

MACHINE DESIGN

JANUARY 24, 2002
www.machinedesign.com



A PENTON PUBLICATION
Periodicals
USPS 881 Approved Poly

Intelligent design/Motion systems

Edited by Miles Budimir

PCs take control

How PC-based, open architectures are changing machine control.

PC-based motion-control architectures used to look about the same. The PC acted as a man-machine interface (MMI) and application specific logic was split between the PC and an intelligent-motion card. The card sent an analog signal representing either a velocity or torque command to an analog drive.

Now, motion-control systems are a lot more flexible as a result of dramatically improved processors, high bandwidth real-time networks such as IEEE-1394 (FireWire), and reliable PC operating systems.

PC-based controllers make it easy to communicate with MES systems and databases and integrate with third-party devices such as vision systems. A familiar user interface and good GUI tools speed development time because programmers don't have to learn a new interface.

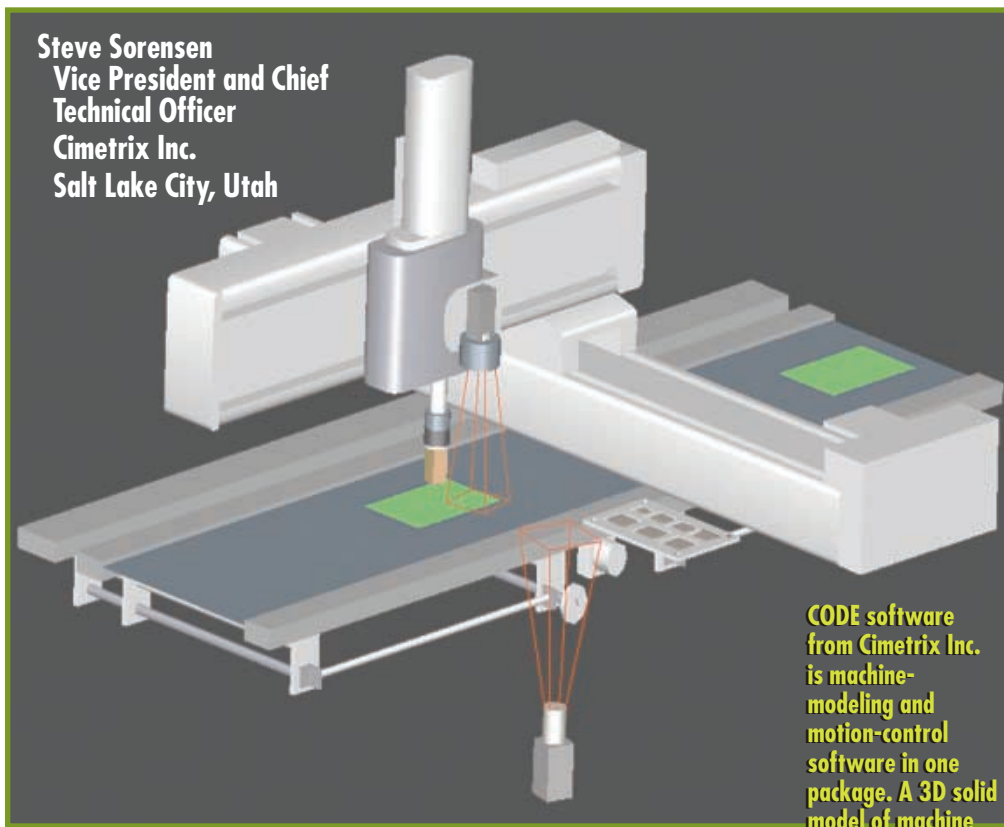
Windows NT or its derivatives (2000 or XP) are viable candidates for machine control. Windows' development and debug environments, coupled with easy integration to external sensors such as vision systems, makes Windows the platform of choice for application logic and path planning.

Real-time extensions such as VenturCom's RTX make possible real-time path planning. Adding a real-time kernel lets a PC generate functions all the way down to a velocity loop. Commutation and current loop closure may also be feasible in the near future.

NETWORKS

Future motion controllers may be gauged by how

Steve Sorensen
Vice President and Chief
Technical Officer
Cimetrix Inc.
Salt Lake City, Utah



CODE software from Cimetrix Inc. is machine-modeling and motion-control software in one package. A 3D solid model of machine kinematics can be simulated and control code generated and tested in the same user environment. Here, a vision-guided robot from Hirata, Indianapolis, is modeled using the CODE simulation program.

real-time data moves between the PC and the drive. Currently, there exist two basic options: standard fieldbuses or general-purpose networks.

Standard fieldbuses including CAN, CAN Open, Profibus, DeviceNet, and Sercos all can handle low-bandwidth communications, and are valid choices between application logic and the path planner. Of these, only Sercos is specifically designed for motion control. It has the determinism and bandwidth to be used below a coarse interpolator.

General-purpose networks such as Ethernet, USB-2, and FireWire are another option. Of the three, FireWire is the most promising. A key feature of FireWire is that it has both an isochronous channel for guaranteed real-time delivery and an asynchronous channel for nontime-critical data.

The current FireWire standard has some limitations including a 4.5-meter maximum cable length and noise sensitivity. However, this has been solved by the latest draft standard called IEEE-1394b. The new spec for copper, plastic fiber, glass fiber, and Cat5 cables extends

distance to 100 meters and pushes speeds to 3.2 Gbits/sec. Scalability and low-cost connection make FireWire ideal for motion control, save a lack of a standard software protocol. If this issue is addressed, FireWire could dominate motion-control networks.

GOODBYE MOTION CARD

Traditional PC-based motion controls use a general-purpose intelligent-motion card. These cards typically control multiple axes and run either a simple executive or a small embedded real-time operating system. They

A modern servocontrol loop

Each block represents a specific control function along with the bandwidth and real-time requirements of the function and its interfaces. Note that the rate of cyclical calculations increases from left to right. Conversely, from right to left, algorithms become more complex and specialized.

The path planner generates the motion profile and is the most complex of the general-purpose function blocks. Path planners tend to be specialized for a particular class of applications. For example, a path planner for an articulated robot following Cartesian paths is far more complex than one moving an X-Y-Z stage from point to point. However, when you add vision and high accuracy requirements to an X-Y-Z stage, the path planner must correct for machine imperfections.

Path planners are categorized into two classes: static and dynamic. Static path planners need only execute once for each move. Because the path is completely known ahead of time, the path planner can run well

ahead of the current machine position, eliminating the need for hard real-time control. Dynamic path planners, on the other hand, continuously reevaluate the path based on real-time feedback from external sensors. Some dynamic path planners reevaluate the path at a fixed frequency (e.g., high-speed conveyor tracking), in which case hard real-time needs are quite high. Or they may reevaluate the path on an event-driven basis (e.g., change end point on the fly, or blend with a late arriving move), lessening real-time requirements.

Another way to categorize paths is point-to-point or continuous path. Point-to-point moves are always done in joint space, and as such, may not require a coarse interpolator. Instead, the path planner interfaces directly to a fine interpolator. For continuous paths, the motion profile is typically specified in Cartesian space. Inverse kinematics, and occasionally inverse dynamics, must be called at a fixed frequency to convert Cartesian path points into joint positions and to

find optimum servogains and feedforward terms. For some systems, it may not be necessary or even feasible to perform these complex calculations at the same rate as the position loop is closed. Here, a coarse interpolator is used. The Cartesian points convert to joint space typically at frequencies ranging from 1 kHz to 100 Hz.

A much simpler interpolation scheme then subinterpolates individual axes into position setpoints at frequencies as high as 20 kHz. A major advantage of a coarse interpolator is that it provides a natural interface point to distribute control. The coarse interpolator runs on the PC, while a motion card or intelligent drive runs the fine interpolator, sometimes even buffering several moves. This way, a coarse interpolator can handle both point-to-point and continuous-path moves.

The output of the fine interpolator is a position setpoint. Typical position-setpoint generation takes place every 1 to 2 kHz, but some extremely low inertia systems do this as fast as 20 kHz.

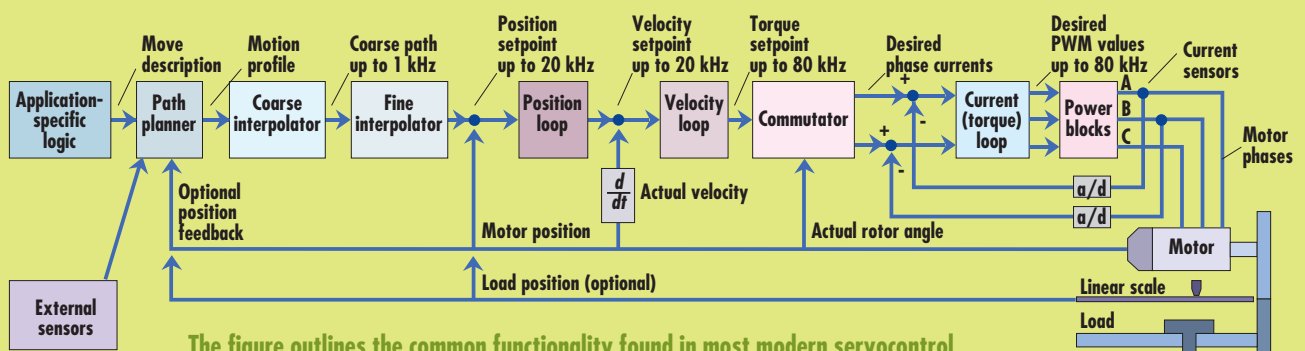
The position loop output is a velocity setpoint and the output of the velocity loop is a torque setpoint. Some control systems combine the position and velocity loop so that the two are indistinguishable. PID control laws with feedforward terms are standard, but systems with complex dynamics may require more sophisticated algorithms such as computed torque.

Commutation has migrated from simple Hall-effect sensor circuitry to being calculated from the sensed angle of the rotor (sinusoidal commutation). Commutation was traditionally done in the drive, though it is now fairly common to perform commutation in a motion card.

Current loop closure is the final step in controlling a servomotor. This is traditionally done in the drive, but now motion cards can send PWM signals directly to the power blocks.

The power blocks do virtually no computations. However, when the current loop is closed in the motion card, the PWM drives may still require logic for safety and diagnostic functions.

Servocontrol-system block diagram



The figure outlines the common functionality found in most modern servocontrol systems. Each block represents a specific function with well-defined interfaces.

perform all common control functions except for power blocks. They are generally weak for application logic and complex path planning, but particularly if inverse kinematics (converting from Cartesian space to axis space) or dynamics (computing motor torques for a given position, velocity, and acceleration) are required.

Intelligent drives, in contrast, provide a power amplifier for one or more axes. Drives generally locate outside of the PC backplane and are either hard-wired to a controller (analog drives) or connected via a real-time network (digital drives). Digital drives are becoming increasingly intelligent and capable of performing many control functions, but they are often restricted to a single axis.

Two alternative architectures look promising. Both are quite different from the traditional PC-to-motion-card-to-drive approach.

The first is the PC-based controller. It moves all of the functions traditionally found in a motion card onto the PC. The PC interfaces directly to feedback sensors and outputs velocity or torque commands directly to

the drives via a low cost I/O interface card. This architecture is best suited for interfacing with analog velocity or torque-mode drives. Cost is the primary advantage of this architecture over a motion card. Another advantage is reduced complexity. Because most intelligence resides on the PC, many of the complexities inherent to a distributed system are avoided. Connecting to external sensors and nontraditional feedback devices is also simplified.

The primary weakness of PC-based controls is the stability of Windows and the need for real-time extensions. However, stability can be addressed by qualifying all of the PC adapter cards and drivers used in the system. Redundant checks, such as blue screen detection, proper shutdown on the real-time extension side, and a watchdog timer that triggers a hard e-stop on the interface card, can help prevent the dreaded blue screen.

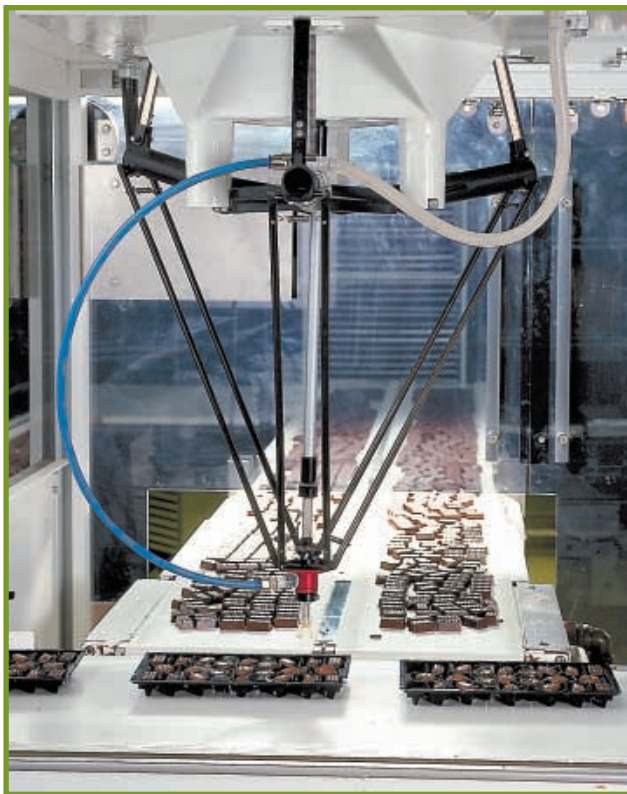
The second drive architecture is based on a standard network. This architecture has no motion-control-specific PC hardware. A standard FireWire (or other) network adapter connects a PC to one or more intelligent drives.

This architecture replaces the hundreds of wires needed to connect drives to a traditional motion card with a single network connection. This lets the PC be packaged into a small size (e.g., a handheld pendant). Using a standard network adapter nearly eliminates motion-control-specific hardware costs on the PC side. But increasingly intelligent drives cost more and may offset this cost savings. Moreover, intelligent drives that include application logic are harder to program and debug. Providing good tools to overcome this falls on the drive manufacturers. ■



Siemens SiPlace SMT machine uses a PC for kinematics, calibration, and machine-vision coordination. The kinematics involves not only the ideal inverse kinematics, but incorporates all of the necessary corrections for mechanical inaccuracies in the machine and rigid body, skew, and scale errors in the parts geometry.

A SIG Pack Delta robot does high-speed conveyor tracking with Cimatrix CODE. In this case, the path is dynamically changing. The path planner and coarse interpolator are located on the PC. Real-time extensions are required because a dynamically changing path cannot be buffered without loss of responsiveness and hence accuracy.



WE WANT YOUR FEEDBACK.

Did you find this material interesting? Circle 829

Do you want more information of this type?

Circle 830

Comment via e-mail to mdeditor@penton.com

What related topics would you like to see covered? What additional information on this topic would you find useful?