A Practical Method of Reverse Engineering and Automatic Path Programming for Robotic Surface Finishing

William T. Adams, John M. Fitzgerald, T.J. Lawley*, O. R. Mitchell†
The University of Texas at Arlington, Automation & Robotics Research Institute(ARRI)
Fort Worth, Texas

Abstract

This paper presents a new method of automatic path planning and trajectory generation for robotic surface finishing. Initial development is on a platform integrated from commercially available components including AutoCAD and a low cost CMM. Automatic program execution is demonstrated using a 6 DOF manipulator on a variety of complex shaped surfaces.

Introduction

The focus of research described in this paper was determined by the needs of end-users associated with The Automated Surface Finishing Consortium. The mission of the Consortium is to develop finishing automation as an end-user technology for U.S. manufacturers. Members are military and commercial manufacturers, universities, federal labs, and commercial suppliers of abrasive process technology and robotics technology.

An objective measure of technology development/deployment impact is the availability of successful commercial off-the-shelf technology. The synergistic interactions of the Consortium members have yielded impressive results to date. Force controlled robot end effectors commercialized by collaborating members have significantly broadened the boundaries of available compliant finishing process capability. A generic off-the-shelf integrated robotic finishing system, the product of another collaboration, is now distributed and supported by the largest robot company in the world. Likewise a low cost high performance simulator based surface path programming aid is now a commercial product. ARRI is host to the Consortium, and a member.

Background

Most material forming processes do not produce finished parts. Most machining operations leave burrs and sharp edges. Large aircraft wing skins are milled by three axis terrace cutting leaving small steps which must be blended to prevent fatiguing stress concentrations. Complex curved surfaces like ship propellers and aircraft landing gear are machined with rounded milling tools which leave a pattern of tool marks which must be ground off by hand. Cast parts require gate and sprue removal and deflashing. When castings are machined they require deburring. Many parts must have their surfaces conditioned for appearance or subsequent plating and coating operations. The stamping and forging of automobile door panels and engine components leave "imperfections" which are finished out by hand. Die cast surfaces of hardware for door handles and hinges are ground and polished. Wrenches, golf clubs, vacuum cleaners, furniture, boat hulls, all have to be ground, sanded and polished to have a nice appearance.

The costs of manual finishing are high. A recent report by the National Center for Manufacturing Sciences suggests that direct finishing costs exceed 25 billion dollars annually based on mechanical manufacturing industry gross sales of 1,000 billion dollars[1] . Often in precision parts manufacturing and particularly with complex shaped parts there is high finishing labor content. Jet engine component manufacturers in the U.S. and Canada report that between 15% and 30% of the direct cost content of their products is finishing labor. A major consequence of manual part finishing is scrap and rework. Although difficult to quantify, interviews of finishing workers in high value added manufacturing invariably reveal costly scrap and/or complicated expensive rework

*Professor, Department of Mechanical Engineering, UTA
† Chairman, Department of Electrical Engineering, UTA

procedures. This is a direct result of manual finishing errors.

Most of the finishing laborers are grinding, filing, sanding and scraping parts by hand. Some finishing jobs are high paying, requiring artisanship based on years of practice. Most finishing jobs are low skill low quality jobs, typically they are the least desirable and lowest paying. The reason is that manual finishing work is drudgery of the worst kind, exposing workers hands and arms to mechanical impact and vibration while requiring repetitive motion, and often producing dangerous and toxic dusts. Fortunately, turnover rates are relatively high in the lower paying jobs which helps to limit prolonged exposure. There is a great need for practical hands-off finishing process capability.

The application of abrasive processes to meet finishing needs is diverse and widespread in manufacturing. The abrasive industry, from mining through tool design, manufacture, and distribution, is a multi-billion dollar per year industry. The number of permutations of tool size, shape, abrasive type, and other characteristics of manual abrasive tools is astronomical. Finishing processes are typically performed manually, the process is planned based on knowledge and experience, and controlled by the operator with his senses of sight, touch, hearing and even smell. Desired output is often not measured but judged. Automatically controlling finishing processes in the same way as human operators control them would be difficult to achieve in practice. Other methods of automation do work.

Scores of robots are now deployed in U.S. factories finishing a wide variety of parts[2]. There are many factors which have contributed to the growing commercial success of robotic finishing. The dominant success factor is the pioneering effort by 3M and others to develop the compliant abrasive finishing processes. The key is compliance between part and tool, critical because it reduces variation of tool force, a dominant process control parameter. Even slight variations on the order of a few thousandths of an inch in the relative location of the robot's commanded tool point path and the actual contact point will result in large changes in tool force reactions if the system is too rigid. Normally occurring tool wear, part-to-part geometric variation, manipulator kinematic error, and set-up errors drive rigid abrasive processes out of control. There is a wide variety of abrasive tools available which are by the nature of their material composition, naturally compliant. Also, several constant force devices are commercially available which provide added compliance in one direction, usually normal to the surface of the part. With constant force end effectors, more aggressive abrasive tools can be used with robots.

The Need for Automatic Programming

A major barrier to further finishing robot deployment is system programming. Generating the robot motion control sequence is a major problem in manufacturing operations which produce parts in a variety of complex shapes. One manifestation of the programming problem occurs when a variety of complex shape parts must be finished. Each part type requires only one program which is executed repeatedly for its specific part type, but the programs have a large number of taught points. An example is the polishing of large asymmetric shaped aircraft skin panels. One manufacturer determined that manual teaching of robotic polishing programs was to too costly when their part mix approached eight to ten part types.

In cases requiring a unique motion sequence to finish each individual part, manual methods must be used because of the lack of practical programming methods. For example in aircraft remanufacturing, aerodynamic surfaces dented in normal service are routinely repaired by applying filler materials which are ground to shape by hand. Since the location and extent of this type of damage is random, each part requires a unique grinding program. Unique plans are commonly required for finishing turbine blades and propellers. The location and amount of excess material left by machining and forging is not entirely predictable. An automatic programming system which can be integrated with commercial robotic surface finishing equipment is needed.

Automatic finishing with compliant abrasives does not require automatic process control and planning. Planning is uncomplicated for many applications because process settings such as abrasive type, tool cant angle, tool force, tool feed, and tool speed remain constant for most or all of the individual job. Path planning and the subsequent trajectory generation still require too much manual effort for many robotic applications.

The approach taken here to developing automatic programming for robotic finishing is to first automate the path and trajectory generation

while depending on the human operator for higher level process planning, supervision and control Interviews with end user companies revealed that using CAD design data as a basis for modeling surfaces for path generation purposes would be impractical. So reverse engineering the surface to create a geometric model was included as a system requirement. For reasons of maximizing practical application success, commercial products are used to the greatest extent possible. Furthermore, overall system cost is minimized so that smaller companies will more readily be able to acquire the technology. This approach serves the objective of providing valuable off-the-shelf programming automation while providing a basis for the development of intelligent and more fully automatic processing.

System Equipment

A personal computer(PC 486-50mhz) and Faro Metrecom Coordinate Measuring Machine(CMM) are used with MS-DOS and AutoCAD as the platform for the automatic path and trajectory programming system. The Faro Metrecom CMM is a 6 DOF spatial digitizer interfaced to AutoCAD. AutoCAD's 3D modelling capability allows creation of, access to, and modification of geometric objects through AutoLISP and C. Customization of AutoCAD's environment through menu building using AutoLISP and Macros makes the objects accessible and easy to manipulate. The Fanuc model S-700 is a 6 DOF serial link manipulator with multi-tasking capabilities using the R-J controller. The R-J controller is off-line programmable. The Karel language programs can be written and compiled on a PC workstation. The R-J controller interfaces to a PC from which compiled robot control code can be accessed. The source code is written in ASCII format in Karel, then it is formatted and translated to executable code using the Karel translator program supplied by Fanuc. A server supplied by Fanuc is used to download the translated code to the controller. The PushCorp constant force end effector which was developed as a compliant abrasive end effector for surface finishing has a dedicated controller with interface for the R-J controller. The end effector mounts directly onto the faceplate of the robot with the axis of rotation of the grinding disk parallel to that of the faceplate.

Setup Procedure

Typically surface models for off-line programming applications are generated using constructive solid geometry (CSG). In this case the geometry will be recreated in a CAD environment from digitized surface location data in a process called "reverse engineering". Before digitizing the surface a common coordinate system is established between the CMM and the robot.

The robot can relate a tool frame to a base frame and provides several programmable frames of both types as well as one predefined frame of each type. Since it is the location and orientation of the grinding disk that is of importance a tool frame is taught at the center of the disk with the CFD positioned at the midpoint of its compliance stroke. There are different methods of teaching a tool frame to this robot through the teach pendent menu, and the method used here is the 3 point method. The desired tool point is placed at some fixed point in space which is taught three times with three completely different end effector orientations. This point on the tool can then be moved in space to any point within the robot's workspace relative to the base coordinate system or a user defined coordinate system.

Next, a user defined coordinate system is defined using a 3 point method for the robot and the CMM simultaneously. First a new origin is taught to the robot, and then the same origin is taught to the CMM by placing its tool point at the tool point of the robot and recording it. The robot is then moved parallel to the X axis of the base coordinate system several inches away and taught a point on the new X axis which is followed by teaching the CMM the same point to define it's X axis. Finally a point is taught to both devices on the positive part of the XY plane. The CMM automatically translates to this new system, and the robot can be instructed to do so through the teach pendent or through off-line programming tools.

This entire process can be done in 10 minutes and is only necessary for initialization or when the set-up is disturbed. Once the two devices share a common coordinate system the Faro CMM can be used to generate a CAD model of the surface to be finished, and the coordinates on the surface will be the same with respect to the CMM or the Fanuc Robot.

Surface Model

The Faro/AutoCAD interface allows the user to move the endpoint of the CMM digitizer through some trajectory in 3D space and generate a 3dpolyline entity with complex entity nodes at a predetermined interval. By keeping the endpoint of the digitizer in contact with a contoured surface, a 3dpolyline that lies on that surface is created.

AutoCAD has the capability to generate a Coons Patch based on four bounding lines in space that are connected at their endpoints. The number of facets, or mesh segments are programmed using the variables surftab1 and surftab2. The importance of this is that facets will be used to determine the path node frames of the end effector trajectory. Mesh density in the direction of travel will affect the smoothness and accuracy of the motion while mesh density in the direction perpendicular to the direction of travel controls the spacing between passes.

An AutoLISP program is activated through the customized pull down menu at which time the Faro 3dpolyline programs are executed and request the user to drag the digitizer over the desired boundary. This operator defined boundary encloses the region on the surface to be finished. The program then automatically performs the required modifications of the polylines to ensure that they are connected at their endpoints. The result is four 3D lines in space connected at their endpoints forming the bounding edges of the Coons Patch surface of m by n surface facets. Upon completion of generating the four boundary lines and assigning the entity data lists to variable names, an interpolated surface is automatically generated using the Edgesurf command. This command requires selection of the four existing bounding edges and is answered by the program with information from the stored entities. The direction of travel is requested upon completion of the mesh generation process and can be in the m or n directions.

Extracting Tool Pose Information from the Surface Model

The problem of determining robot tool poses from surface data involves several steps. A brief outline is helpful in describing the work that has been automated.

1. Create or access an existing surface

A function is implemented to number the nodes of the mesh and add this integer data to the actual entity data which was created by AutoCAD. This is done to simplify the development of an algorithm to access individual nodes and their immediate neighbors on the surface.

2. Determine a unit surface normal vector

This is done by averaging several normal vectors in the region. Each node P being considered as a path node has four immediate neighbors including two in the direction of travel and one each on either side perpendicular to that direction. These will be referred to as P_1, P_2, P_3, and P_4. The cross product between vectors originating at P and terminating at two of the neighboring nodes yields an approximate normal vector to the surface at that point. An algorithm is implemented that performs this same operation for four quadrants of the irregular quadrilateral formed by these points with P as the origin for each vector which results in four approximate surface normal vectors at the node P. These four approximate surface normals are then averaged and normalized to get an approximate unit surface normal at the node P. This procedure is performed for each node on the surface excluding those on the bounding edges.

3. Define an angle of attack

When the program that calculates and draws the surface normals is complete, another function is automatically called that requests a cant angle[3]for the tool. This angle is the pitch angle of the tool that will be used for the finishing process. The user can type a number in degrees at the command line after which the information is used to calculate tool frames at each node of the path.

The calculation of tool frames is based on the transformations found in Craig's Text[4]. The result is shown graphically on the screen in the form of a coordinate system at each node in the orientation that the end effector will be in at that node in order to maintain the desired cant angle with respect to the surface and the direction of travel.

4. Generate the Karel Program

At this point the tool frame information must be expressed in the language understood by the

robot. The user is prompted for a filename which will be used to store a Karel program in the form of *fname*.kl and a data file in the form of *fname1*.kl , *fname2*.kl and so on for each branch. Another program then implements an algorithm developed to determine which nodes belong to a branch of the path, and builds the path as a list containing branches which them selves contain lists. A branch list is of the form (1 5 6 7 8) where the first number is the branch number and determines it's order of execution among branches, which is followed by the node numbers which define which nodes will be traveled to in this branch and their order. Each branch begins at a boundary and follows the mesh in the previously chosen direction until it reaches the opposite boundary and will include all nodes along the way that are not actually on the boundary. The initial and final nodes are extrapolated linearly based on the two first or last nodes respectively with a user defined offset in the direction away from the part in order to allow a smooth approach or departure to or from the surface. These extrapolated nodes will not show up in the path data list and are calculated just before being written to the data files

The final path variable will look something like this :

(pa1 (1 7 8 9 10 11) (2 18 17 16 15 14) (3 20 21 22 23 24))

and will be used to write the path data files. The path data file contains only information that is relevant to the Robot controller and it's path data structure.

Each branch of the data is written to a data file in a specific format that is shown in the following which was taken from an actual data file.

```
536.6210 103.9143 -208.2082 175.8836 3.6680 -9.1591
479.4422 114.2409 -259.2189 175.8836 3.6680 -9.1591
422.2633 124.5675 -259.4296 178.8781 -5.6877 -11.3613
366.8345 136.8362 -270.6763 -177.4433 -14.8075 -11.1896
312.7229 146.2052 -288.9642 -174.9756 -19.1880 -11.4856
259.5330 158.6093 -308.7990 -173.1321 -20.1050 -13.6317
206.3431 171.0134 -277.8339 -174.9756 -19.1880 -11.4856
```

This data file represents branch 1 from the path list and each line in the file represents the data for X, Y, Z, and the Euler angles required to define position and orientation of the end effector for a particular point on the surface. There are several types of data formats in Karel including the type *path*. A path type variable is a structure which can contain several other specific types of data including position data in the form shown above. Several positions and orientations can be stored in a single path variable, and then used in a move_along statement which instructs the robot to move it's end effector through those points sequentially. There are also motion control statements that can be used in conjunction with the move_along statement that affect speed and acceleration between nodes.

A LISP program is then called that writes a Karel program to be translated and sent to the controller. The Karel program is then loaded into the controller and executed at which time it reads the data files and builds path variables in a routine called build_path. Each branch of the path is built before being used in motion statements that cause the robot to follow them. The move_along statements are executed sequentially and the tool moves across the surface as planned.

The actual grinding process can be initiated either through the Karel program by way of an option included in the custom menu, or by using the teach pendant.

Conclusions

A practical method of automatically generating tool point paths and robot trajectories for surface finishing has been developed and implemented using off-the-shelf technology components. This method is expected to only increase average robotic finishing system costs on the order of 5%-10% of total cost. High level process planning and supervision remain under human operator control.

More research is needed to develop practical process planning and control methods to supplement the operators process knowledge so that new applications requiring new processing recipes can be quickly implemented. Also sensor applications must be developed which augment the operators ability to assess surface condition and geometry.

A longer range objective is to develop automatic finishing as a smart processes which can be driven using feature based CAD data from intelligent high level supervision and control systems. Automatic processing capability must be broadened beyond the compliant abrasive process methods used today.

Efforts to build the manufacturing information technology infrastructure to support intelligent agile production continue, but the automatic planning and control of most shop floor processes including finishing is not adequately understood. Furthermore, the application of sensor technology needed for automatic observation to support finishing process control is not understood. Much empirical work will be done to discover practical process planning, control, and sensing methods for automatic finishing. The automatic path generation capability developed through this research will immediately improve application development productivity.

Acknowledgments

The research described in this paper was conducted in part with funding from Vought Aircraft Co. Through the ARRI membership program in ARRI's finishing laboratory. Vought also supplied complex shaped fuselage skin and fixturing. Tim Graf of 3M provided valuable information on end-user path programming needs. 3M's Abrasive Products Division donated all abrasive tools. The FANUC S-700 robot was consigned by FANUC Robotics North America, Inc. PUSHCORP. Inc. of Dallas, TX supplied the force controlled end effector and donated valuable technical support. FARO consigned the CMM.

References

[1]National Center for Manufacturing Sciences, Automatic Inspection Systems for Edge and Surface Finishing, Technical Report NCMS-88-MPM-9, April 1989.

[2]Tim Graf. Deburring, Finishing, and Grinding Using Robots and Fixed Automation: Methods and Applications. In *Proceedings International Robots & Vision Automation Conferenc*, pages 20.1 - 20.22, 1993.

[3]James C. Davis, Path Planning Methods for Robotic Grinding of Complex Contoured Parts.MS thesis , University of Texas at Arlington, 1993

[4]John Craig, *Introduction to Robotics* Addison-Wesley Publishing Company, 1986.