Motion Controllers, Programming, and System Design Basics

Nate Holmes
Group Manager, Motion R&D
National Instruments
Course Goals

- Understand the following:
  - Motion Controller Components and Responsibilities
  - Motion Control Types and Common Architectures
  - Motion Control Programming and SW Approaches
  - Integrating Motion into a Larger Application
Motion Controller Components and Responsibilities
Motion Controller Responsibilities

- Supervisory Control
- Trajectory Generation
- Control Loops (depending on control mode)
- Motion I/O
Motion Controller Responsibilities
Supervisory Control

- Initialization
- Command sequencing and coordination
- Event handling
- Fault detection (limit monitoring, e-stop, drive faults, etc...)

- Must run deterministically
- Configured by user application through provided API
• Generates setpoints of a motion path defined by the API and move constraints
  • Position (most common)
  • Velocity (usually trying to maintain, or switch between setpoints)
  • Torque (Examples: lifecycle/quality testing, web tensioning, etc...)

• Must support axes coordination and multidimensional profiles. May take into account the way that the axes are coupled and calculate inverse kinematics
Interpolation

- Accepts setpoints from the trajectory generator and interpolates additional points between them for smoother control.
- Resulting points are used by the control loop
- Different methods will produce noticeably different profiles
- Typically performed in hardware for speed of operation / determinism

![Cubic B Spline](image1)
![Catmull-Rom](image2)
![Linear Interpolation](image3)
Motion Controller I/O

- Input
  - Commanded setpoint (Ex: Position)
  - Feedback signal(s)
  - Limit and home switches
  - Drive status (Ex: fault present?)
  - Position capture (also called touchprobe, store feedback position at event occurrence)

- Output
  - Drive enable
  - Drive command
  - Position compare (also called breakpoints, position trigger output)
Why Provide Feedback to the Controller?

- Close control loops as necessary
- Allows for controlling or synchronizing other devices with motion
- More complex feedback schemes like dual encoder to compensate for lost motion due to backlash, compliance, and other sources of error in the mechanical system
What Executes Where?

Application Software

Motion controller

Drive

Motor

Mechanicals

Feedback Device
What Executes Where?

Motion

- User Program
- Supervisory Control
- Trajectory Generator
- Interpolation
- Position Loop
- Velocity Loop
- Torque Loop
- Motion I/O

Motion I/O

Supervisory Control

Trajectory Generator

Interpolation

Position Loop

Velocity Loop

Torque Loop

Motion I/O

Application Software

Motion controller

Drive

Motor

Mechanicals

Feedback Device

?
Motion Control Types and Common Architectures
Motion Control Architectures

- **User Program**
- **Supervisory Control**
- **Trajectory Generator**
- **Interpolation**
- **Position Loop** (1-100 kHz)
- **Velocity Loop** (4-100 kHz)
- **Torque Loop** (20-100 kHz)
- **Motion I/O**

### PC
- **Plug-In Motion Controller**
- **Motion Drive**
  - Analog ±10V or Step/Dir

### PLC or PAC
- **Smart Drive**
  - Industrial Comm (EtherCAT)

### PC
- **Integrated Motion System**
  - Serial or USB
Plug-In Motion Controller

- Ease of use – programming environment can run on the PC that does motion
- Quick prototyping and testing because there is no deployment when developing and using the same PC
- Supervisory control and trajectory generation can be deterministic – card can be running an RTOS
- May be less rugged
- Form factor is often larger than a real-time target
- PC often running a non-deterministic OS (Windows, Linux desktop)
PLC + Smart Drive

- PLC may be already part of the system
- Can be programmed with IEC 61131-3 standard languages
- PLC may be controlling other I/O which eases coordination with motion
- Smart drive takes on some of the motion controller responsibilities
- Motion controller, drive, stage(s) as part of the same package
- Easy to set up and integrate (most commonly seen in lab systems)
- Can be controlled directly from PC or HMI with USB, Ethernet, Serial
- Limited functionality for synchronization
- Non-deterministic commands and response to I/O
- Purpose built – limited flexibility
“Softmotion”

· Term has been around for a while in industry and used by different companies to mean slightly different things.

· In general, it means turning a general purpose computer/controller into a motion controller through the addition of motion specific software (toolkit, module, etc...)
  · Motion on a PLC (Ex: CODESYS Motion + CNC)
  · Motion on a Windows PC (Ex: Kingstar Soft Motion)
  · Motion on an Embedded RT Controller (Ex: NI LabVIEW SoftMotion Module)
Controller / Drive Communication (Motion Fieldbuses)

- Direct signal
  - +10V Analog
  - Step/Direction
  - P-Command (position command)

- Deterministic Ethernet based digital buses
  - EtherCAT
  - CanOPEN
  - Profibus
  - PowerLink
  - SercosIII
  - Etc...

- Non-Deterministic buses
  - TCP/IP (Ethernet)
  - Modbus
  - USB
  - Serial

Plug-In Controllers
(still seen more with stepper, but some consider legacy)

PLCs, IPCs, Embedded Controllers
(factory automation, current trend, recommended for new designs)

Integrated Motion Systems
(Labs, Research)
Direct Smart Drive Commands
(commissioning, simple apps)
Motion Control Programming and Software Approaches
The Application Software

- Where you define what a motion system is supposed to do
  - Configuring the motion controller
    - Supervisory control settings
    - I/O configuration
    - Motion bus settings
    - Etc...
  - API (Application Programming Interface) to motion functions
  - Defining move types and constraints
  - Sequencer or state machine or other architecture to chain moves
  - Interactions with the rest of the system

- Depending on system architecture, sometimes you have the motion application software and the drive configuration software, and sometimes these are combined into the same tool
Standardization

- IEC-61131-3
- PLCOpen
- CiA 402
- ...
CANopen Drive Profile

- CiA (CAN in Automation) 402: a CANopen device profile for drives
  - (CiA 402, same as IEC 61800-7-201/301)
  - Standardize functional behavior for controllers and drives
  - Defines a drive state machine and communication methods
  - Defines motion profiles and homing modes

- Used for defining communication to many drives
  - CANopen drives
  - EtherCAT drives (CANopen over EtherCAT (CoE))

- Object Dictionary
- SDO – Service Data Object
- PDO – Process Data Object
Safety and Motion

- Machines and systems are inherently hazardous with potential to cause injuries to persons or damage to the environment.
- Machines and systems must be designed so that errors or defects will not create dangers.
- The interaction between machine and operator must be designed to ensure dangerous situations cannot occur.

- Definitions driven by standards...
  - EU requires machine manufacturers to carry out a pre-design hazard analysis and risk assessment (performance levels, risk graphs, classification into categories...) as described by the European Machinery Directive 2006/42/EC
  - Machinery safety standards IEC/EN 61508
    - ISO 13849 machine safety requirements
    - IEC 62061 safety level
    - And more (ANSI B11, NFPA 79-2012, etc...)
  - Lots of material available...
# Functions for Safe Motion

<table>
<thead>
<tr>
<th>Function</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>STO (Safe Torque Off)</td>
<td>STO interrupts the power supply to the motor safely. The motor has no torque.</td>
</tr>
<tr>
<td>SS1 (Safe Stop 1)</td>
<td>The drive is stopped by controlled braking. The power supply for the motor is interrupted safely and the motor generates no torque.</td>
</tr>
<tr>
<td>SS2 (Safe Stop 2)</td>
<td>The drive is stopped by controlled braking and then remains at a controlled standstill. The control functions of the drive remain.</td>
</tr>
<tr>
<td>SOS (Safe Operating Stop)</td>
<td>Monitors the stop position reached and triggers for deviations above the prescribed limits SS1. The control functions of the drive remain active.</td>
</tr>
<tr>
<td>SDI (Safe Direction)</td>
<td>The SDI function ensures that the drive can only move in a defined direction. In case of error SS1 is triggered.</td>
</tr>
<tr>
<td>SSR (Safe Speed Range) 1</td>
<td>Monitors the drive to maintain a defined speed range. In case of error SS1 is triggered.</td>
</tr>
<tr>
<td>SLS (Safe Limited Speed)</td>
<td>Monitors the drive to maintain a defined speed limit. In case of an error is triggered SS1.</td>
</tr>
<tr>
<td>SBC (Safe Brake Control), SBT</td>
<td>Controls external brakes. SBT (Safe Brake Test) (not standardized) Test function for external and internal brake motor holding brake.</td>
</tr>
<tr>
<td>SLP (Safe Limited Position)</td>
<td>Monitors the absolute position of the drive. If the predefined limit is reached, or if the brake torque is too small to stop the drive within the limit, SS1 is triggered.</td>
</tr>
<tr>
<td>SLI (Safe Limited Increments)</td>
<td>Monitors the relative position of the drive based on the current position when the function is triggered. SS1 is triggered when the specified limit is reached.</td>
</tr>
</tbody>
</table>
Motion Programming Possibilities

- **IEC 61131-3**
  - PLCopen Motion Blocks
  - **Examples:** Codesys, B&R, Rockwell, Beckoff, ABB, Siemens, etc...

- **Company Specific Language or Implementation**
  - **Examples:** GalilTools, Elmo EASII, LabVIEW SoftMotion Module, Kollmorgen Automation Suite

- **Generic IDE (Integrated Development Environment) with company specific driver**
  - **Examples:** Galil, Delta Tau, Elmo, Maxon, etc... libraries for various languages
The five IEC 61131-3 Programming languages

**Function Block Diagram (FBD)**

**Ladder Diagram (LD)**

**Sequential Flow Chart (SFC)**

**Instruction List (IL)**

**Structured Text (ST)**

```plaintext
VAR CONSTANT X : REAL := 53.8 ;
Z : REAL; END_VAR
VAR aFB, bFB : FB_type; END_VAR

bFB(A:=1, B:='OK');
Z := X - INT_TO_REAL (bFB.OUT1);
IF Z>57.0 THEN aFB(A:=0, B:="ERR"); ELSE aFB(A:=1, B:="Z is OK"); END_IF
```
Motion Controller SW Functionality

- Drive Enable/Disable/States
- Reference Moves
  - Homing
  - Limits
- Straight line moves
  - Blending
  - Superimposing
  - Contour
  - Stopping
- Multi-axis Moves
  - Gearing
  - Camming
  - Coordinate Spaces
    - Arc
    - Inverse Kinematics
- Position Capture (Touchprobe) /Position Compare
Drive Enable/Disable/States

- Programatically enable or disable different drives in the system
- Potentially move drives through different states in their state machine
  - Ex: DS 402 state machine for EtherCAT or CANopen drives
    - Not ready to switch on
    - Switch on disabled
    - Ready to switch on
    - Switched on
    - Op Enable
    - Quick Stop Active
    - Fault reaction active
    - Fault
Reference Moves

- Establishes axis position and reference point for all future moves
  - Not necessarily needed for systems that can persist position info through power cycles (multi-turn absolute encoders) or where position control is not important

- Usually part of initialization / startup routines, or after a recoverable fault

- Searches for a switch state change or encoder index according to commanded parameters such as:
  - Initial search direction, final search direction
  - Signal edge parameters to look for (rising or falling) and to stop on (forward or reverse)
  - Reference velocity and acceleration, approach velocity
Limit/Home Switches

- Used in reference moves
- Mechanical, Optical, Inductive, ...
- Monitored by supervisory control during operation
- When activated, motion controller/drive performs configured action
  - How to stop axis where limit was activated (drive behavior)
  - How rest of the system behaves (other axis, other functions)
- Motion Controller or Drive Configuration SW used to
  - Enable/Disable switches
  - Configure polarities
  - monitor status
- Can be combined with Software Limits defined by the controller
The following three parameters are required to create straight line moves:

- Target position
- Maximum velocity
- Acceleration/deceleration values
S-Curve Profile

- Used for smoother changes in the acceleration profile

- The fine control over the shape of these ramps is very useful for tailoring the performance of a motion trajectory based on the inertial and frictional forces, motor dynamics, and other mechanical limitations in motion systems

- Provides finer control that helps reduce the jerk and decreases unwanted vibrations

- Configurable via a “smoothing” factor, Accel. Jerk, and Decel. Jerk
Blending Moves

- Several velocity profiles superimposed during contiguous position moves
- Continuous motion takes precedence, so end points of middle moves may not be reached
Contour Moves

- Contoured moves are useful when you want to generate a trajectory that cannot be easily constructed from straight lines and arcs OR the trajectory is coming from a different part of the system or is being calculated on the fly.
- Defined by a table of points, can be dynamically updated
- Can be created for an axis or coordinate space
- Contour interval determines $dt$ between points
Following Error Limit

- Generates an error when the difference between the commanded position and actual position crosses a specified threshold
- Immediately disables the axis
- Useful safety net for a system that has become unstable or is otherwise uncontrollable
- User configurable
• The following are three ways to stop a system:
  • Halt: immediate, tries to hold position
  • Decelerate: applies a user-defined deceleration to the stop.
  • Kill: disable drive, torque is immediately removed from the motor, free spin to a stop
Electronic Gearing

- Synchronize the movement of one or more slave axes to a master device (trajectory of another axis, encoder, or ADC)

- Effect is similar to that of mechanical gearing, but more flexible in that you can change gear ratios, and you can do things like superimpose a move over a geared axis

- Gear ratio can be:
  - higher or lower than master (usually represented as a ratio, slave:master)
  - positive or negative (resulting in same or opposite direction as master)
  - updated on the fly (results in acceleration or deceleration of slave)
Electronic Camming

- Similar to gearing in that the move of an axis is proportional to that of the master, but according to a *cam profile* instead of a gear ratio. It’s nonlinear coordinated motion between two axes.

- Definition of the cam profile differs depending on application SW
  - Defining a table, importing from a file, graphical cam editor, etc...
  - Moving point to ensure boundary conditions and constraints are met, ensure continuity, and minimize jerk.

Gearing

```
\[ \text{Gear ratio} = \Delta \text{Slave} / \Delta \text{Master} \]
```

Camming

```
\[ \text{Master Cycle Begins to Repeat} \]
```

```
\[ \text{Ratio 1} \quad \text{Ratio 2} \quad \text{Ratio 3} \quad \text{Ratio 4} \quad \text{Ratio 1} \quad \text{Ratio 2} \]
```
Coordinate Spaces

- A logical grouping of axis, commonly representing a physical coordinate space ([X,Y] 2D) or ([X,Y,Z] 3D)
- A move can be performed on a coordinate space and defined in a way to make programming easier
  - Get from Pt. A \((X_1, Y_1, Z_1)\) to Pt. B \((X_2, Y_2, Z_2)\), what are the move constraints?
  - What about different paths?
Arc Moves

- Any vector space can be configured to run in a circular or helical path
- Start and travel angle determine orientation and size of move
- Radius determines the angular velocity \( w = \frac{v}{r} \)
Position Capture and Compare

• **Position Capture**
  • Records the current position when an external trigger signal is received
  • Can be used to correlate measurements to position data

• **Position Compare**
  • Outputs a trigger signal on a DO line when specific positions are reached
  • One use is to trigger events from other devices at specific positions
Integrating Motion in an Application
Motion is Usually a Subsystem

- Motion
- HMI
- I/O
- Vision
- Enterprise
- ...

Application or Machine
Integrating Motion Into an Application
Integrating Motion Into an Application

- **EtherCAT**
- **Ethernet**
- **To PC**
- **USB Hub**
- **Basler USB Cameras**
- **NI Integrated Steppers** (Stepper drive + motor + encoder in one package)
- **Kollmorgen AKD EtherCAT drive/AKM brushless servomotor axes**
- **cRIO-9068**
- **NI 9512 x2**
- **NI 9401 x2**
- **NI 9474**

**Electromagnets**

24 RGB LEDs per board controlled by FPGA custom protocol to TI LED driver chips.
Machine Builder Triangle of Challenges

Higher Throughput

Higher Quality

Lower Cost
What really makes a machine smart is the ability to tightly integrate multiple specialized subsystems to meet machine requirements.

![Diagram showing Advanced Machine Control](image)

**Motion Control Integration**
- High frequency measurements
- Signal processing and filtering
- High speed timing and triggering
- Custom or model based control
- Multi-axis coordinated moves
Building a Complete System

Motion
- User Program
- Supervisory Control
- Trajectory Generator
- Interpolation
- Position Loop: 1-100 kHz
- Velocity Loop: 4-100 kHz
- Torque Loop: 20-100 kHz
- Motion I/O

Other Machine Subsystems and Capabilities
- HMI
- Machine Vision
- Data Acquisition
- Process Control
- Control Design and Simulation
- Filtering and Signal Processing

PC
Real-Time Controller
Motion Drive

Supervisory Control
Trajectory Generator
Interpolation
Position Loop
Velocity Loop
Torque Loop
Motion I/O
Iterative Design Process

1. Analyze the mechanical system
2. Select motor/stage
3. Select a power drive
4. Determine a motion profile
5. Select a controller
6. Select a software platform
Things to Think About

- Control loop rates, motion participation in other processes
- Motion synchronization with I/O
- Flexibility to add/change an axis, control mode, etc...
- Communication ability to other parts of the system
- Code integration with other parts of the system
Summary

- Understand the following:
  - Motion Controller Components and Responsibilities
  - Motion Control Types and Common Architectures
  - Motion Control Programming and SW Approaches
  - Integrating Motion into a Larger Application
Nate Holmes
Group Manager, Vision and Motion R&D

National Instruments Corporation
11500 N Mopac Expwy
Austin, TX 78759

nate.holmes@ni.com