How to Choose a Servo Controller for Your Application

In the realm of automation there are many items that need to be considered during the design phase to get the project together and headed in the correct direction. There are mechanical criteria that can at times dominate the design approach, as the forces and motion profiles are at the forefront of what drives the outcome of automation. In conjunction with the mechanical criteria, the motion control electronics need to be seriously considered at the start of the design process. When controls and mechanics are well-integrated, the result is a robust machine that performs well at a reasonable cost.

To proceed in choosing the control approach that will best suit your project, first let's look at the basic component options available. There are a vast array of control products, and they can typically be broken down into groups defined by how much development effort is required to produce a fully functioning control system. The groupings are: design based on an FPGA using an ASIC motion control chip, board level controllers and box level controllers.

As you can imagine, starting with a chip-level design requires the greatest amount of development effort, as the overall circuit design needs to be created, parts sourced, PCB designed and manufactured, firmware and interface code written and proofed, and overall product testing completed. While starting with a blank FPGA will require huge amounts of programming and troubleshooting time, using an ASIC motion control chip designed precisely for control of a motion system can greatly reduce a lot of risk and development time, with very little difference in per-unit cost.

Board and box level controllers significantly reduce the development time, as the basic structure of the control system has been developed and proven, but the downside for these controllers is that the per-unit cost is higher and the performance and features are harder to change or customize. In making a decision on which course to pursue, one's own capability needs to be assessed, as does the quantity of control systems that will need to be manufactured. (For small quantities it tends to make more sense to start at the higher-level components, as the development costs are low enough to justify the higher component cost.)

This whitepaper will cover what factors need to be taken into consideration when selecting a servo motion controller, as well as common motion profiles and other feature considerations that may be helpful depending on your application.

Define the # of axes:

The first step is to determine how many motors will be in the system and how they need to be controlled. Each motor that is going to be electronically controlled will be defined as an axis. If three motors are used to move three linear slides that are all orthogonal, this would define a Cartesian space and is a common approach for 3-axis systems such as milling machines. When a fourth motor is added to the Cartesian space, typically rotational, this would create a 4-axis machining center. Once the number

of motors has been determined, the next step will be to decide whether these motors need to be coordinated in their movement.

Define the motion set:

Now that the number of motors required and how they interact is defined, the next step is to determine the type of movement required. There are two types of coordination between motors: linear interpolation and circular interpolation. Linear interpolation is when two or more motors are moved in conjunction in a linear fashion. For two linear motors, the movement would describe a line between the two motors. Circular interpolation moves the motors in a non-linear fashion, so two motors utilizing circular interpolation would create a circular motion between the motors. The basic shapes that can be created with linear, circular and linear/circular interpolation movements are shown in Figure 1.

Linear-independent axis

If the motors controlled move independently of each other, any combination of motor controllers could be used. For example, if there are four motors in a machine, four 1-axis controllers could be used, as long as there is a mechanism or communication between controllers to ensure the motors perform their tasks at the appropriate times. Depending on the architecture of the machine, it might make sense to have all of the single-axis controls integrated into an assembly, or to use a multi-axis controller in order to keep the wire mess down.

Linear-coordinated axes

If a machine needs to coordinate two or more motors, the controller is going to need to have an interpolation function. With interpolation, the relative movement of each motor is referenced to the other motors in the interpolation group. It is easy to visualize coordinated linear movement when dealing with two or three orthogonal axes. The classic Cartesian coordinate system of X, Y, and Z are useful to describe this. If the X and Y axes represent linear motor 1 and 2, they form a plane. Coordinated movement between those two motors can be described by a line with an angle θ between either the X or Y axis. When this system is expanded to include the Z axis, the movement would be described by a line with the angle θ and an angle ϕ that describes the elevation of the line relative to the X-Y plane.

This is only a visual representation of the linear interpolation between axes and isn't the only way linear interpolation can be used; any type of multi-motor movement that has a linear relationship between the motors would qualify. For example, if motor 1 needs to move 2.5 times as far as motor 2 in the same period of time, there exists a linear relationship between them. You *could* just use independent

controllers to do this and program them for this feature; for a static design that might work, but in practice errors will creep in that wreak mayhem on operations. When the motors are linked via coordinated motion, the movement is relative. You will be able to make adjustments, and if something causes one motor to slow down, the controller can compensate to keep the motion smooth. To have truly coordinated movement, the motors or the movement induced would require feedback to the controller so movement aberrations can be detected.

Linear interpolation is generally simpler to program, as the controller does the calculations. By specifying the end target position for all involved motors, the controller will take that information and will calculate and adjust the output pulses as the movement is executed, so each axis reaches its position at the same time. Nippon Pulse's PMX-4EX box controllers are an example of this, where a 4-axis interpolated move is commanded by: X[target]Y[target]Z[target]U[target]. If one of the axes isn't given a value, the corresponding motor will sit idle.

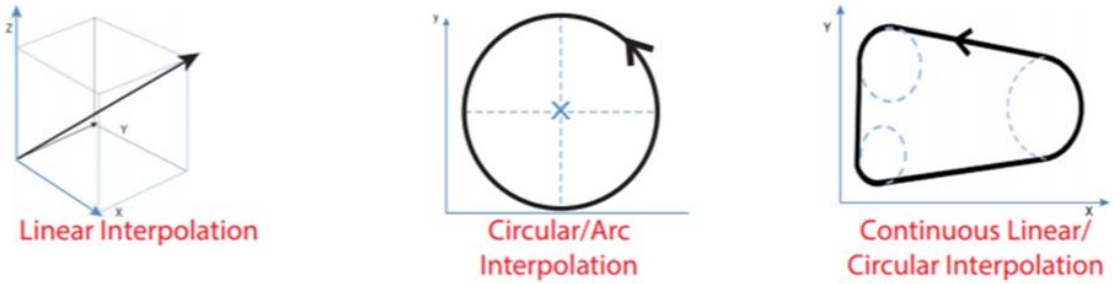

Figure 1. Linear, circular, and linear/circular interpolation movements

Circular-coordinated axes

A circle is mathematically described by the equation $R^2=X^2+Y^2$. There is nothing in that equation that lends itself to a linear thought process, so if we want to have a non-linear type movement relationship between motors, we need to talk about circular interpolation. The ability to move two motors with a circular movement relationship allows for the generation of contours, or to have a motor speed up and slow down relative to other motors in the circular interpolation group. This can be visualized by looking at the rate of change in Y and X values at the top of a circle ($\pi/2$ radians). The value of Y will change very little in comparison to the value of X as it passes the top of the circle so the Y speed is low and the X speed is high. At the π radians position on the circle, the situation is reversed where the Y speed is high and the X speed is low. A graphic of this is shown in Figure 2.

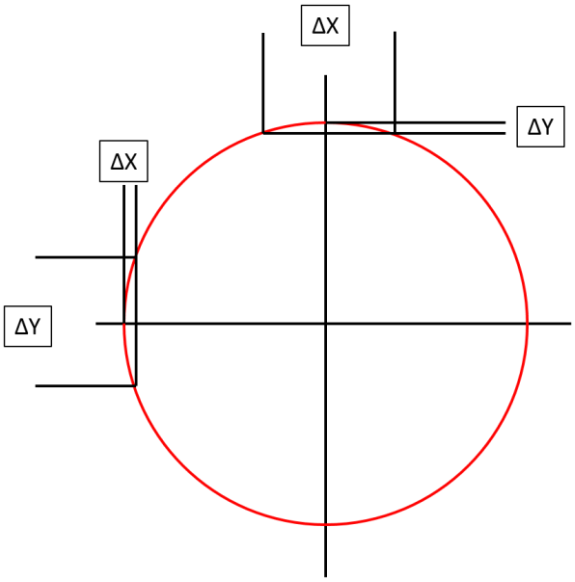

Figure 2. Non-linear changes in X and Y values

Linear & circular-coordinated axes

Motion in only a linear, linear-coordinated or circular-coordinated fashion limits the shapes that are possible. In the case of 3-axis machining you could make pointy objects like a pyramid (linear) or a circle. It gets more interesting when you combine the two interpolation types for linear/circular interpolation. With this type of motion capability contours are possible, which opens up the ability to produce almost any three-dimensional shape or to do highly intricate and/or complicated engravings and etchings.

Other considerations:

Depending on how and what you need to control, the needed level of sophistication of the controller will change. Many controllers have levels of sophistication that include linear interpolation only across the axes and linear/circular interpolation models. In addition to the ability to have coordinated motion across multiple motors, there are other considerations in choosing a controller. Consider the output control capability, how the I/O is configured, and what the programming/control set looks like. These can range from no flexibility and little capability to a lot of flexibility and capability. At the low end of the spectrum, the cost will be low (as is the development/learning curve), while on the other end you'll pay for the enhanced features and it'll take time to learn how best to utilize it.

Output control:

What type of output control is necessary? Are you going to control stepper motors with no feedback systems or do you need a position- or velocity-controlled motion? How fast do you need to be able to move and at what resolution? The answers to these questions will determine if you need encoder feedback to be able to control for position or velocity. The speed and resolution will determine the output pulse frequency capability of the controller and the encoder input frequency capability required to meet the movement. Typically, the output pulse is equivalent to the movement of an encoder pulse. The PMX line of box controllers at Nippon Pulse have a 6 MHz output pulse frequency and a 5 MHz encoder input frequency, which is more than enough for any stepper application and most servo systems.

I/O configuration:

A fully operational system usually needs to have inputs other than an encoder and outputs other than pulse and direction to a motor driver. The additional inputs help to give the controller other safety-related information as to where the motion is in space through the use of end limit switches, home switches, and other trigger switches. When these are triggered, the controller can use that information to define a coordinate system. This in turn allows for a well-defined motion profile. It also can ensure that certain motions or actions are completed before follow-up actions are executed.

The additional outputs can be used to send trigger signals or indications of actions completed. Typically, the outputs are digital in nature with on/off states. These can be used to send signals that are used to synchronize motion or actions between multiple controllers. As an example, the PMX line of controllers at Nippon Pulse typically have a set of safety inputs (plus/minus end limits, home and enable) along with 8 general purpose digital inputs and 8 general purpose digital outputs. In addition, some controllers have analog inputs for enabling joystick control.

Programming/control set:

How you communicate with the controller and what you can tell it to do is just as important as the inherent capabilities of the system's hardware design. A useful controller will have the ability to receive commands from an external user-written program and an onboard program/interface, which will allow for testing functions and can run on its own if necessary. Controlling through an external program will allow for incorporation of external data into the motion control process. The external program can process and analyze the data and use that as input to determine the next set of motions needed. If that level of sophistication isn't needed, then an onboard control with programming capability should be accessible to alleviate the need to develop a user interface.

Choosing a controller:

With these factors and requirements in mind, you should have an easier time determining your controller needs, especially in regards to whether your application would best benefit from the use of controller chips (with all the benefit of design-from-scratch programming capabilities) or a more turnkey all-in-one solution with fewer opportunities for customization, such as a box controller.

If you require further assistance in selecting the best controller for your application, contact Nippon Pulse's applications engineers, who would be happy to walk through your application's requirements and make suggestions. For more information about Nippon Pulse's chip, board and box-level controller options, visit our website at nipponpulse.com.