COBEM 99
XV Congresso Brasileiro de Engenharia Mecânica
15th Brazilian Congress of Mechanical Engineering
22 - 26 de Novembro de 1999 / November 22 - 26, 1999 Águas de Lindóia, São Paulo,

A CAD/ROBOTICS SYSTEM TO "OFF-LINE" PROGRAMMING OF ROBOTS

Altamir Dias[f]
Universidade Federal de Santa Catarina, Departamento de Engenharia Mecânica
C. P. 474 – 88040-900 – Florianópolis, SC, Brazil;
Leonardo B. de Toledo
Universidade Federal de Santa Catarina, Curso de Pós-Graduação de Engenharia Mecânica
C. P. 474 – 88040-900 – Florianópolis, SC, Brazil;
Michael Deisenroth
Industrial & Systems Eng. Dept., Virginia Polytechnic Institute, Blacksburg, Virginia –USA

ABSTRACT: The effort presented in this paper explores the development of a system for programming robots directly from CAD based geometric models. The interaction between several different software systems to define paths, to obtain the inverse kinematics of robots, and to prepare data to be translated in a procedural language for programming robots is required. The concepts were tested using a computacional model of a PUMA 560 robot to follow a defined path over a B-spline surface. Cartesian position, velocities, and accelerations were obtained from the software system and analyzed prior to running the control program on the robot.

Keywords: Trajectory, path planning, kinematics.

1. INTRODUCTION

Today, the robots are the most significant element in the automation system in manufacturing industry process. Automation system denotes a development of technologies to substitute handling operations performed by human being in a manufacturing process, as regards not only the execution of physical operations, but also the transformations needed to complete the whole process itself. It is spread in all fields' human activities, but mainly in industrial plant tasks. After 1970's more and more systems are coming using the robots and manipulators as part of productive system. And now, most of repetitive and hazardous operations performed by human being are progressively being substituted by this kind of machine.

One of the main characteristics of industrial robots is the fact that being a more complex machine, joining computers and mechanical parts, resulting in a device easily reprogrammable, without to have being changed physically as it occurs in the rigid automation. Typical applications include performing tasks like arc and spot welding, spraying paint and polishing. Additionally, they can be programmed to explore the 3D space, to get

[f] Thanks: CAPES- Coordenação e Aperfeiçoamento do Ensino Superior-Brazil.

measurements and test products and its quality, within a product test system and in environments where human activities would be in risk.

A recent development in robotics is to program the robots far of production line, by techniques of "off-line" programming. Off-line programming avoids that the robots have to be stopped while they are programmed or taught to perform a new task. Such approach save more time than programming robots directly in the production system.

In this paper, we are going to explore robot applications where the motions of the robot can be described by a set of continuous paths. Welding, gluing, polishing and painting can be classified in this set of tasks. In such applications, the path of the end-effector can be described by a geometric offset of curves on the contour of object that needs to be worked. CAD/CAM systems offer consistent geometric representation of parts that facilitates getting data needed to describe paths over the parts contours.

2. INTEGRATION CAD-ROBOTICS

The integration of CAD/CAM has been used widely in programming industrial machine tools. The basic approach is to extract information from the CAD file format and translate its contents to a language that can be interpreted by the controller of machines used in manufacturing. The same approach can be applied for programming industrial robot. Although, robots have new features, such as more degree of freedom in moving and more complexities in programming, the geometric properties of objects that need to be manufactured can be used to know in advance the robot joint motion to perform a desired task.

2.1 CAD/CAM Data Exchange

Computer databases have been used in defining geometry and nongeometry data for all phases of the product design and manufacturing. However, there are several incompatibilities among entity representations that complicate exchanging modeling data among CAD/CAM systems. It is complicated for several factors like the complexity of CAD/CAM systems, the varying requirements of organizations using them, the restrictions access to proprietary database, and the rapid pace technological change.

Databases are designed to exchange product data among CAD/CAM systems. They have to provide information about shape, non-shape, design and manufacturing data from the product. IGES (Initial Graphics Exchange Specification) data format has been used by CAD/CAM systems as a standard exchange file. However, it is focused on CAD-to-CAD exchanges, where only shape and non-shape data between one system to another. Needs to have exchanges of complete product descriptions have been proposed with PDES (Product Data Exchange). The STEP, “a international standardization project”, whose scope format incorporated the four types of modeling data cited above, will be providing in near future a better way to exchange data to manufacturing systems.

For the purpose of path planning trajectory data from boundary of parts are necessary. Modeling based in surfaces and solid modeling offering in theirs data, topological information that are needed for this objective. If path-planning programming is done in specific CAD/CAM systems, graphical interface tools are offered in order to get data according with the file format specification of system. Some systems provide interaction with common language programming, such C, C++, LISP and others. Otherwise, a specific interface environment has to be used in order to recreate the shapes where path planning are going to be planned. Language environment programming has offered resources to create an interactive environment for that purpose. Mathematical packages, such MatLab, Mathematics, with theirs internal languages, are also having functionality to develop application where path planning can be done.

2.2 Tool path generation

The main task in the part programming is to describe the trajectory that the manipulator’s end-effector has to follow, which include the path as well as the direction, and speed of travel. In the beginning of robots’ programming history, most of tasks were programmed manually using the preprocessor languages or even though with the help of a calculator. The result was transferred to the memory of the equipment to be used later. More complex programs are been used in computer's environment programming together a specific part programming language. Specifics preprocessor language, like APT, AML, SIMPLE, VAL and others vendor language, is used to instruct manipulators to follow the activity planned.

A way to create robot task planning programs is to use directly CAD/CAM systems. It allows the task programmer to access the computer's capabilities interactively. Also, it uses geometry described in form of points, arcs, line and so on, in the same way as we reproduce an engineering drawing. The graphical terminal can be used to display the resulting end-effector path geometry. It allows earlier verification of a program avoiding costs associated to machine setups for program tests. Also, CAD/CAM systems provide productivity gains. The programmer generates an end-effector path by selecting geometric elements with a digitizer attached to the display terminal. Several auxiliary and postprocessor commands are entered through the terminal. The machine tool information is automatically extracted from the system and storage in a magnetic medium to be used by a postprocessor. Finally, the program can be reproduced on a punched tape or whatever medium required by the specific NC machine tool.

Some advantages of applying computer-aided part programming are described as follows:

- It reduces the manual calculations involved in determining the geometric characteristics of the part. By having the bulk of calculations handled by a computer, most of the errors are eliminated.
- A program preparation system should have included capabilities as path simulation on a computer screen. So, most errors in that program can be identified before the program is loaded in the machine. It means some saving in machine time.
- A set of events can be programmed with fewer commands, leading to shorter programs that take less time to prepare and are more convenient to store.

Most of commercial CAD/CAM system such as CADAM, Computervission, and CATIA have the capability of generating NC machining instructions based on the geometric definition of a workpiece. A single idea about those systems available on the market is given as follow (Wysk, 1998).

Numerical control offers accumulated experience in part programming and variety of methodology that can be used as initial point for tool path generation. Most of literatures are related to numerical control milling path generation, drawing from engineering, computer science and mathematics. It forms a considerable set of underlying information to the path planing generation for robots. Although the number of degree of freedom of robots are greater than machine used in numerical control, many geometric aspects to generate path are the same and can be used to plan tasks.

There has been published a considerable amount of researches on numerical control tool path generation since beginning of 80s (Dragomatz, 1997). Many of material are written in a subject-specific terminology, which might cause some difficult to cross information relate to particular area of interest in tool path generation.

The tool paths are a sequence of the curves on the design surface. To find curves on design surface is common to use methods of generating tool paths by intersecting user-specified surfaces with the design surface (Bobrow, 1985, Huang, 1994). Surface-surface computations are very expensive and should be avoided (Loney, 1987). A good solution is to use iso-parametric curves as tool paths, but in general the spacing between iso-parametric lines will be non-uniform and the specific surface finishing could not be reached. Offsets on the design surface have been used by Suresh. He uses a constant scallop height on the manufactured surface to evaluate the offset surface (Suresh, 1994). Sarma has said that this method yields the shortest overall length of tool paths compared to the other path generation method (Sarma, 1997). However, this method doesn't provide local and global control of the tool path. The sequence of tool path is generated from an initial path curve that might be in bound of surface.

3. PATH PLANNING

Several different software systems, different mathematical concepts and algorithms compose a task planning system to programming robots. Computational procedures are needed to represent the part surfaces, compute matrix transformations, and evaluate the inverse kinematics of robots. The geometric description of continuous path is generated and position, velocities and accelerations of the robot joints are evaluated and compared against possible limits. The results have to be translated to a specific robot language and the programming is done. Additional analysis algorithms can be included to improve the performance of the system but are not included in this discussion.

3.1 Defining Path on the Surfaces

Operations such as arc welding flame cutting, deburring, painting and routing appears as continuous path robotics applications. In that manufacturing operations, a tool is rigid attached to the end-effector of a robotic manipulator. The tool has to travel in a continuous and smooth trajectory in a 6-dimensional configuration space. Three dimensions are related by a operation point of the end-effector, while the remaining is associated with the end-effector. Those applications require that the task take place along a warped curve, resulted of intersections of surfaces, while the path to be traversed is prescribed as time function. To define the orientation a Frenet-Serret frame is associated with every point of the path.

Manufacturing activity on surface use some standard forms of path definitions. Continuous path defined in machining operation can be a way to generate path. In this case, the most common moving of tools on machining are: zig-zag, spiral inside-out tool path, and spiral outside-in move. A zig-zag path is showed in the "Figure 1".

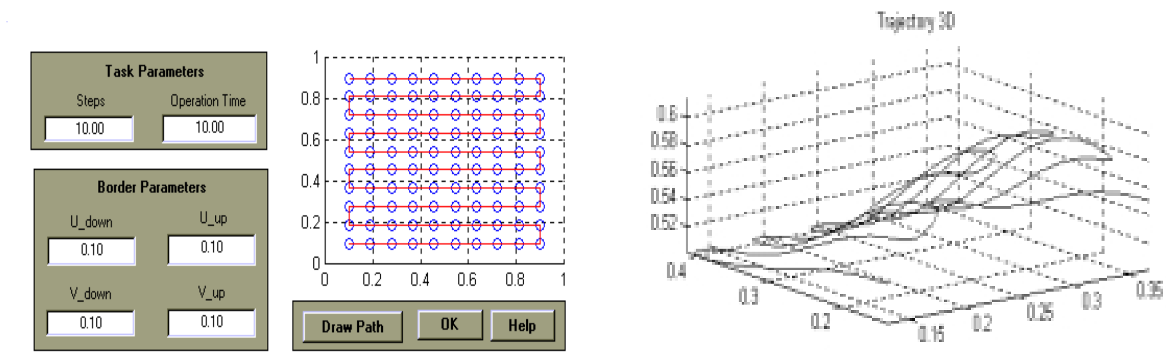

Figure 1. A zigzag end-effector path on parametric space and the path on 3-D surface.

The moving can be planned either in the parametric space of the part's boundary or in the Cartesian space - on surface itself. The former is easy, because the path is generated in the 2-D space. However, it doesn't give precision finishing as required in the machining operation. The last case requires using geometric properties of surface and algorithm to describe the moving adequately. Geometric relationships give the information to evaluate the steps to be followed by tools on the path, in such way that the tool follows exactly the shape of surface. Also, the "Figure 1" shows a zigzag path on the 3-D surface.

.

(a) (b)

Figure 2. Forward step, cordal deviation (a) and scallop height resulting of two paths.

We can associate with every point of curve to be followed an orthonormal triad of vectors, which defines the tangent, the normal, and the binormal to the path curve. Therefore, when this set of vectors is properly arranged in matrix array, rotation is obtained and also the orientation of curve. These vectors are known the Frenet-Serret vectors. Geometric concepts of curvature and torsion are useful to establish the relationships between Frenet-Serret vectors and others important properties from the geometry. The curvature is interpreted as the rate of change of orientation of the tangent vector with respect of the arc length and the torsion represents the rate at which the curve quits the plane of the tangent and normal vectors. The Frenet-Serret vector formulas and the chain rule can be worked to have the time rate Frenet-Serret vectors.

For trajectory planning purposes, the tool paths can be approximated by a series of straight-line segments. There are several ways to get the definition of the forward step along the path. One common approach is to evaluate the forward step f_s by the distance between the main path on the geometry and the path performed by the tools, as showed in the "Figure 2a". The difference defined above is called of cordal-deviation and can be controlled by the manufacturing tolerance required δ_1 to machining a part.

The maximization of the forward-step is a critical problem. The objective is to achieve the value of f_s given the tolerance δ_1 of design. There are several approaches to solve this problem. Loney and Ozsoy determine the forward-step by solving the polynomial equation by imposing certain geometric constraints (Loney, 1987). Huang proposes an improving on this approach where they correct the value due to the fact that the tool in general is not normal to the tool path in the end of the chord (Huang, 1992). A simplistic approach was proposed by Suresh and indicated in several literatures, is to use a circular approximation as a function of δ_1, and showed as follow (Suresh, 1994):

$$f_s = \sqrt{4\delta_1(2\rho - \delta_1)^2} \tag{1}$$

where ρ is the local curvature radius along of $P(t)$.

Another factor to consider is to obtain a general path with a sequence of parallel moving that the tools has to follow in order to accomplish the task. It is defined by the side step that is the deviation of the tool relative to the prior path and showed geometrically in the "Figure 2b". The unamachined region is called the scallop or cusp, or yet, finishing criteria will be used as criteria to establish the value of side step to be done.

The step-side is a function given $g = f(h_s, r_t, \rho)$, where h_s is the permissible tolerance of finishing criteira (scallop-height), r_t the tool radius, and ρ is the local curvature radius of the surface. The end shape of the tool is assumed to be a sphere with radius r_t, and it will simplify the computations.

Let's define the radius of the surface ρ as $1/\kappa$. Now, ρ can be obtained from the relationship between the equations of the first and second fundamental form of surfaces. The terms u' and v' in the equation of curvature, are replaced by $u_g^{'}$ and $v_g^{'}$, where the subscript g are computed the direction of side step. Being $u_g^{'}$ and $v_g^{'}$ vectors perpendicular to the direction of tool path, it satisfy the following expression

$$(P_u u^{'} + P_v v^{'}) \cdot (P_u u_g + P_v v_g) = 0 \tag{2}$$

Let define the ratio between $u_g^{'}$ and $v_g^{'}$ equal $\alpha = u_g^{'} / v_g^{'}$. So, we can express the equation of curvature, and in consequence to compute the radius ρ as a function of ratio α. It reduces the equation

$$\rho = \frac{E + 2F\alpha + G\alpha^2}{L + 2M\alpha + N\alpha^2} \tag{3}$$

where E, F, G, L, M and N are terms from the first and second fundamental form of surface.

Now, the side step g can be evaluateed as a function of the h_s, r_t, and ρ. Let's consider the geometric case presented in the "Figure 3". Different cases can appear depending on the curvature of the surface. The surface can be a plane, $\rho = \infty$, or a surface either convex or concave curvature. Considering the case of convex curvature, $\rho < 0$, g is evaluated using the following relationships:

$$\begin{aligned}
Oc_3 &= \sqrt{(Oc_2)^2 - (c_2c_3)^2} = \sqrt{(r_t + \rho)^2 - a^2} \\
c_3x &= \sqrt{(c_2x)^2 - (c_2c_3)^2} = \sqrt{r_t^{\,2} - a^2} \\
Ox &= \rho + h = Oc_3 - c_3x \\
h &= \sqrt{(r_t + \rho)^2 - a^2} - \sqrt{r_t^{\,2} - a^2} - \rho \\
a &= \sqrt{r_t^{\,2} - (\sqrt{r_t + \rho^2 - a^2} - \rho - h)^2} \\
g &= \frac{2a}{(1 + r_t / \rho)}
\end{aligned} \tag{4}$$

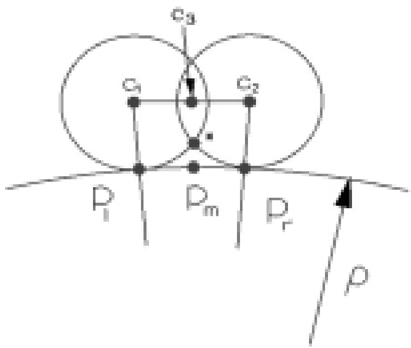

Figure 3. Tool-positions for two consecutive paths.

where $c_1c_2 =$ distance between tool center in two consecutive paths $= 2a$; $p_m x =$ finishing criteria $= h_s$; $p_l p_r =$ side-step $= g$; $\rho =$ radius of curvature along $p_l p_r$; $O =$ center of curvature. Some of these geometric elements are showed in the "Figure 3". The value of a in the "Equation 4" is evaluated iteratively from an initial value of a, which converges faster. Others cases can be easily computed by using these geometric elements and triangles relationships.

Now we can generate a new tool path if we know the previous tool path. It is doing in such way that the finishing criteria traduced as a scallop-height will be kept constant along the surface. We assume that the previous out master path is defined by $P(u,v)$, which is a parametric surface. At a point $P_i(u(t_i), v(t_i))$ we determine the allowable side-step using one of cases above depending from the local curvature of surface. The parallel point P_g is defined as the point on the surface along the geodesic path perpendicular to the previous path. It is a distance g from the original point P_i. From that information, we can write that $(P_g - P_i)\cdot(P_u u' + P_v v') = 0$. It specifies that the parallel point P_g should lie on the path orthogonal to the path previous. The distance g from the initial point is given by $\| P_g - P_i \| = g$.

Let P_g expand as a Taylor series neglecting the second and high order terms: $P_g = P_i + P_i(u(t), v)\Delta u + P_i(u, v(t))\Delta v$. If we combine the last three expressions, it can be written in terms of the coefficients of the first fundamental form of a surface. So,

$$E\Delta u \cdot u' + F(\Delta u \cdot v' + \Delta v \cdot u') + G\Delta v \cdot v' = 0 \tag{5}$$

$$E(\Delta u)^2 + 2F\Delta u\Delta v + G(\Delta v)^2 = g^2 \tag{6}$$

The value of Δu and Δv can be computed from the equations above and result in

$$\Delta u = \frac{\pm g(F \cdot u' + G \cdot v')}{\lambda \cdot ds} \tag{7}$$

$$\Delta v = \frac{\pm g(E \cdot u' + F \cdot v')}{\lambda \cdot ds} \tag{8}$$

where $\lambda = \sqrt{EG - F^2}$ and ds is the definition of the first fundamental form of surface. Thus, the parallel path P_g to $P(u, v)$ is defined by

$$P_g(t) = P(u(t) + \Delta u(t), v(t) + \Delta v(t)) \tag{9}$$

Also, the sign of Δu and Δv are defined by direction of end-effector moving.

3.3 The Kinematics Problem

Basically, the problem of path planning of robot can be done whether in the joint space or in the workspace. Joint space is a mathematical space defined by the number of joints variables associated to the robot and its degree of freedom that the robot can have which gives the dimension of that space. On the other hand, the workspace is place where the position of the end-effector is defined and tasks are performed. It is normally defined as the E^3 dimensional space, although some task can be defined with a lower dimensional space. The relationship between both spaces is given the direct kinematics equation of the manipulator. The position and orientation of the end-effector frame are function of joint variables with the respect to the base frame. A robot is a sequence of links and joints, which are related each other by a set of translations and rotations matrix respectively. The chain of transformation of each frame attached to the joints, result in the following equation:

$$T_n^0(q) = A_1^0(q_1)\cdot A_2^1(q_2)\cdots A_n^{n-1}(q_n) \tag{10}$$

$T_n^0(q)$ is the position of the end-effector, $A_i^{i-1}(q_i)$ is the matrix that represent the translation and rotation of the joint frame i related to the joint frame $i-1$. The matrix $A_i^{i-1}(q_i)$ is obtained using the Denavit-Hatenberg convention. It relates the relative position and orientation between the frames in terms of geometric measures of links and their respective angles.

The position of end-effector can be given by a minimal number of coordinates with regard to the geometry of the structure. And its orientation can be specified in terms of a minimal representation describing the rotation of the end-effector frame with respect to the base frame by the using of Euler or RPY angles.

In this way, it is possible to describe completely the end-effector frame by means of the $(m \times 1)$ vector s.

$$s = \begin{bmatrix} p \\ \phi \end{bmatrix} \tag{11}$$

where p and ϕ describe the end-effector position and orientation. The joint space denotes a space of $(n \times 1)$ vector of joint variables q:

$$q = \begin{bmatrix} q_1 \\ \vdots \\ q_n \end{bmatrix} \tag{12}$$

where q_i represents θ_i for a revolute joint and d_i for a prismatic joint. In this way, the dependence of position and orientation of the end-effector from the joint variables can be written as:

$$s = f(q) \tag{13}$$

where $f(.)$ is a nonlinear $(m \times 1)$ vector function that allows computation of the operational space variables from the joint space variables.

The inverse kinematics consists of the determination of the joint variables corresponding to a given end-effector position and orientation $q = f^{-1}(s)$. This problem is more complex due to some reasons: the equations to solve are quite always nonlinear and it is not possible to get a closed-form solution. Also, multiple solution may exist and in the case of redundant manipulator infinite solutions can be appear. Finally, some kinematics structure can result in no admissible solutions. However, the existence of solutions is guaranteed if the given end-effector position and orientation belong to the manipulator dexterous workspace. So, "off-line" path planning can be given a way to check admissible solutions to a task that can be planned in advance.

4. RESULTS

A MatLab interface GUI was developed to test the current approach. Also, it took advantage to use the built-in functions of MatLab environment. In this environment we can read the surface where the path will be and to specify the to be followed by the robot's end-effector. The “Figure 4” shows the interface used to deal with the elements and others parameter needed to evaluate a task. So, the initial data input needed to begin the analysis is the surface to be worked, the robot model and the path definition. The output are the position, velocities and accelerations in the Cartesian space and the positions and velocities of joints. The model of robot is setup by Denavit-Hatenberg convention, joint types and parameters that define the links. A MatLab package developed by Corke is used to build the algorithms to evaluate the inverse kinematics of robot (Corke, 1996). The analysis can be run interactively at different initial robot's arms position to find path, which does not violate joint limits. A wireframe simulation of the robot moving on the surface is also available.

A zigzag path is shown in the “Figure 1”. In this example the path is defined in the parametric space. “Figure 5” illustrate the plot obtained for position, velocities and acceleration in the Cartesian space, and the joint position and velocities in the joint space. A simulation of the robot can be performed in order to check if it can complete the task without to violate any constraints in the workspace.

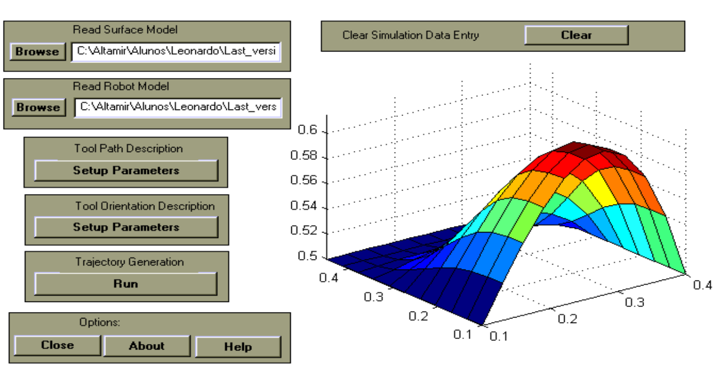

Figure 4. MatLab GUI interface to path planning.

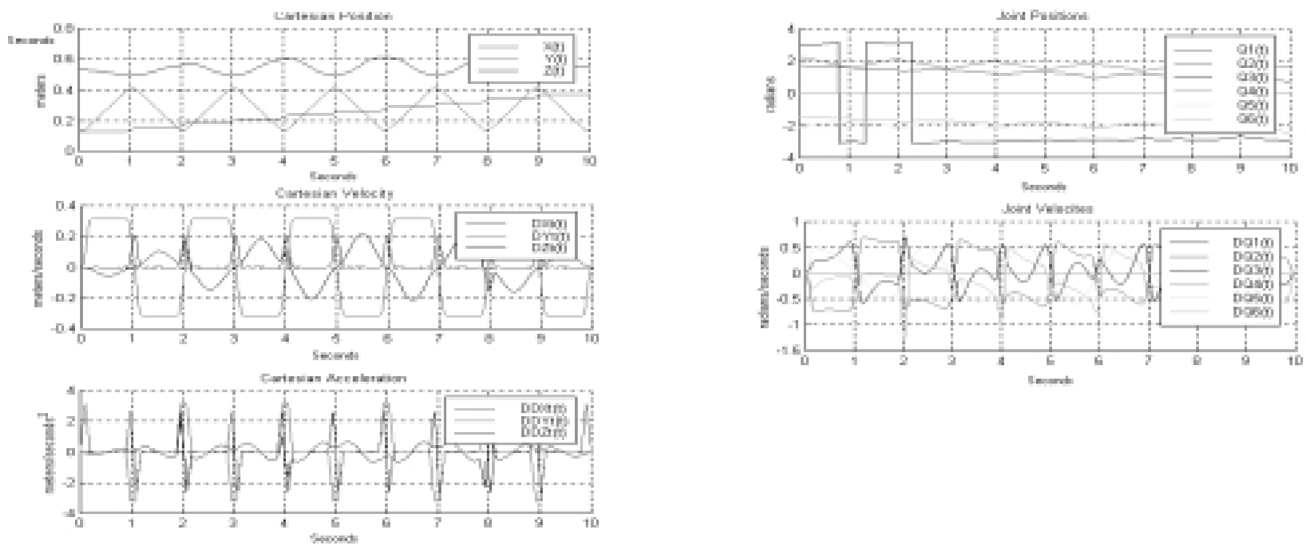

Figure 5. Cartesian and joint data of the path planning.

5. CONCLUSION

The present work shows that the integration of CAD robotics requires the use of information from different areas. CAD/CAM systems offer geometric information about the boundary of parts that has to be machined. It can be read through an appropriate translator in order to obtain data to reconstruct the shape of surface where the task will be planned. Path planning methodology used in the CAM machining activities, and geometric information from mathematical representation the surfaces are evaluated to find the path geometry and the pose the end-effector of robots in each position on the surface. From robotics theory, the inverse kinematics algorithms and methodology give support to evaluate the position and velocities of joint on that path. If constraints are violating, iteration from the user is done to recalculate the path. A task is said planned if no violation occurred. In such case, a reposition of the surface in the workspace of robot is needed, or a subdivision of the surface area have to be done in order to complete the planning and recalculate the path and other information. Once is concluded that the path is possible to be completed to surface position and initial pose of robot, the data can be translated to the robot's language.

Some new steps can be followed in order to improve the present approach. The present approach has to use several programs to be tested. The base is done in the MatLab environment. Although, MatLab offers good tool and skills to execute fast the tests of the algorithms involved, it has the restriction in using in the whole chain of a design/production system. So, an appropriate programming language, like C++, C, Java and so on, should give more flexibility in interact with CAD/CAM systems available in design/manufacturing offices. Also, it makes easier to develop algorithms that can be supported by CAD system, in robotics analysis and by CAM programming, because a language more generic offer more tools to integrate all the processes that path programming involves.

Also, a module to translate the data obtained to a robot language should be developed. Although, it might be difficult, because robot's vendor normally offers specific robot language; a standard way to translate data to programming robot will facilitate the integration required in programming.

In the future, an programming environment that include obstacles in the workspace, work cells programming concepts, and use virtual reality has to be developed in order to give continuity to the present research.

REFERENCES

Bobrow, J. E., 1985, Solid modelers improve NC machine path generation techniques. Computers in Engineering, 1, 439-444.

Corke, P. I., 1996, A robotic toolbox for MatLab, IEEE Robotics and Automation, 3,24-32.

Dragomatz, D. and Mann, S., 1997, A classified bibliography of literature on NC milling path generation. Computer-Aided Design, 29, 239-247.

Dias, A., Toledo, L.B., and Deisenroth, M., 1999, An “off-line” Programming System for Industrial Robots”, Flexible Automation and Intelligent Manufacturing Conference, 835-846.

Huang, Y. and Oliver, J.H., 1992, Non-constant parameter NC tool path generation of sculptured surfaces. Computers in Engineering, 1, ASME.

Huang, Y. and Oliver, J.H., 1994, Non-constant parameter NC tool path generation on sculptured surfaces. International Journal of Advanced Manufacturing Technology, 9, 281-290.

Loney, G.C. and Ozsoy, T. M., 1987, NC machining of free form surfaces. Computer-Aided Design, 19, 85-90.

Sciavicco, L. and Siciliano, B., 1996, Modeling and control of robot manipulators, McGraw-Hill, New York.

Sarma, R. and Dutta, D., 1997, The geometry and generation of NC tool paths. Journal of Mechanics Design, 119, 253-258.

Suresh, K. and Yang, D.C.H., 1994, Constant scallop-height machining of free-from surfaces. Transactions of ASME, 116, 253-259.

Wysk, R.A., Chang, T.C. and Wang, H. P., 1998, Computer aided manufacturing. Prentice Hall, New Jersey.