JTTEE 3:208-215

Computer-Aided Torch Trajectory Generation for Automated Coating of Parts with Complex Surfaces

T. Vivekanandhan, A.R. Kashani, and R. Echempati

The work presented in this paper is concerned with the design and development of an efficient computer software package for off-line programming of robots for spray coating. The developed package (called CA-COATS) allows the spray torch to track a path, maintaining the specified spraying distance and proper orientation relative to the part surface at all points, and thus ensuring uniform coating thickness. This software package is done off-line and is complete in itself, eliminating the need for another computer-aided design database of the base surface (although the package can be made to interface with such a database if so desired). Because the algorithm has no iteration, it is computationally efficient. With minor modifications, this package can also be extended to waterjet coating-removal systems.

1. Introduction

1.1 *Coating Robots*

INDUSTRIAL robots are programmed to perform specific tasks using general-purpose programming languages. Extensive research is being done to add intelligence to robots, thereby making them useful for particular applications in an interactive manner. This paper discusses robots suitable for thermal spray applications and related topics.

Spraying is one of the few processes that make effective use of the flexibility offered by industrial robots. Coating surfaces normally have three-dimensional (3D), varying geometries, which makes the programming of the torch motion over and around the surface an important task. Proper torch trajectory for a coating process requires that:

- The torch be held at a particular offset distance from the surface while tracking the surface
- The torch be held at a particular orientation (frequently normal) to the surface at each point while tracking the surface
- The torch track the surface at a particular speed without stopping over the surface

A coating robot should have efficient microprocessor control for the robot motions, expanded disk memory, a developed flex-arm, and an easy programming mode. For general coating applications, open-ended Cartesian robots with six or seven axes are used. For coating of closed-surface components, special robots are designed to give some degrees of freedom of motion (one to four) to the part so that it can move during the coating process. This reduces extreme link motion and maintains the required orientations at the end effector. Gantry-type robots are used for massive symmetrical components.

Key Words: automation, manufacturing, robotic control, spraying complex surfaces, computer-aided geometric design

T. Vivekanandhan, Applied Automation Technologies, Inc., Sterling Heights, MI 48314, USA; **A.R. Kashani** and **R. Echempati,** Mechanical Engineering and Engineering Mechanics Department, Michigan Technological University, Houghton, MI 49931, USA

Hydraulic robots were introduced in painting lines during the early 1970s. DeVilbiss Inc., a leading surface-coating equipment company, in collaboration with Trallfa Robotics, Inc. (Norway), developed an advanced hydraulic-drive spray-painting robot (Ref 1). In the past, robots for painting were hydraulically operated because of the problems of using electricity in hazardous environments. The benefits of electric-drive robots were excluded from spraying applications until 1985, when GM Fanuc Robotics Company introduced an electric painting robot (Ref 1). Graco Robotics demonstrated the EBX-1 electric painter in 1987 (Ref 1).

Electric robots are now used in spray booths and are widely accepted because of their smooth motion, low power consumption, and reliability. They are also valued for their speed, accuracy, and easy maintenance. AKR Robotics, Inc. has developed a six-axis electrohydraulic manipulator for coating applications (Ref 2). Spline Robotics AB (Sweden) has developed a spline spraying system for coating of parts that are difficult to access. The design of the Spline robot gives the arm an extreme accessibility and long reach from a very compact base. Several other commercial coating robots are discussed in the literature.

Coating applications generally deal with surfaces; therefore, the robot should scan the entire surface by tracking individual paths that are incremented at the end of each path. During this tracking of individual paths, the end effector should be maintained at a given offset distance with proper orientation relative to the surface. Three general methods are used to program coating robots: assisted on-line programming, vision-assisted programming, and off-line programming.

1.2 *Robot Programming*

Almost all present-day coating robots are programmed using the so-called lead-through method, also known as the power teach-in method (Ref 3). In this method the operator leads the arm through specified locations, while the robot controller stores the locations for playback. Another method of assisted on-line programming involves moving the end effector to desired corner locations using a teach pendant, and then allowing the robot controller to develop interpolated straight-line paths between the corner points during playback.

In the vision-assisted programming method, the part surface is learned using a vision system utilizing the recognized part surface, while a path-planning algorithm is executed for generating path locations (Ref 3). Economical optical sensors are also used to learn the surface iteratively to determine the torch path for coating.

The off-line programming method is used to develop spray path locations on a computer while the robot is still in production on another job (Ref 3). This method generates a spray path for the computer-aided design (CAD) data of the part geometry. Because the program is generated off-line in a computer, simulation of the process is possible. This method also reduces robot downtime.

A few off-line software packages are commercially available with capabilities for spray painting. For example, ROBCAD (Technomatrix Technologies) is a complete off-line package suitable for most robots and has add-on modules for spray-coating applications. The paint package automatically generates the path of the painting gun for a well-structured CAD data file of the part surface geometry. Because ROBCAD is equipped with robot libraries and path generation/simulation modules, it also allows control of gun velocity, orientation, and standoff distance. Due to its intensive solid model animation, it is implemented on a Silicon Graphics (Silicon Press, San Diego, CA) workstation. The IGRIP program (Deneb Robotics Inc., Auburn Hills, MI) uses a CAD source of the surface geometry to generate spray paths. The spray paths generated maintain offset distance and orientation relative to the surface. This software also has different off-the-shelf robot libraries stored for reference and simulation, and requires a Silicon Graphics workstation.

The work presented in this paper is based on the development of an efficient and economical off-line package for applications such as spray painting and plasma spraying. This method and the developed algorithm allow the spray torch to track a path, maintaining the specified spraying distance and orientation relative to the surface at all points, and thus ensuring uniform coating thickness.

The off-line programming method developed to satisfy the above requirements is explained in two phases. In the first phase, position and orientation information for the torch trajectory is generated. In the second phase, transformation of the data to a robot environment and development of a postprocessor to generate the required robot program are undertaken.

2. Phase 1: Torch Trajectory Generation

This section presents the mathematical analysis for generation of the base and the offset surfaces of the coating torch.

2.1 *Generation of the Base Surface*

For the generation of position and orientation information, the base surface should be completely defined. Considering the complexity demanded by engineered surfaces, a periodic bicubic B-spline surface (Ref 4, 5) has been selected for this work:

$$Q_{ij}(u, v) = [u][B][V_{i,j}][B]^T[v]^T \quad \text{(Eq 1)}$$

where $Q_{i,j}(u, v)$ is the (i, j)th bicubic B-spline surface patch; u and v are the parametric variables, which represent the corner points of the patch; $V_{i,j}$ are the control vertices of the patch; B is the transformation matrix for B-spline curves; and $[\ ]^T$ represents the transpose of the respective matrices. These individual quantities can be represented in matrix form as (Ref 4, 5):

$$[u] = [u^3\ u^2\ u1] \quad \text{(Eq 2)}$$

$$[B] = \tfrac{1}{6}\begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{bmatrix} \quad \text{(Eq 3)}$$

$$[V_{i,j}] = \begin{bmatrix} V_{1,1} & \cdots & V_{1,4} \\ \cdots & \cdots & \cdots \\ V_{4,1} & \cdots & V_{4,4} \end{bmatrix} \quad \text{(Eq 4)}$$

$$[v] = [v^3\ v^2\ v\ 1] \quad \text{(Eq 5)}$$

A digitizer is used to sample a limited number of points on the surface to be coated (referred to here as the "coating surface") that highlight the surface profile. Using this collected information, an interpolation method generates a surface through these points. Coordinate measuring digitizers are typically of the probe type, the tip of which is moved to a point on the surface; then the 3D data collect key (usually a foot pedal) is activated (Fig. 1). The digitizer sends out the x, y, z coordinates of the digitized point with respect to the digitizer coordinate system. The

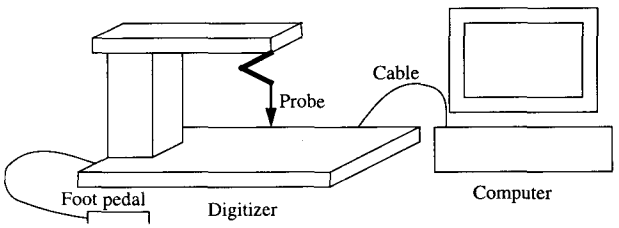

Fig. 1 Schematic of a digitizer interfaced with a computer

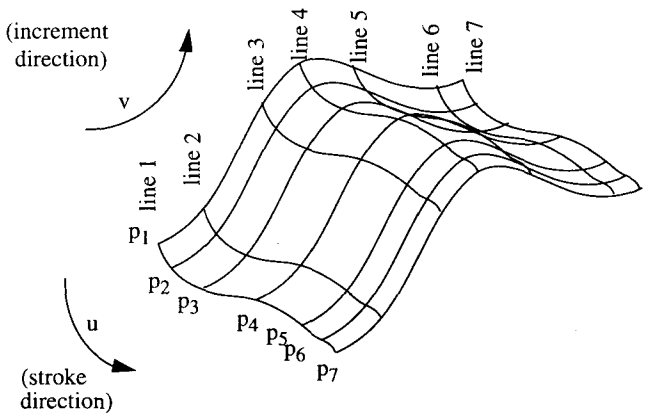

Fig. 2 Surface division for digitization

digitized data are interactively received by a suitable interfaced computer program, which stores the data in a file. Because the surface data files are stored in arrays and later used as inputs for a surface-generation algorithm, the digitization process must be performed in a particular sequence. The algorithm for surface generation uses fitting of quadrilateral surface patches, so the coating surface is first divided into rectangular patches by drawing lines in both parametric directions, u, and v (i.e., u-direction lines and v-direction lines) (see Fig. 2).

Figure 2 also shows the stroke direction, which is the direction in which the spray torch scans the surface, and the increment direction for incrementing the spray torch at the end of each stroke. During digitization, the u direction is the stroke direction and the v direction is the increment direction. For coating applications, the less complex direction usually is chosen for the stroke direction. The file containing the digitized points is used as the input to the surface-generation algorithm, and the digitized points are the interpolation points for the given sampled surface.

2.2 *Surface Interpolation Algorithm*

This section discusses the algorithm developed to generate a bicubic B-spline surface interpolating the digitized points. The first phase of generating a surface is to determine a set of control points, which when input in the B-spline definition will generate a description of a surface that interpolates the given network of points. Here, the B-spline control vertices are the unknowns to be determined and each digitized point to be interpolated is a condition to be satisfied. The complex surface is considered in a piecewise manner. The Barsky and Greenberg (Ref 6) method is used to generate the interior and boundary control points, and the extended boundary control points are determined through linear extrapolation. These extended boundary control points are required to define the circumferential boundary patches of the surface. A control-points module is written to generate the control vertices using the above method. This program reads the digitized points array as input and generates all the required control vertices—$V_{i,j}(x)$, $V_{i,j}(y)$, and $V_{i,j}(z)$—which are then stored in the control-points array.

Once an appropriate set of control vertices has been determined, a surface can be constructed that interpolates the network of points. By varying the parameter values (u and v) in the defining equation, any number of mesh points can be generated in each patch. To define the mesh spacing for coating applications, the term "spray pass increment" has been used, which is the distance the torch must move after every pass. This depends on the torch offset distance, the spray pattern, and the overlap between passes required to obtain a uniform coating (Fig. 3). Usually, the spray pass increment distance is predetermined and fixed for a particular coating setting. This determines the mesh spacing in the increment direction (see Fig. 2). The mesh spacing in the stroke direction should be varied according to the complexity of the surface; therefore, the program also allows different spacing in the u and v directions (Fig. 4). Spray pass increment generally is considered as a reference for the mesh spacing in both parametric directions (u and v). For this reason, even at the stage of digitization, care should be taken to maintain an integer number of spray pass increments between subsequent digitized points to ensure accurate mesh spacing during surface generation.

The number of intervals in the u and v directions are calculated in each patch using the mesh spacing distance and the digitized end points of the current patch (see Eq 2). For this patch, the surface points are calculated starting from the corner point ($u = v = 0$) and incrementing the interval distance in the v direction first until $v = 1$. This is repeated for each u interval increment until $u = 1$. This method allows a constant mesh spacing to be maintained throughout the surface in all the patches.

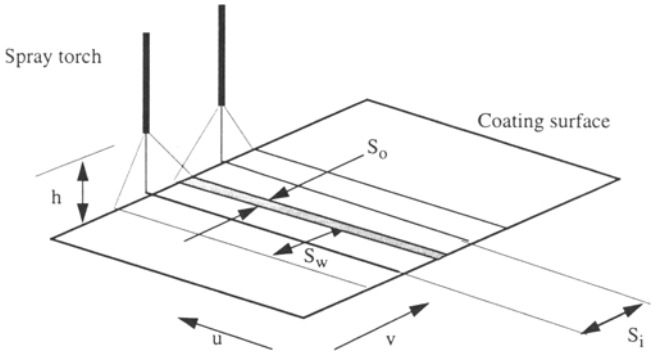

Fig. 3 Spray pass increment (S_i). S_w, spray pattern width; S_o, spray pattern overlap; h, standoff distance

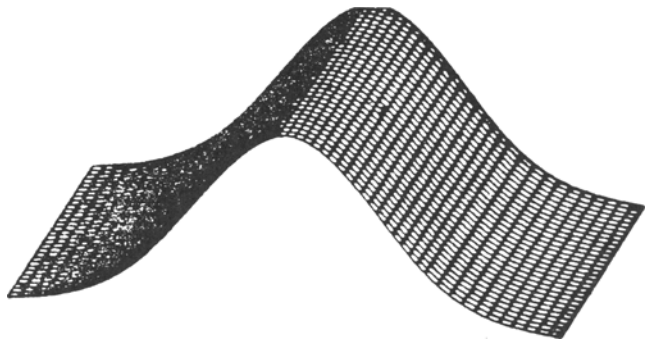

Fig. 4 Surface mesh in the u and v directions

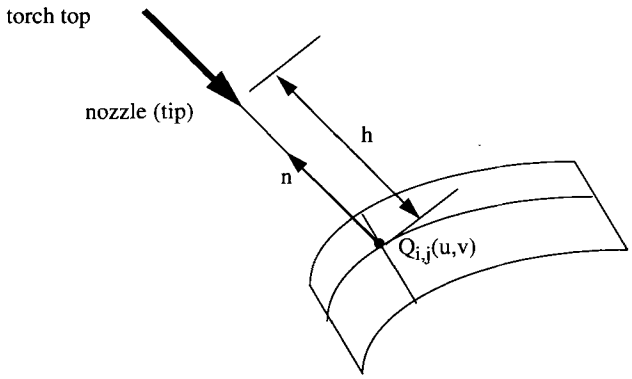

Fig. 5 Standoff distance (h) and orientation. **n**, unit normal vector for mesh point; $Q_{i,j}(u, v)$, coating surface mesh point

2.3 *Generation of the Offset Surface*

During coating, the torch is held at a certain standoff distance relative to the part surface. Generally, the torch is held normal relative to the surface point being sprayed, and thus the standoff distance is measured in the normal direction. For oblique orientation, the x, y, and z angles with respect to the normal should be specified. Consider a single coating-surface mesh point $Q_{i,j}(u, v)$ in a patch (i, j) with specified u and v parameter values, as shown in Fig. 5. Generation of the offset position coordinates involves the following steps:

- Determining the tangent vectors in the u and v directions
- Determining the unit normal to the surface at that point
- Rotating the unit normal vector in sequence with respect to the x, y, and z axes, by the given torch orientation angles. This rotated unit normal vector is called a unit orientation vector.
- Multiplying this final unit orientation vector with the standoff distance to determine the location of the nozzle tip
- Multiplying the unit orientation vector by the sum of the standoff distance and torch length to determine the position of the other end of the torch, to calculate the global orientation, and for simulation purposes

The partial derivatives of Eq 1 with respect to the u and v variables are

$$\frac{\partial}{\partial u}\{Q_{i,j}(u, v)\} = [\mathbf{u}'][B][V_{i,j}][B]^{\mathrm{T}}[v]^{\mathrm{T}} = [x'_u \; y'_u \; z'_u] \quad \text{(Eq 6)}$$

$$\frac{\partial}{\partial v}\{Q_{i,j}(u, v)\} = [\mathbf{u}][B][V_{i,j}][B]^{\mathrm{T}}[v']^{\mathrm{T}} = [x'_v \; y'_v \; z'_v] \quad \text{(Eq 7)}$$

where $[\mathbf{u}']$ is obtained by differentiating Eq 2 with respect to u, and $[v']^{\mathrm{T}}$ is obtained by differentiating the transpose of Eq 5 with respect to v. A single prime over a quantity represents the first derivative.

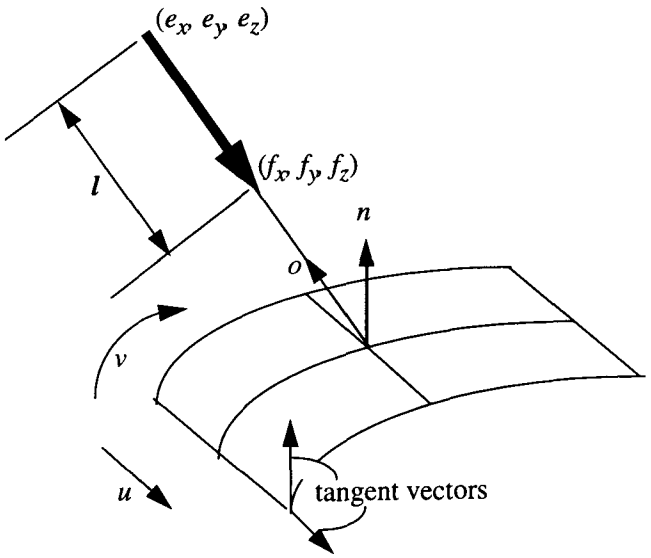

Fig. 6 Unit normal vector (**n**) and unit orientation vector (**o**). l, torch length

Equations 6 and 7 define the tangent vectors in the u and v directions, as shown in Fig. 6. The normal to the surface, **N**, is then calculated by taking the cross product of these two tangent vectors as:

$$\mathbf{N} = \frac{\partial}{\partial u}\{Q_{i,j}(u, v)\} \times \frac{\partial}{\partial v}\{Q_{i,j}(u, v)\} \quad \text{(Eq 8)}$$

$$= [x'_u \; y'_u \; z'_u] \times [x'_v \; y'_v \; z'_v] \quad \text{(Eq 9)}$$

The three coordinates of the normal vector are

$$\mathbf{N}_x = (y'_u \; z'_v) - (z'_u \; y'_v) \quad \text{(Eq 10)}$$

$$\mathbf{N}_y = (z'_u \; x'_v) - (x'_u \; z'_v) \quad \text{(Eq 11)}$$

$$\mathbf{N}_z = (x'_u \; y'_v) - (y'_u \; x'_v) \quad \text{(Eq 12)}$$

The magnitude of the normal vector is now given by:

$$|\mathbf{N}| = \sqrt{\mathbf{N}_x^2 + \mathbf{N}_y^2 + \mathbf{N}_z^2} \quad \text{(Eq 13)}$$

Thus, the x, y, and z components of the unit normal vector **n** are

$$\mathbf{n}_x = \frac{\mathbf{N}_x}{|\mathbf{N}|}, \; \mathbf{n}_y = \frac{\mathbf{N}_y}{|\mathbf{N}|}, \text{ and } \mathbf{n}_z = \frac{\mathbf{N}_z}{|\mathbf{N}|} \quad \text{(Eq 14)}$$

One of the main advantages of using the B-spline surface representation for coating surfaces is that it is easily programmable on a digital computer. Another advantage is its smooth second-order continuity property. This results in a unique normal vector for every point, which is very important in this application (Ref 7). Next, the orientation vector, defined as the vector along the axis of the torch, is specified by the user by defining the three angles xa, ya, and za. The unit orientation vector is obtained by rotating the unit normal vector first about the x axis through the angle xa, then about the y axis through the angle ya, and finally about the z axis through the angle za. The angles xa, ya, and za must be defined considering this order of rotation. For normal orientation, $xa = ya = za = 0$.

Three-dimensional rotational transformation matrices in the x, y, and z directions are then performed to rotate the unit normal vector to obtain the final unit orientation vector [**o**] (Fig. 6). Figure 6 also shows the position vectors of the tip of the nozzle, **f**(u, v), and the top (extended offset point) of the nozzle, **e**(u, v), at a given standoff distance (Fig. 5) in the global coordinate system.

A computer module generates and stores the data in arrays of the 3D position points for the tip and the top of the torch. These data are stored as a collection of 3D points to be used in torch path planning. Three arrays are used for torch trajectory planning: the coating surface mesh array, the offset surface mesh array, and the extended offset surface mesh array. All three arrays have the same number of elements. The requirements for the torch path are as follows:

- The torch must be held at a specified standoff distance relative to the surface in the specified direction.

- The torch must be oriented at a particular given orientation relative to the surface.
- The torch must be traversed over the work in a controlled path to scan the entire surface at the given spray pass increment, with minimum overspray and controlled speed.
- The spray path must have a certain amount of overlap with the preceding pass.
- The torch must not stop on the part.

The first two requirements are satisfied by moving the torch tip on the offset surface and the torch top on the corresponding points in the extended offset surface. To satisfy the third condition, the torch is started from the corner point B and moved in the direction of stroke, as shown in Fig. 7, until it reaches the last point in the current stroke (i.e., point C). To avoid overspray, the torch is moved from point C in the increment direction by the spray pass increment distance to point D. At the end of each stroke the torch is incremented to the next stroke. The entire surface is scanned in a continuous loop, thereby preventing the torch from stopping at any point on the surface.

The path-planning module consists of arrays for the coating surface, offset surface, and extended offset surface. The array $C[u][v]$ refers to the array of 3D points on the coating surface mesh. The u and v parameters are the indices in the u and v directions used to specify the mesh point location. For example, $C[10][15]$ refers to the 3D (x, y, z) mesh point corresponding to the 10th line in the u direction and the 15th line in the v direction. Similarly, the offset surface mesh is defined by the array $f[u][v]$, and the extended offset mesh is defined by the array $e[u][v]$.

To implement the torch path, the torch is incremented through the offset mesh points, maintaining the corresponding orientation at each point. Thus, inputs to the path-planning module are the offset mesh points array and the orientation at each point. This module generates the sequence of increments of array indices so that the torch follows the planned path in a continuous loop and scans the complete surface.

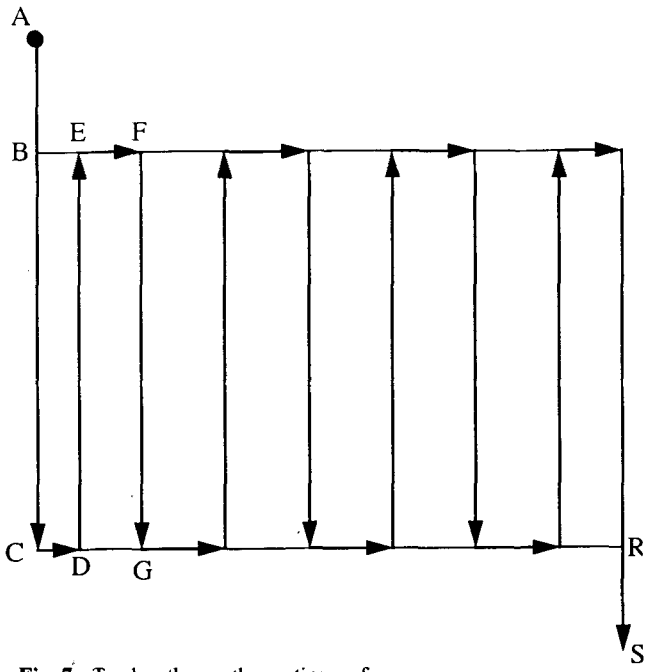

Fig. 7 Torch path over the coating surface

One of the requirements for the torch path is that the position and orientation of the offset points be generated to coat the corresponding coating surface mesh. Because the measurements are predetermined to maintain the spray pass increments, the torch paths achieve precise coating characteristics. All the torch positions at each increment for a sample surface are shown in Fig. 8.

The coating surface mesh provides information about the surface only at the mesh points; the surface points between the mesh points are unknown. Because the offset surface mesh points are calculated for every coating surface mesh point, the location points between adjacent offset surface mesh points also are not available. Thus, the torch trajectory between adjacent mesh points is not fixed. Mesh spacing in the increment direction is fixed and cannot be changed for a given coating setting. In scanning the surface, the critical direction is the stroke direction of the torch. To get a close approximation to the original path, the torch is moved through a straight line between mesh points, gradually changing the orientation from initial orientation to final orientation, as shown in Fig. 9. The next complexity in choosing a straight-line path between mesh points is the transition between two consecutive straight-line paths. This transition must be as smooth as possible and should be continuous. This is ensured by the continuous path option available in all standard robots.

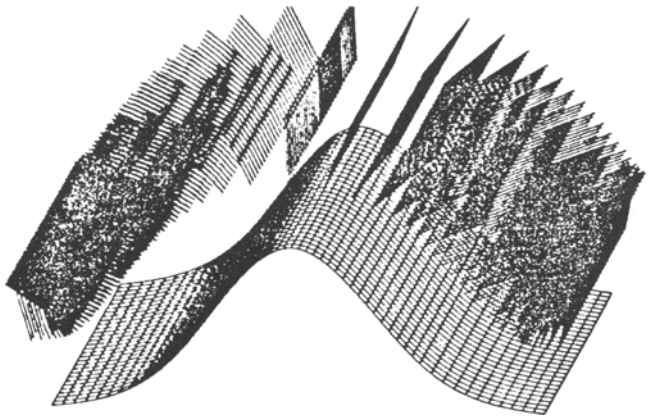

Fig. 8 All the torch positions for a sample surface

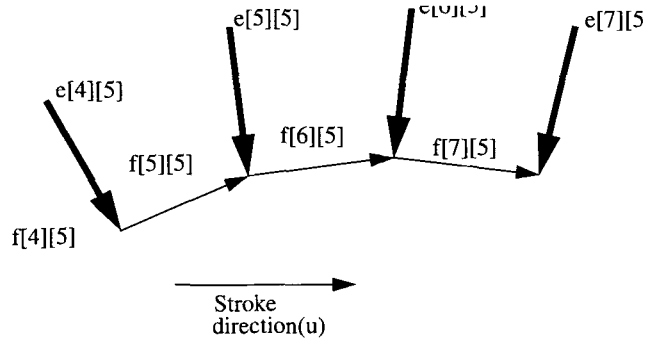

Fig. 9 Path complexity: straight line, continuous motion

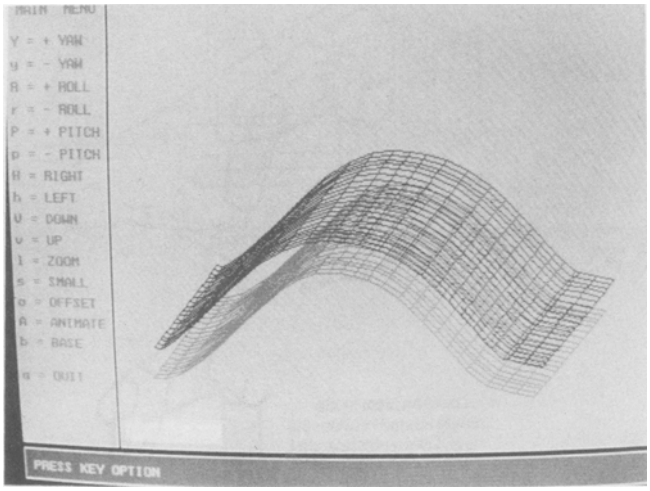

Fig. 10 Sample surface and offset surface mesh

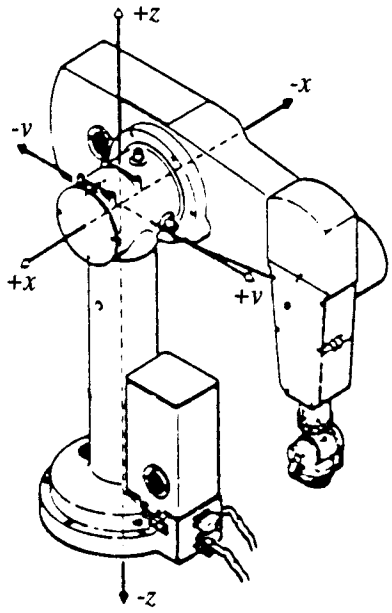

Fig. 11 PUMA world coordinate system. Source: Ref 8

The generated coating surface and the torch path over the surface must be checked for accuracy before implementation. The simulation module developed in this work provides a menu-driven user-interface graphics routine to display and manipulate the surface and offset surface mesh on the screen. The graphics screen is shown in Fig. 10. The simulation module has three submodules: wire frame surface display module, surface manipulation module, and spray torch animation module.

The wire frame surface display module is designed to display a 3D mesh of the coating surface and the corresponding offset surface. The surface manipulation module uses a menu-driven graphics screen that manipulates the orientation of the displayed surface on screen to obtain the desired view. Many surface manipulation options are available, such as ±PITCH(P) for rotation about the x axis, ±ROLL(R) for rotation about the y axis, and ±YAW(Y) for rotation about the z axis. Finally, the spray torch

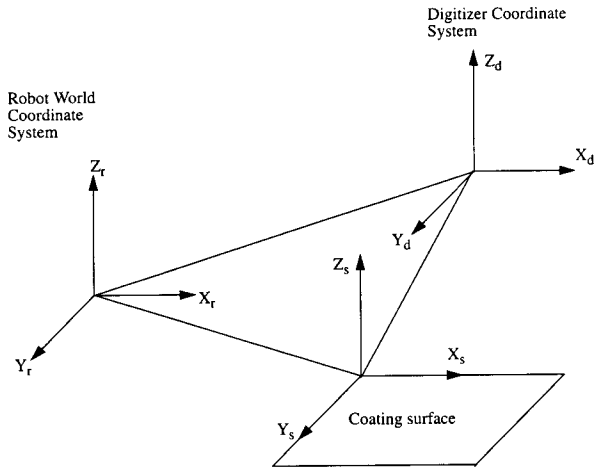

Fig. 12 Digitizer-to-robot coordinate mapping system

animation module animates the torch over the coating surface in the planned path. This animation helps to check the process for torch collisions with the surface.

3. Phase 2: Implementation and Results

The functionality and accuracy of the developed algorithm is demonstrated and the results are verified in this phase of the work. A robotic system used for coating complex surfaces should have six or more degrees of freedom. In this work, a PUMA 760 industrial robot, which is a six-axis system consisting of a teach pendant, computer controller, robot arm, and peripherals, is used for torch manipulation. A description of this system and its method of programming is given in Ref 8. For the present coating application, the following options of the PUMA robot are selected: (1) programming mode using the VAL II, (2) definition of location point using transformation, (3) straight-line motion between successive path points, and (4) continuous-path mode between straight-line motions. The robot mapping module, postprocessor generation, and downloading modules developed in this work are discussed briefly.

3.1 *Robot Mapping Module and Location File Generation*

The PUMA robot operates on two main coordinate systems: the world mode and the tool mode. In this work, the world mode is selected. The reference coordinate system for the world mode is fixed in the robot arm base, as shown in Fig. 11. The files in the digitizer coordinate system are mapped to the robot coordinate system using the "FRAME" command option available in VAL. The coating surface is located in the robot work space, three points are sampled in a particular order, and the x, y, and z mapping transformation coordinates are calculated. These mapping coordinates are used to transform the digitizer coordinate system to the robot world coordinate system. The "FRAME" command is discussed in detail in the VAL programming manual (Ref 8). The coordinate transformations are represented in Fig. 12. The mapped offset and extended offset files have the x, y, and z coordinates in the robot coordinate system at the base of the ro-

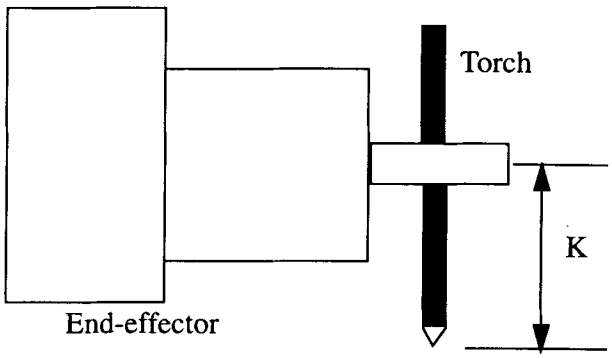

Fig. 13 Tool tip offset (K) from end-effector coordinate system

bot. These files are used to generate the location transformation (X, Y, Z, O, A, and T) for all points of the robot tool tip. For example, consider a single location point, with 3D coordinates of offset points ($f(fx, fy, fz)$) and 3D coordinates of the corresponding extended offset point ($e(ex, ey, ez)$) in the robot coordinate system. The location transformations X, Y, and Z represent the x, y, and z coordinates of the end effector—that is, the x, y, and z coordinates of the point centrally located between the fingertips in the world coordinate system. In the present work, the torch is clamped to the end effector (held between the fingers), as shown in Fig. 13. The offset point should coincide with the nozzle (tip) of the torch, which is at a fixed distance, K, from the center of the end effector. To compensate for this offset due to the torch mounting, the "TOOL" command available in VAL is used. This command offsets every position information by a given amount in the x, y, z direction to compensate for the center of the finger.

The orientation, altitude, and tool (O, A, and T, respectively) angles specify the orientation of the hand at position X, Y, Z. The O, A, and T angles are described in Fig. 14. In this application the axis of the tool (torch) is aligned in the "Tool Y" direction, and the O, A, and T angles are calculated as per Fig. 14. Thus, the mapped X, Y, and Z coordinates of the offset point and their O, A, and T angles are calculated for each location and stored in a file sequence as the transformation locations. Each transformation is named in ascending order of the sequence of via points.

3.2 *VAL Path Program Generation*

The path program is a list of instructions the robot is commanded to perform. The task to be programmed for this application is to move the robot tool through the planned path at a preset speed. The robot modes for this application, as discussed earlier, are preset. For this application only a few VAL commands are needed. A program is written for all the via points of the continuous-loop planned path in sequence. This program is created in the VAL program module in the computer and later downloaded to the robot controller through an interactive communication program.

3.3 *Downloading Files to the Robot Controller*

The VAL location file and the VAL program file are created in an off-line computer and stored on disk. These locations and program files are downloaded for execution to the robot controller using an RS232 serial communication link. An interactive ro-

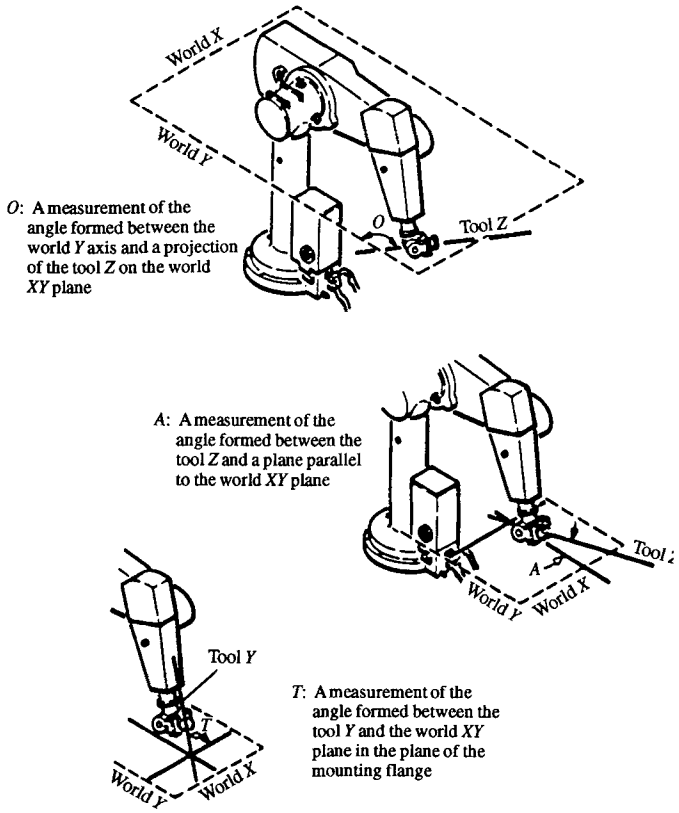

Fig. 14 Descriptions of O, A, and T angles. Source: Ref 8

bot/computer interface program is developed in BASIC for downloading the programs. Once the location files and VAL program are loaded to the robot controller, the files are transferred to the robot disk for later use. The communication program has the following modules:

- *Asynchronous port*: Here the asynchronous port is opened for bidirectional communication at 9600 baud. The advantage of using terminal communication is that the maximum communication speed can be achieved.
- *Error check*: This module checks for port communication errors and displays any error statements.
- *Robot initialization*: This module loads the VAL system, calibrates the robot, and initializes the robot.
- *VAL location file*: This module opens the location file in the floppy disk for reading, receives the location transformation (X, Y, Z, O, A, T), and stores the location point in the given name in the robot controller.
- *VAL program file*: This module opens an edit file for generation of the program, reads the VAL program from the disk, and stores it in the controller memory.
- *Execution*: This subroutine executes the stored program. The execution of the program results in moving the torch fixed to the end effector through the location points, while maintaining the required orientation in the planned path and thus coating the surface.
- *Store*: The location files and the VAL program are transferred from the robot controller memory to the disk mem-

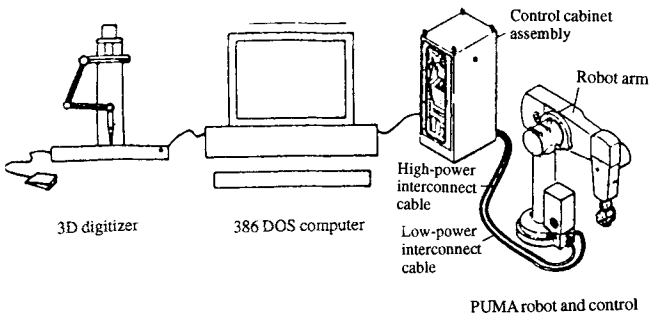

Fig. 15 Experimental setup

Store: The location files and the VAL program are transferred from the robot controller memory to the disk memory available in the robot. This disk can be used to execute the program later.

The robot communication program is interactive and user-friendly, allowing easy storage and program execution.

3.4 *Experimental Setup*

The computer-aided torch trajectory generation technique described in this paper is implemented using a 3D digitizer (PERCEPTOR), a 386 DOS computer, and a Unimation PUMA 760 robot with a spray torch mounted at the end effector (Fig. 15). The digitizer is interfaced with the computer, where the algorithm resides. The algorithm generates the torch path and animates the process on screen. It also has additional modules for robot location and program generation. The computer is interfaced with the robot controller terminal to download the programs and for execution\

Considering the complexity in both parametric directions, a surface digitization mesh has been chosen (see Fig. 2). The surface is placed in the digitizer workspace and the points are digitized in the order specified in Phase 1 of this paper. During digitization, the probe tip is placed on the point to be digitized and the foot pedal is depressed. This sends the 3D digitized coordinates to the output port. The interactive digitizer/computer communication program receives the 3D information, displays the 3D coordinates, and stores the digitized points file.

The software package developed for this application is called CA-COATS and has been implemented at Michigan Technological University (Ref 9). The package is interactive, and the screen-by-screen method of interactive input makes the system more user-friendly. To run the program, the disk containing the digitized data is inserted in the computer. A series of screens appear one after the other, prompting for input values.

4. Conclusion

The work presented in this paper provides an efficient method for solving the difficulties in path programming for automated coating applications. This method uses a sampled surface to construct the surface geometry and generates the torch trajectory as well as the required path program. The algorithm developed generates the bicubic B-spline surface interpolating the sampled points, thus completely defining the coating surface. Using the coating surface information, and maintaining the standoff distance and orientation requirements, the torch positions and orientations are calculated. Then the torch path to coat the given surface is generated. The generated coating surface, offset surface, and torch path are checked using the simulation module. The torch trajectory generation system developed is verified for various complex sampled surfaces. The results of the interactive menu-driven program are presented in the form of figures. Animated torch paths for different standoff distances and orientations are checked. An additional module for surface manipulation on screen allows the user to check for any collision between the torch and the coating surface. The surface generation module is checked using several digitized surfaces for accuracy.

Because a PC is used for the off-line programming, the system is economical. This method lessens the programming effort considerably. As the computer is used to generate path programs, the programming is user-friendly and has adequate flexibility. This off-line method completely eliminates robot downtime. The animation of the torch motion helps to check the process before implementation, also preventing downtime. This method is designed to accommodate further variations in the coating process, including torch speed, changes in number of passes, rotation of components during coating, and use of coordinated robots.

This method is suitable for painting, thermal spraying, and other coating applications. With minor modifications it can also be used for waterjet paint-removal systems, coordinated robots, and programming paths for buffing and polishing robots. The work presented in this paper provides the base for the development of a full-fledged automated coating system and for implementation with other commercial coating robots.

References

1. DeVilbiss Inc., Spraying Their Way to Success, *Ind. Robot*, Vol 16 (No. 1), 1989, p 13
2. M. Lande, Automated Painting System Installed on the Fly, *Robotics Eng.*, Vol 8 (No. 7), 1986, p 5
3. R.R. Schreiber, Portrait of a Painting Robot, *Manuf. Eng.*, Vol 107 (No. 5), 1991, p 55-61
4. M.E. Mortenson, *Geometric Modeling*, John Wiley & Sons, 1986, p 220-226
5. D.F. Rogers and J.A. Adams, *Mathematical Elements for Computer Graphics*, McGraw-Hill, 1990, p 445-456
6. B.A. Barsky and D.P. Greenberg, Determining a Set of B-Spline Control Vertices to Generate an Interpolating Surface, *Comput. Graphics Image Process.*, Vol 14, 1980
7. G. Farin, *Curves and Surfaces for Computer Aided Geometric Design—A Practical Guide*, Academic Press, 1988
8. "Unimate Puma Mark II Robot 700 Series," No. P/N 398N1A and No. 398N2A, Unimation Inc., Danbury, CT, Aug 1983
9. T. Vivekanandhan, R.A. Kashani, and W.A. Johnson, Intelligent Plasma Torch Trajectory Generation for Thermal Spraying of Parts with Complex Geometry, *Thermal Spray: International Advances in Coatings Technology*, C.C. Berndt, ASM International, 1992, p 217-220