Use of a PC-Based Open-Architecture Control System in Engineering Education

R. Bryan Greenway and Qiyi G. Zhang
University of Kansas

Abstract

This paper describes the use of a PC-Based Open-Architecture Control System (OACS) for training engineers in the programming and control of manufacturing systems such as machine tools and industrial robots. In this investigation, PCs with motion control cards were integrated with a milling machine and a robot. Projects based on the use of OACS in engineering education include: 1)cutting force measurement and control; and 2)motion command generation. This paper explains the basic structure and characteristics of an OACS as well as engineering principles and laboratory experiments which can be demonstrated and incorporated.

Introduction

Progress in manufacturing automation relies on the development of CNC machine tools, robotics and other industrial automation devices. While the structure of machine tools and robotics evolved relatively slow, the control algorithms and methodologies developed rather rapidly. Starting from the later 80s and early 90s, Open-Architecture Control Systems for machine tools, robotics and other automation manufacturing devices have been investigated worldwide. In the United States, several national laboratories, major industrial companies and university laboratories, including the KU CIM lab, have been involved. While some developed their own hardware, most used commercial motion controllers (like PMAC from Delta Tau Data System, PCX from Motion Engineering) incorporated in a personal computer. More detail information about the OACS research can be seen in [1]-[4].

Compared with the conventional closed type of controllers, the OACS have various advantages in flexibility, extendibility, and longevity. From an educational and research point of view, the OACS provide more access to the control system. Since the architecture is open it is easier to experiment with other aspects of motion control such as servo filter design, motion command generation, error compensation, and integration with other manufacturing systems. With an OACS testbed, a wide variety of research aimed to improve the manufacturing capability and quality has been proposed and investigated.

This paper explains the basic structure and characteristics of an OACS as well as engineering principles and laboratory experiments which can be demonstrated and incorporated. In the second section of this paper, the challenge of OACS and basic concepts of an OACS as well as its development and perspectives are discussed. In Section three, the research work on OAS conducted at the KU CIM lab are summarized. Projects that involve using of OACS in engineering education are introduced in Section four. Finally, Section five gives the discussion and conclusion.

The Challenge of OACS

Conventional machine tool and robot controllers are based on proprietary designs that prevent various user, machine tool and robot builders, and system integrators from implementing creative production improvements in a truly cost effective manner. Today's controllers are single company products or product lines, and the concept of an open system is often not evident, even among and within the product lines offered by individual vendors. These controllers have been developed as monolithic systems, requiring a large investment by developers, that in some cases exceeds the returns generated over a relatively short product life.

The OACS, on the other hand, allow individual companies to add their own unique capabilities to controllers while remaining compatible with systems supplied by other vendors. The open architecture enables suppliers to customize control systems to fit users' needs more closely, while the industry benefits by adopting standards from the best current commercial practices. By adopting an open architecture, an OACS will contain packaged, proven technology that is widely applied and is likely to remain a standard in the marketplace.

An OACS support a wide range of processing and discrete part manufacturing applications, including machine tools of all type, robot, electronic assembly, material handling devices, inspection devices, and virtually all types of equipment in both manned and unmanned environments, and networked and stand-alone configuration']].

In the last five years, demand has increased gradually for open-architecture control systems in the machine tool, robotics, and general automation industries. This demand has been fueled by ever-increasing need to cut automation costs and implementation times. This demand has not been satisfied because few open-architecture controllers available. Now, some controller venders are trying to commercialize their own OACS, which promises to make the demand for open architecture explode and change the face of automation industries.

Although there has been much discussion about what features constitute an OACS, no clear definition has yet emerged. But four criteria seem to be key to an open-architecture control system[4]:

1. Open systems must be based on standard computing architectures, such as VME or ISA bus, and standard processors, such as Motorola 680x0, PowerPC or Intel Ix86/Pentium-based systems;
2. They must be based on standard operating systems, such as MS DOS, Unix or WindowsNT;
3. They must be programmable in standard programming language, such as Microsoft's Visual Basic and Visual C++ (for Windows application) or C/C++ (for DOS application);
4. The control software must be open and extendible to allow original equipment manufacturers or end-users to integrate of custom control algorithms such as mechanism kinematics, force control, motion and I/O control.

The advantages of implementing open-architecture control systems are clear. Because of mass produced hardware developed for the general computing industry, open-architecture controllers can be provide at significantly lower hardware, software, and maintenance cost than their proprietary counterparts.

Because the programming and operating environments of the control system are based on standard interfaces, the training costs associated with continuously learning proprietary systems are eliminated. Standard languages, such as C/C++, reduce the learning curve for new engineers and commercially available software productivity tools, such as Graphical User Interface builders and generators, can significantly reduce the time to get a new product to market.

Standard hardware and operating systems means that automation users can choose the best price/performance hardware and software solution to meet their specific needs. This allows fast and easy incremental gain in system performance. For example, if a user wishes to obtain newly supported features from proprietary vendors, they generally must replace the controller. However, when based on standard hardware and operating systems, updating system performance on an open-architecture platform generally involves simply installing new updated software.

Open-Architecture Control Investigation in KU CIM Lab

- In the KU CIM lab, open-architecture control research began in the summer of 1994 when an Intel 486 PC with a PMAC motion control card was integrated with a Hurco milling machine. Hardware connection, including tool changer and spindle speed control, was completed. Basic software was developed to run the system and a force table was integrated with the control system. Cutting force measurements for a master's project were conducted by using this system. Also, a 90 MHz Pentium personal computer and an Motion Engineering's 8-axis PCX motion controller were utilized as the controlling unit to construct the open-architecture control on an American Robot (seen in Figure 1). The hardware connection was completed and an object-oriented program has been developed and partially implemented for the on-board jogging, system calibration, forward and inverse kinematics transfer, and point-to-point motion. In the next section, the projects that use the OACS for training engineering students will be discussed in more detail.

Use of OACS in Engineering Education

Using the KU CIM OACS platform, several projects have aimed to familiarize students with the characteristics of the open-architecture control algorithms and to implement advanced experiments with other aspects of CNC control, including cutting force measurement and control, advanced software programming and integration, and robot parametric curve trajectory generation.

1) Cutting force measurement and control

As mentioned before, a six-dimensional force table was integrated into the OACS for a milling machine from the beginning of its implementation. This investigation turned out a master's thesis and real-time cutting force data was used in another. The research includes cutting force measurement and analysis, cutting force locus display and control, and cutting force and feed rate integration

The cutting forces are vital in machining processes. Using real-time cutting force measurement, one can predict and prevent tool breakage, incorrect workpiece placement and instantaneous cutting conditions. By use of the OACS, the spindle speed can be more precisely controlled, which makes the measurement for different machining parameters much easier. Additionally, the cutting force measurement can be integrated into machining feed rate control. The PC-based PMAC motion controller has the ability to vary its feed rate during motion. So, using the cutting force measurement system, the machining parameters can be controlled in real-time. For example, if for some reason (change of workpiece shape or material), the cutting force increases, the feed rate or spindle speed can be slowed down or speeded up to keep and maintain optimal machining conditions therefore to improve the machining quality.

2) Motion command generation

For conventional CNC machines, including machine tools and industrial robotics, tool paths or end effecter trajectories that are not linear or circular arcs are typically piecewise approximated with line segments. This approximation produces error between the desired path and the commanded path, which is termed chordal deviation. Non-uniform Rational B-spline (NURBS) have long been favored in CAD system because they offer an exact, uniform representation of a wide variety of common curves such as circles, parabolic, ellipses, hyperbolas and other complex curves. NURBS are also computationally efficient and allow great flexibility in defining free-form curves and surfaces. By basing the tool path or robot trajectory on NURBS, the position error generated by approximating curves with straight line segments will theoretically be eliminated. Another benefit of using NURBS is that the amount of space occupied by part programs and the time taken to transfer them can be significantly reduced. Large sections of part programs often consist of many small linear move commands.

Each of these commands must be read by the controller and executed. In some cases, the feed rate of the machine is limited by the rate **at** which these line segments can be read. Additionally, these part programs consume large amounts of storage memory, and can take considerable time to download to the controller.

- To illustrate the problem, let us look at an example. In Figure 2(a), a typical second-order curve has been approximated with the line segment AB and CD. The controller will interpolate linearly between the points during the motion, generating points 1 to 6, which do not lie on the curve. This is the chordal deviation which **is** usually **a** parameter specified **to the** CAD/CAM system for **use** in determining the number of lines to use for the approximation. The smaller the acceptable chordal deviation, the more line segments used **to** approximate the curve and the larger the part program. In contrast, if the controller were be able to handle NURBS directly, the interpolated points would lie exactly on the second-order curve **(at the** accuracy of the machine), as shown in Figure 2(b). Also, only one motion instruction is required: **a single** NURBS move from A to C. The benefit **is a** more accurate fit of the motion to the curve with a reduction in size of the part program. The Open-Architecture Control System offers the possibility to develop this technique.

In robotics implementation, for example, the controller commands the six stepping motors to drive the six robot joints to realize the parametric trajectory in 6-D space (3-D position and 3-D orientation). Trajectory generation begins by first calculating the next command point along the curve using the equations derived in [5] and [8] (the orientation will be defined by another set of parametric functions); then translating this new point into the robot's six joint angles using inverse kinematics' formulas; finally the motion control card gives the next step command for each of the six axes simultaneously. All the calculations have to be done at the sampling rate. Conventional trajectory generation is realized by using point-to-point motion commands that come with the controller's software. However, for the parametric curve trajectory generation, the next point command has to be pass to the control loop by writing into the loop.

Concluding Remarks and Acknowledgments

Interest in open-architecture control systems is gaining momentum fueled by industrial demand. For educational institution, use of open-architecture control system in engineering education seems necessary and very beneficial. It will strengthen the theoretical and practical research in the area of manufacturing automation, efficiently spread the awareness of this developing technology, and prepare graduates of all levels for research and production needs. This paper introduced the practical use of OACS in engineering education. Several on-going projects were proposed and discussed.

The authors would like to thank Fred Proctor of the National Institute of Technology for his comments and email, Mike King of Allied Signal for his suggestion and input, and Carl Locke, Dean of the School of Engineering, for his financial support of our lab.

Reference

[1] B. M. Andderson, J. R. Cole and R. G. Holland, A Open Standard for Industrial Controller. ***Manufacturing*** *Review* Vol. 6, No. 3, September 1993.

[2] F. Proctor, B. Damazo, C. Yang and S. Frechette, Open Architectures for Machine Control. National Institute of Standard and Technology, Gaithersburg, Maryland 28099.

[3] T. Bailey, Y. Ruget, A. Spence and M. Elbestawi, Open-Architecture Controller for Die and Mold Machining,. ***Proceeding of the American Control Conference,*** Seattle, Washington, June 1995.

[4] Peter H. Manly, The Challenge of Open Architecture, ***Robot World,*** Summer 1995.

[5] J. Chou and D. C. Yang, Command Generation for Three-Axis CNC Machining. ***Journal of Engineering for*** *Industry,* August 1991.

[6] Y. Koren, C. C. Lo and M. Shpitalni, CNC Interpolators: Algorithm and Analysis. ***Manufacturing Science and Engineering,*** ASME 1993.

[7] M. Shpitalni, Y. Koren and C. C. Lo, Real-time Curve Interpolators. ***Computer-Aided Design,*** November 1994.

[8] C.K. Au and M. M. Yuen, United Approach to NURBS Curve Shape Modification. ***Computer-Aided Design,*** Vol. 27, No 2, 1995.

[9] Qiyi G. Zhang, Study on Parametric Curve Trajectory Generation for a Robot Using an Open-Architecture Controller, ***Ph.D. Research Proposal,*** The University of Kansas, November, 1995.

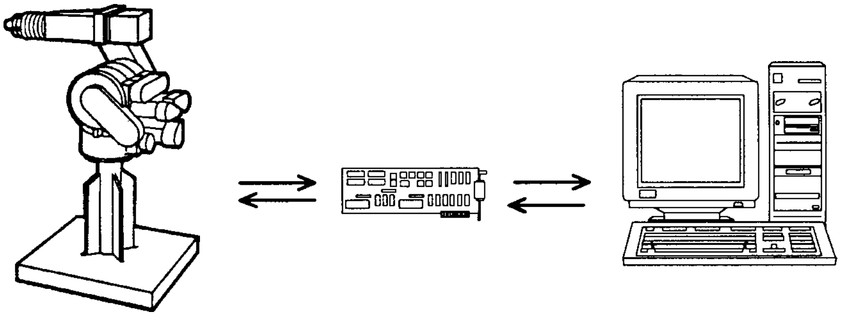

Figure 1. The setup of OACS for an industrial robot

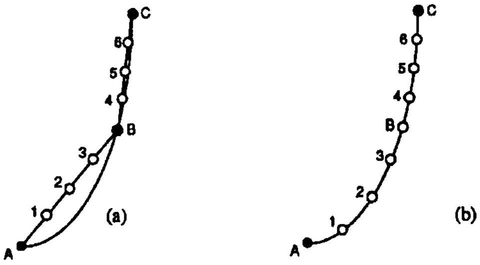

Figure 2. (a) Approximation of a curve with line segments, and b) Fit of a curve with NURBS.

R. BRYAN GREENWAY, JR.

Dr. Greenway is a Assistant Professor in the Department of Mechanical Engineering at the University of Kansas. He graduated from the University of Kansas in 1992 with a Ph.D. in Mechanical Engineering. He joined the faculty at the University of Kansas in 1991. His teaching responsibilities include design, manufacturing, and microprocessor application to mechanical systems.

QIYI GEORGE ZHANG

Qiyi G. Zhang is currently a Ph.D. student in mechanical engineering at the University of Kansas. He earned his M.S.M.E. from the University of Kansas in 1994. He worked as an instructor at North China Institute of Technology for three years. He has been working as a Student Teaching Assistant in the Department of Mechanical Engineering at the University of Kansas for four years.