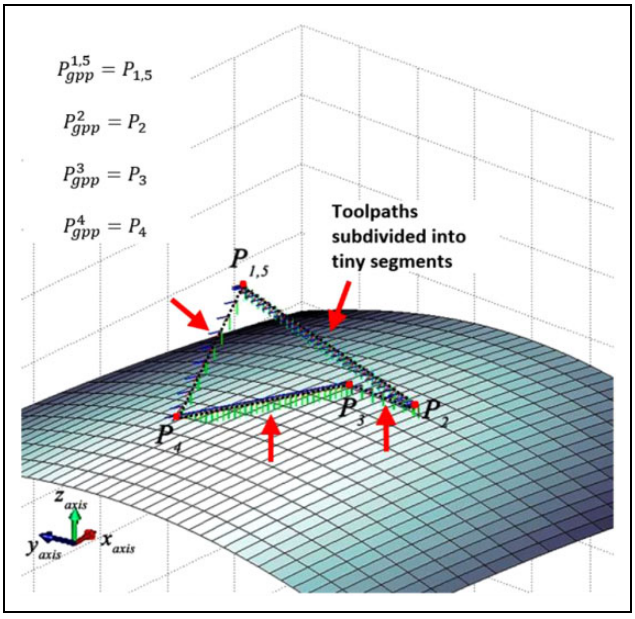

Figure 3. Example of linear interpolations created by five manually recorded poses.

and

$$P_{gpp}^{i+1} = [\,hx_f^{i+1} \quad hy_f^{i+1} \quad hz_f^{i+1} \quad E_z^{i+1} \quad E_y^{i+1} \quad E_x^{i+1}\,] \tag{2}$$

$$P_{gpp} = \begin{bmatrix} hx_f^1 & hy_f^1 & hz_f^1 & E_z^1 & E_y^1 & E_x^1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ hx_f^n & hy_f^n & hz_f^n & E_z^n & E_y^n & E_x^n \end{bmatrix} \tag{3}$$

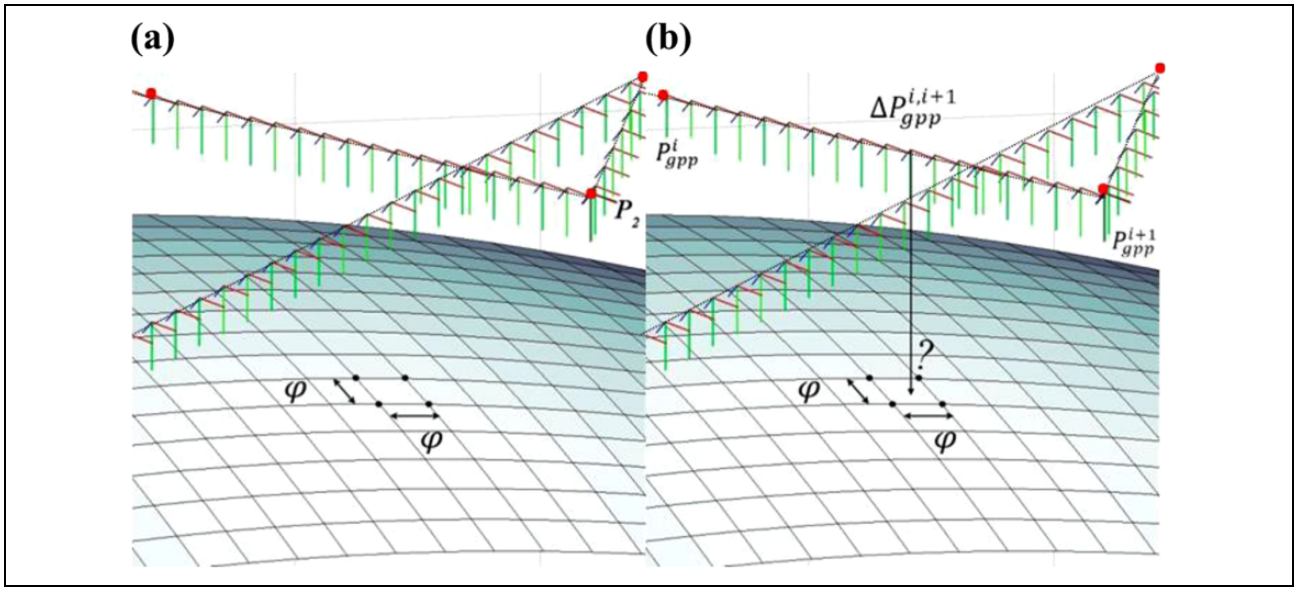

Figure 4. (a) Distance between adjacent points ϕ in the point cloud data and (b) segment of a straight toolpath $\Delta P^{i,i+1}_{gpp}$ to be optimized.

Every element of P_{gpp} can also be expressed by a matrix of 4×4 elements, where the first three columns represent the orientation and the last column is the corresponding position; for this case, P^i_{gpp} is expressed as in equation (4), where $s(\delta) = \sin(\delta)$ and $c(\delta) = \cos(\delta)$, as a simplification for any angle δ

$$P^i_{gpp} = \begin{bmatrix} c(\alpha^i)c(\beta^i) & c(\alpha^i)s(\beta^i)s(\gamma^i) - s(\alpha^i)c(\gamma^i) & c(\alpha^i)s(\beta^i)c(\gamma^i) + s(\alpha^i)s(\gamma^i) & hx \\ s(\alpha^i)c(\beta^i) & s(\alpha^i)s(\beta^i)s(\gamma^i) + c(\alpha^i)c(\gamma^i) & s(\alpha^i)s(\beta^i)c(\gamma^i) - c(\alpha^i)s(\gamma^i) & hy^i_f \\ -s(\beta^i) & c(\beta^i)s(\gamma^i) & c(\beta^i)c(\gamma^i) & hz^i_f \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{4}$$

As a result of implementing equation (4),[17] straight toolpaths subdivided into tiny segments were created as a first approximant reference of the desired optimized trajectories.

Step 3: Toolpath optimization

In the third step, every segment from the straight toolpaths "$\Delta P^{i,i+1}_{gpp}$" is optimized using the steepest descent algorithm. The algorithm finds an approximation of the 3D-workpiece surface, where every $\Delta P^{i,i+1}_{gpp}$ traverse over the CAD-workpiece. The vector $\Delta P^{i,i+1}_{gpp}$ represents every possible transition from P^i_{gpp} to P^{i+1}_{gpp}; therefore, this term can be converted from Euler angles to a representation using homogeneous transformation matrices, $\Delta P^{i,i+1}_{gpp}$, P^i_{gpp}, and $P^{i+1}_{gpp} \in M_{4x4}$ are represented similarly to equation (4)[17] (see equation (5)). These matrices correspond to the translation and orientation with respect to the TCP. Thus, it can be concluded that any transition from P^i_{gpp} to P^{i+1}_{gpp} can be represented as in equation (5), where $\Delta hz^{i,i+1}_f$ corresponds to the initial estimation $x^{(0)}$, of the design variable x for the optimization problem (see Figure 4)

$$\Delta P^{i,i+1}_{gpp} = [\Delta hx^{i,i+1}_f \; \Delta hy^{i,i+1}_f \; \Delta hz^{i,i+1}_f \; \Delta E^{i,i+1}_z \; \Delta E^{i,i+1}_y \; \Delta E^{i,i+1}_x] \tag{5}$$

The first two elements $\Delta hx^{i,i+1}_f$ and $\Delta hy^{i,i+1}_f$ in equation (5) are kept constant, whereas the third element $\Delta hz^{i,i+1}_f$ forms part of the objective function as the initial estimation $x^{(0)}$ of the design variable x. This means that objective function converges to an optimal by moving along the Z axis from the initial estimation $\Delta hz^{i,i+1}_f$ *as* $x^{(0)}$ with respect to the 3D-workpiece surface.[13]

For every $\Delta P^{i,i+1}_{gpp}$ and $\Delta hz^{i,i+1}_f$ *as* $x^{(0)}$ that traverse over the CAD-workpiece (see Figure 5(a)), it is necessary to find a "j" number of points defined as DBW^j to calculate the desired optimized surface route. These points DBW^j can be found calculating the Euclidean norm as the difference of

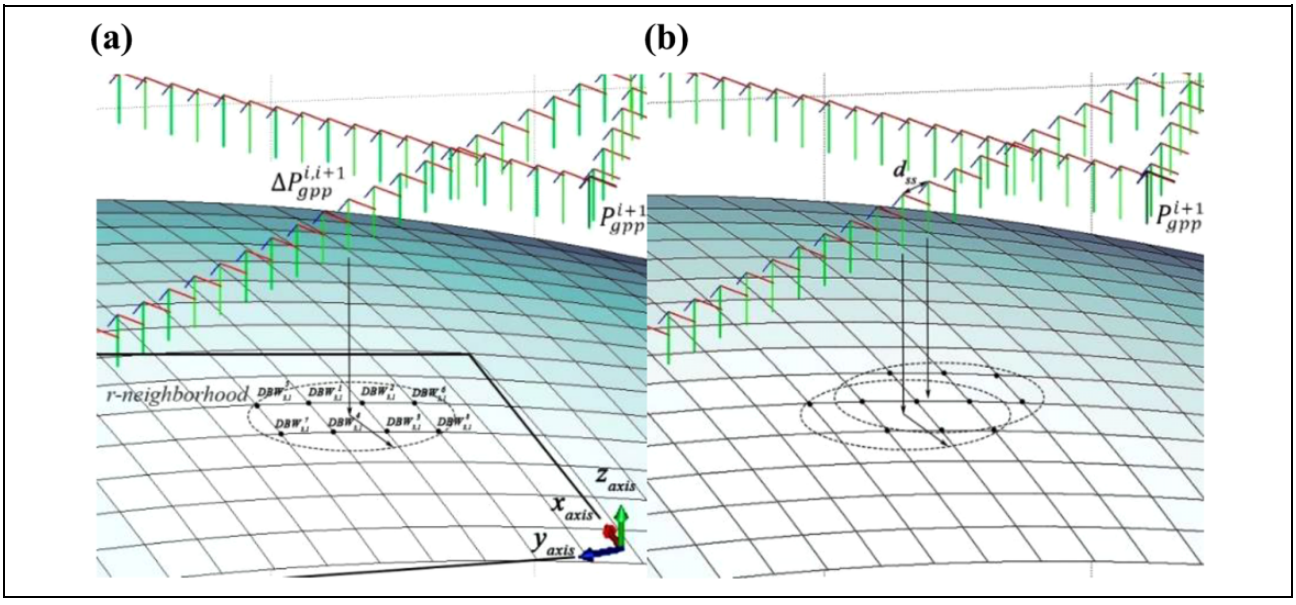

Figure 5. (a) r-neighborhood surrounding $\Delta P_{gpp}^{i,i+1}$. (b) Distancing between linear segments for one straight toolpath.

the xy coordinates of every point of 3D-workpiece with respect to $\Delta hz_f^{i,i+1} as\ x^{(0)}$. A priority queue is ordered using the Bubble sort algorithm and a radius of neighborhood "$r-neighborhood$," where "j" contained in DBW^j is any number from "w" points chosen to optimize each segment of $\Delta P_{gpp}^{i,i+1}$. Since xy coordinates are not modified for every $\Delta P_{gpp}^{i,i+1}$, only $DBW_{3,1}^j$ (equation (6)) is evaluated in the object function

$$DBW^j = [\, hx_w^j \quad hy_w^j \quad hz_w^j \,] \tag{6}$$
$$hz_w^j = DBW_{3,1}^j$$

The distance between every transition d_{ss} is defined as the difference in between $\Delta P_{gpp}^{i,i+1}$ and $\Delta P_{gpp}^{i+1,i+2}$ and depends on the task and the desired accuracy, very small point intervals lead to a time increase in machining processes, whereas longer distances generate a rougher surface[19] (see Figure 5(b)). For continuous sequences of manually recorded poses such as P_{gpp}^i, P_{gpp}^{i+1}, $P_{gpp}^{i+2}, \ldots$ the linear path-planning correlates d_{ss} with the sum of all the distances that conform a continuous path, P_{AB} (see equation (7)), where t_{inc} indicates normalized increments from 0 to 1, the normalized time interval Y represents all possible segment transitions on P_{gpp}^i, P_{gpp}^{i+1}, $P_{gpp}^{i+2}, \ldots$[13]

$$t_{inc} = \frac{d_{ss}}{P_{AB}};\ \mathrm{Y} = [t \in \Re : 0 \le t \le 1] \tag{7}$$

Steepest descent algorithm implementation

The steepest descent (SD) algorithm for unconstrained optimization problems is a popular method for minimizing differentiable functions.[20] For this case, every segment $\Delta P_{gpp}^{i,i+1}$ is individually analyzed using $DBW_{3,1}^j$ and SD algorithm. While $\Delta hx_f^{i,i+1}$ and $\Delta hy_f^{i,i+1}$ from every $\Delta P_{gpp}^{i,i+1}$ remains constant, $\Delta hz_f^{i,i+1}$ moves along to Z-axis to approximate to the surface (see Figure 6). Equation (8) presents the objective function that minimizes the sum of the squared residuals (where a residual is defined as the difference between the chosen points from the 3D workpiece surface $DBW_{3,1}^j$ and the design variable x). The number of terms in the objective function w increases as more elements of $DBW_{3,1}^j$ are selected to reconstruct the surface. Since this is a quadratic function, it is possible to calculate the step size with equation (9)[20]

$$\min f(x) = \sum_{j=1}^{w} (DBW_{3,1}^j - x)^2 \tag{8}$$

$$\lambda^k = \frac{-\left(df\left(x^{(k)}\right)/dx\right)\hat{s}^{(k)}}{\hat{s}^{(k)} d\frac{df\left(x^{(k)}\right)/dx}{dx}\hat{s}^{(k)}} \tag{9}$$

Thus, $\hat{s}^{(k)}$ is obtained from $f(x)$ for the case where λ^k is dynamic as depicted in equation (10), while the approximation of $x^{(k+1)}$ is as in equation (11)

$$\hat{s}^{(k)} = \frac{-\left(df\left(x^{(k)}\right)/dx\right)}{df\left(x^{(k)}\right)/dx} \tag{10}$$

$$x^{(k+1)} = x^{(k)} + \lambda^k \hat{s}^{(k)} \tag{11}$$

A variant of steepest descent algorithm applied to the optimization of manually recorded toolpaths is presented in Algorithm 1.

Algorithm 1. Steepest descent algorithm.

begin
  step 1. Declaration of variables and initialization
    given $x^{(0)}$ (initial parameter), $k\Delta 1$ (current iteration), ε (stop criterion constant) and λ^k (step size);
    evaluate $f\left(x^{(k)}\right)$, $df\left(x^{(k)}\right)/dx$, λ^k *and* $\hat{s}^{(k)}$;
  step 2. Main loop
    repeat
      evaluate $\hat{s}^{(k)}$ *and* λ^k;
      evaluate $x^{(k+1)}$, $f\left(x^{(k+1)}\right)$, $df\left(x^{(k)}\right)/dx$;
    until $df\left(x^{(k)}\right)/dx < \varepsilon$;
end

The stopping criterion for algorithm 1 is defined as $df\left(x^{(k)}\right)/dx < \varepsilon$, where ε is a constant that compares the change rate with respect to the objective function. Once every segment $\Delta P_{gpp}^{i,i+1}$ from every straight toolpath that has been optimized, the desired optimized toolpaths can be smoothed to avoid any possible discontinuity[21] (see Figure 7(a) and (b)). Finally, the desired optimized toolpaths can be displaced along the Z axis of its corresponding coordinate frame, with a standoff distance between the 3D-workpiece surface and the desired TCP (see Figure 7(c)).

Algorithm 2 shows the complete procedure used to optimize trajectories in the industrial robot for both the simulation and experimentation implemented in the following section.

Algorithm 2. Procedure to optimize the trajectory.

begin
  initialize $P_{gpp}^{1,\ldots,n}$;
  for $(i = 1$ *to* $n-1)$ **do**
    select a pair of P_{gpp}^{i} and P_{gpp}^{i+1} and **calculate** t_{inc};
    for $(t = 0$ *to* $1,\ t = t + t_{inc})$ **do**
      $h = t/T$;
      consider $Pos_1 = P_{gpp}^{i}$ and $Pos_2 = P_{gpp}^{i+1}$ using

$$Pos_1 = \begin{bmatrix} nAx & sAx & aAx & pAx \\ nAy & sAy & aAy & pAy \\ nAz & sAz & aAz & pAz \\ 0 & 0 & 0 & 1 \end{bmatrix},\ Pos_2 = \begin{bmatrix} nBx & sBx & aBx & pBx \\ nBy & sBy & aBy & pBy \\ nBz & sBz & aBz & pBz \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

      obtain $\sigma = Pos_1 * D(h)$, using Pos_1, Pos_2 and $D(h) = T(h) * Ra(h) * Ro(h)$;[17]
      declare the transition $\Delta P_{gpp}^{i,i+1} = [\ \Delta hx_f^{i,i+1}\quad \Delta hy_f^{i,i+1}\quad \Delta hz_f^{i,i+1}\quad \Delta E_z^{i,i+1}\quad \Delta E_y^{i,i+1}\quad \Delta E_x^{i,i+1}\ ]$, where $\Delta hx_f^{i,i+1} = \sigma_{1,4}$, $\Delta hy_f^{i,i+1} = \sigma_{2,4}$ and $\Delta hz_f^{i,i+1} = \sigma_{3,4}$ correspond to the position obtained from $D(h)$ using $\Delta E_z^{i,i+1}$, $\Delta E_y^{i,i+1}$ and $\Delta E_x^{i,i+1}$;
      optimize the trajectories selecting the nearest DBW^{Ω} as follows:
      find the number of points Ω that conforms to the point cloud data
      for $(\mu = 1$ *to* $\Omega,\ \mu = \mu + 1)$ **do**
        find the Euclidian Norm λ of DBW^{Ω} and $\Delta P_{gpp}^{i,i+1}$:

$$\lambda = \sqrt{(x_f^* - x_w^{\Omega})^2 + y_f^* - y_w^{\Omega 2}}$$

        if $\lambda < \xi$ **then**
          add the node DBW^{Ω} and its corresponding λ to DB;
        end
      end
      apply Bubble Sort algorithm to DB and select the three closest points with respect to $\Delta P_{gpp}^{i,i+1}$, naming them as DBW^{j}, with $\Gamma = [j \in \mathbb{N},\ 1 \le j \le w]$;
      apply the steepest descent algorithm;
      add the standoff distance to z_w^{opt}, that is the solution of the corresponding optimization problem, from $\phi = [\ x_w^{opt}\quad y_w^{opt}\quad z_w^{opt}\quad \Delta E_z^{i,i+1}\quad \Delta E_y^{i,i+1}\quad \Delta E_x^{i,i+1}\ ]$;
      store ϕ into the database DB_2;
    end
  end
  execute the smoothing process, using DB_2 as input, based on a penalized least squares method, DCT, and IDCT;
  execute the program corresponding to the robot kinematics (simulation on Matlab®);
  execute the post-processor (for ABB® systems) /* conversion of Matlab® code into RAPID for experimentation
end

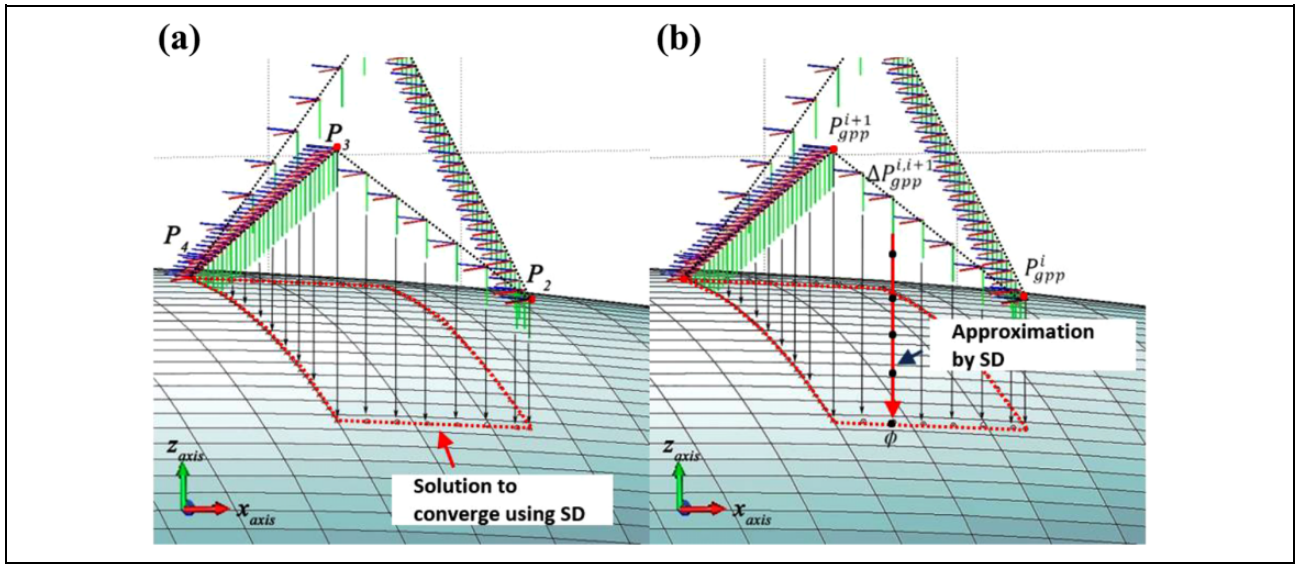

Figure 6. (a) Optimization of every recorded toolpath with respect to the z-axis coordinate frame. (b) Approximation of the 3D-workpiece surface using SD.

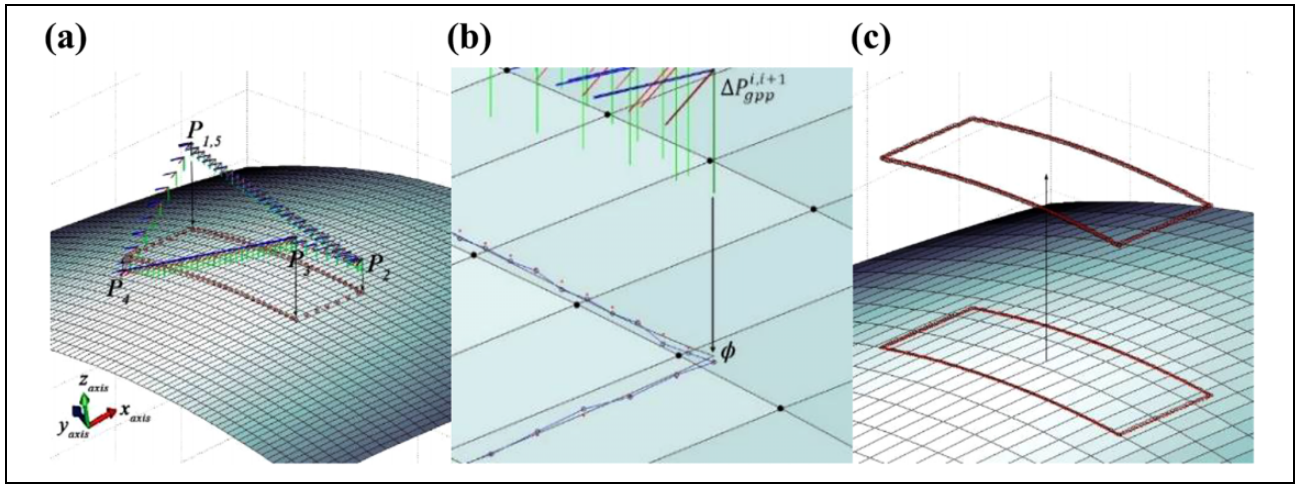

Figure 7. (a) Toolpath traced over the curved surface, where $\boldsymbol{P}_1, \ldots, \boldsymbol{P}_5$ represent the initial poses recorded by the programmer. The dotted path corresponds to the optimized trajectory over the surface, (b) implementation of additional smoothness, (c) standoff distance applied to the desired optimized trajectory.

Experimentation

The experimentation was carried out using an ABB industrial robot IRB 1600-7/1.45 type A, (Figure 8) and a program-like based on algorithm 2 that converts a set of instructions in P_{gpp} format into RAPID language (the language to program ABB robots). The performance of algorithm 2 was evaluated using three baseline trajectories over the 3D-workpiece surface. The first trajectory represents a closed trajectory with sharp corners, the second a zig-zag trajectory with sharp corners, and the third a closed trajectory with rounded corners [13]. Robot TCP motion was recorded and compared with desired trajectories to measure mean absolute error (MAE).

Trajectory 1 presents variable standoff distance in the Z axis simulating manual induced error during point-to-point programming or lead-through method (see Figure 9(a)). In this example, the trajectory finalizes its trajectory in the same pose as the beginning and it is composed of five poses as shown in Figures 3 and 9(a).

Trajectory 2 depicts a zigzag programmed in 2D (with components only in xy coordinates), where the trajectory has repetitive sharp corners (see Figure 9(b)). In this example, 14 poses were declared to create trajectory 2, wherein the start and end form an open trajectory.

Trajectory 3 presents a closed trajectory without sharp corners. The main purpose of this trajectory is to move the

TCP as close as possible from one corner to another without stopping. For this case, 41 poses characterize the trajectory, as presented in Figure 9(c).

The 3D-workpiece surface from Trajectories 1–3 (see Figure 9(a) to (c)) is known and was selected to validate algorithm 2 using an industrial robot. Equation (12) represents the 3D-workpiece surface with defined intervals of $0.6 \leq hx_w \leq 0.9$ and $-0.4 \leq hy_w \leq 0.4$, $hz_w = f(hx_w, hy_w)$, where hx_w, hy_w, hz_w can be any coordinate vector in the Euclidean space. The surface was represented as a point cloud, with three different densities 656, 2511, and 9821 points, corresponding to separations of $\phi = 0.02$, 0.01, and 0.005 m, no additional information from the 3D-workpiece was required

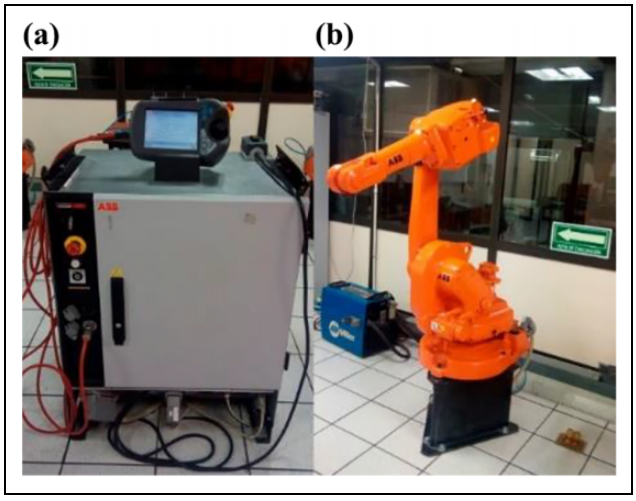

Figure 8. (a) ABB IRC5 controller. (b) Robot Arm ABB IRB 1600-7/1.45 type A.

$$f(hx_w, hy_w) = 0.5 + 1.8^{-1.5\left(\sin(hx_w)cos(hy_w/2)\right)\sqrt{hx_w^2+hy_w^2}} \tag{12}$$

Algorithm 2 was computed in MATLAB® with a graphic user interface that includes the 3D-workpiece surface, robot, toolpath, and an additional window that allows selecting a file with P_{gpp} poses (see Figure 10). The application simulates the robot motion performing the optimized desired trajectory before its practical implementation. At the end of the simulation, the application translates automatically the P_{gpp} poses into RAPID code (the native language from ABB robots). The resulted file was loaded into the robot controller using a USB drive. The rapid code

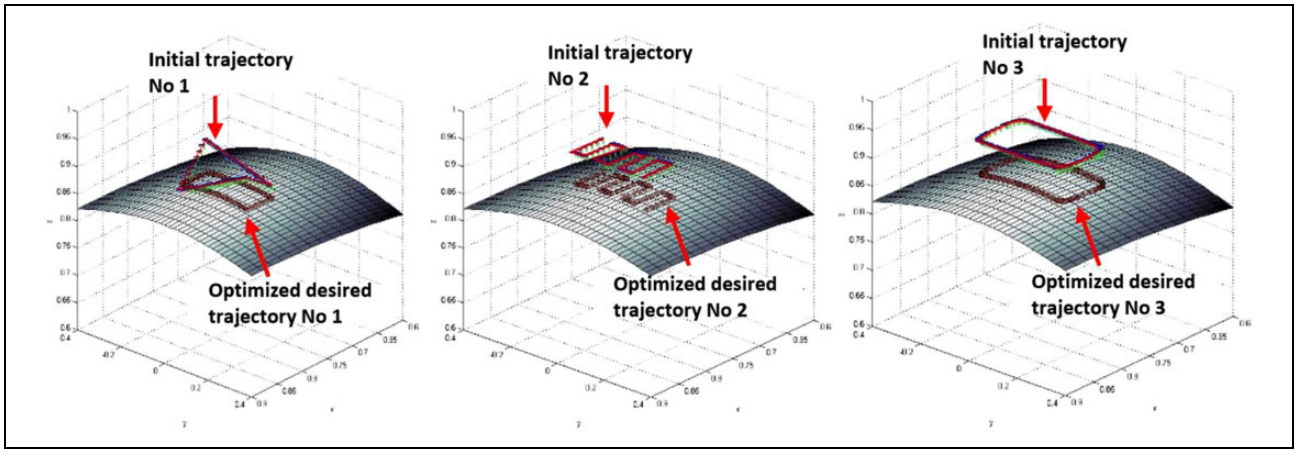

Figure 9. Manually recorded toolpaths and their corresponding desired optimized trajectories over the surface.

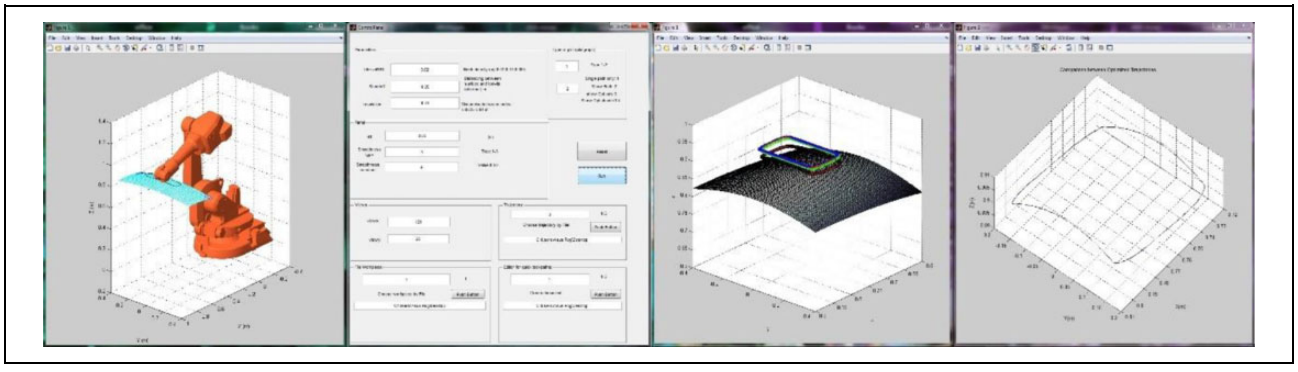

Figure 10. Graphic user interface application generated in MATLAB® to simulate the performance of an ABB IRB 1600/7 1.45 type A robot.

created in MATLAB also includes additional instructions to store the real TCP location from the robot, via serial communication in real time using a cable that connects the controller with a PC.

Results

The three presented trajectories in the previous chapter were simulated and evaluated using the ABB Industrial robot IRB 1600-7/1.45 type A (Figure 8**)**. Other configuration parameters are w for $w = 3$ and the distancing defined as d_{ss} for $d_{ss} = 0.0025$ m. Table 1 includes the number of poses for each trajectory before and after the implementation. This information serves to calculate the robot programming lines required. The number of segments generated in Table 1 depends mainly on d_{ss}.

Table 1. Number of points manually and automatically recorded.

Trajectory	No. of poses manually recorded	No. of segments recorded automatically
1	3	307
2	14	273
3	41	353

Table 2. Standoff distance error.

	Trajectory 1		
	$\phi = 0.02\ m$	$\phi = 0.01\ m$	$\phi = 0.005\ m$
Best (mm)	0.004076	0.034669	0.00014
Worst (mm)	0.434296	0.395630	0.374642
Average (mm)	0.26759	0.199623	0.103027
SD σ (mm)	0.08058	0.073679	0.06253
	Trajectory 2		
Best (mm)	0.0000907	0.001042	0.002324
Worst (mm)	0.483938	0.292049	0.314384
Average (mm)	0.184529	0.113632	0.080471
SD σ (mm)	0.108992	0.072224	0.047713
	Trajectory 3		
Best (mm)	0.01401	0.000648	0.000182
Worst (mm)	0.764634	0.381466	0.20452
Average (mm)	0.432307	0.190036	0.091117
SD σ (mm)	0.222385	0.115686	0.052644

The mean absolute error (MAE) was measured from the difference between the desired trajectory route based on equation (12) and the TCP final measurement from the robot (reading the angular motion from encoders/resolvers and the length of every link). Table 2 shows the MAE obtained from the robot varying the mesh density (ϕ), for $\phi = 0.02$m, $0, .01$m, and $0, .005$m. Table 2 shows the MAE when $\phi = 0.005$ m, taking into consideration the individual average error for every reference coordinate in a trajectory and the total error as the sum of them.

The results obtained from Table 2 demonstrate that the mesh density (ϕ) impacts directly on the reduction of the average error produced for the three trajectories from an average error of 0.4 mm to 0.1 mm. Likewise, Table 3 shows the MAE obtained from the same trajectories sorting the MAE by coordinates. The methodology presents comparable results with other methodologies taking into consideration the proposed mesh density[7,13] The mesh density ϕ and distance between every segment transition d_{ss} directly increases or decreases the accuracy of the optimized trajectory. The error produced in xy coordinates was at some point expected and might be attributed to external factors from the proposed methodology such as robot serial configuration, low structural rigidity, and gear backlash[22]

Conclusions and future work

In this article, a novel algorithm implementation combines the manually recorded toolpaths stored in a text file with the use of a CAD-workpiece model to reduce manual error induced and optimize the final robotic toolpath. The steepest descent algorithm finds the surface route wherein the manually recorded toolpaths traverse over the CAD-workpiece.

The results shown in the previous chapter demonstrates that the proposed methodology can reduce the manual error induced using the CAD-workpiece as a reference. The results obtained for the three different trajectories

Table 3. MAE corresponding to every coordinate conforming to the real TCP.

	Trajectory 1			Trajectory 3	
Coordinate	MAE (mm)	Average (mm)	Coordinate	MAE (mm)	Average (mm)
X	0.001705	0.112393	X	0.001931	0.101113
Y	0.007660		Y	0.008065	
Z	0.103027		Z	0.091117	
Coordinate	Trajectory 2				
X	0.005065	0.089929			
Y	0.004393				
Z	0.080471				

evaluated indicate that the accuracy of the trajectory depends directly on the mesh-distancing ϕ, the distance between every segment transition $d_{ss} = 0.0025$ m, and external factors such as robot serial configuration, low structural rigidity, and gear backlash. The quality of the 3D-workpiece surface reconstruction also depends on the objective function. As for now, the implemented methodology only utilized three points from the 3D-workpiece surface for every $\Delta P_{gpp}^{i,i+1}$ and $\Delta hz_f^{i,i+1}$. As future work, we plan to explore using a bigger number of points DBW^j for $w > 3$ to obtain a better approximation of complex 3D-workpiece surfaces for the use in robotic automated fiber placement applications.

Acknowledgements

The authors would like to offer our special thanks to Professor Albert J Shih for all the support provided during the elaboration of this article, in the same way, the authors would like to thank Arturo Gil for the ARTE library. All authors also thank CONACyT of México.

Declaration of conflicting interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

Funding

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: This work was supported by SIP Projects 20211583 and 20210825 at the Instituto Politécnico Nacional of México and the University of Michigan.

ORCID iDs

Miguel Angel Funes-Lora https://orcid.org/0000-0002-8259-4396
Eduardo Vega-Alvarado https://orcid.org/0000-0001-9464-7996
Raúl Rivera-Blas https://orcid.org/0000-0001-6368-7805

Supplemental material

Supplemental material for this article is available online.

References

1. Chen H, Xi N, Sheng W, et al. A general framework for automatic CAD-guided tool planning for surface manufacturing. In: *Proceedings—IEEE international conference on robotics and automation*. 2003. Epub ahead of print 2003. DOI: 10.1109/robot.2003.1242132.
2. Chen H, Thomas F and Xiongzi L. Automated industrial robot path planning for spray painting a process: a review. In: *4th IEEE conference on automation science and engineering, CASE 2008*. 2008. Epub ahead of print 2008. DOI: 10.1109/COASE.2008.4626515.
3. Andulkar MV and Chiddarwar SS. Incremental approach for trajectory generation of spray painting robot. *Ind Rob* 42. Epub ahead of print 2015. DOI: 10.1108/IR-10-2014-0405.
4. Milfelner M, Kopac J, Cus F, et al. Intelligent system for machining and optimization of 3D sculptured surfaces with ball-end milling. *J Achiev Mater Manuf Eng* 2006; 14.
5. Zhou P, Zhao X, Tao B, et al. Time-varying isobaric surface reconstruction and path planning for robotic grinding of weak-stiffness workpieces. *Robot Comput Integr Manuf* 64. Epub ahead of print 2020. DOI: 10.1016/j.rcim.2020.101945.
6. Li Y, Chen T and Liu D. Path planning for laser cladding robot on artificial joint surface based on topology reconstruction. *Algorithms* 13. Epub ahead of print 2020. DOI: 10.3390/A13040093.
7. Mineo C, Pierce SG, Nicholson PI, et al. Robotic path planning for non-destructive testing—a custom MATLAB toolbox approach. *Robot Comput Integr Manuf* 37. Epub ahead of print 2016. DOI: 10.1016/j.rcim.2015.05.003.
8. Mineo C, Pierce SG, Nicholson PI, et al. Introducing a novel mesh following technique for approximation-free robotic tool path trajectories. *J Comput Des Eng* 4. Epub ahead of print 2017. DOI: 10.1016/j.jcde.2017.01.002.
9. Denkena B, Schmidt C and Weber P. Automated fiber placement head for manufacturing of innovative aerospace stiffening structures. *Procedia Manuf* 2016; 6: 96–104.
10. Jeffries KA. Enhanced robotic automated fiber placement with accurate robot technology and modular fiber placement head. *SAE Int J Aerosp* 6. Epub ahead of print 2013. DOI: 10.4271/2013-01-2290.
11. Hong L, Wang B, Yang XL, et al. Offline programming method and implementation of industrial robot grinding based on VTK. *Ind Rob* 47. Epub ahead of print 2020. DOI: 10.1108/IR-04-2019-0093.
12. Bedaka AK and Lin CY. CAD-based robot path planning and simulation using OPEN CASCADE. In: *Procedia Computer Science*. 2018. Epub ahead of print 2018. DOI: 10.1016/j.procs.2018.07.119.
13. Funes-Lora MA, Portilla-Flores EA, Vega-Alvarado E, et al. A novel mesh following technique based on a non-approximant surface reconstruction for industrial robotic path generation. *IEEE Access* 7. Epub ahead of print 2019. DOI: 10.1109/ACCESS.2019.2897079.
14. Xingguo Y and Li T. Virtual engineering: model based off-line programming method for industrial robot. In: *2008 IEEE international conference on robotics, automation and mechatronics, RAM 2008*. 2008. Epub ahead of print 2008. DOI: 10.1109/RAMECH.2008.4690872.
15. González-Galván EJ, Loredo-Flores A, Pazos-Flores F, et al. An optimal path-tracking algorithm for unstructured environments based on uncalibrated vision. In: *Proceedings—IEEE international conference on robotics and automation*. 2005. Epub ahead of print 2005. DOI: 10.1109/ROBOT.2005.1570496.
16. Shen F and Tarbutton J. A voxel based automatic tool path planning approach using scanned data as the stock. *Procedia Manuf* 2019; 34: 26–32.

17. Paul R. Manipulator Cartesian path control. *IEEE Trans Syst Man Cybern* 9. Epub ahead of print 1979. DOI: 10.1109/TSMC.1979.4310109.
18. Bekey G. Springer Handbook of robotics (B. Siciliano and O. Khatib; 2008) Book Review. *IEEE Robot Autom Mag* 15. Epub ahead of print 2008. DOI: 10.1109/mra.2008.928399.
19. Rong-Shine Lin YK. Efficient tool-path planning for machining free-form surfaces. *J Eng Ind* 1996; 118: 20–28.
20. Himmelblau DM. *Applied nonlinear programming*. New York: McGraw-Hill. 1972.
21. Garcia D. Robust smoothing of gridded data in one and higher dimensions with missing values. *Comput Stat Data Anal* 2010; 54: 1167–1178.
22. Denkena B. New production technologies in aerospace industry: *Proceedings of the 4th machining innovations conference*, Hannover, September 2013. Springer International Publishing. https://books.google.com/books?id=RaDjAAAAQBAJ (2013)