

RESOLVER-BASED, CLOSED-LOOP POSITION AND VELOCITY CONTROL FOR THE LANSCE-R WIRE SCANNER*

J. Sedillo, LANL, Los Alamos, NM 87545, U.S.A.

Abstract

This study evaluates a technique for the closed-loop position and velocity control of a wire scanner actuator. The focus of this technique is to drive a stepper motor-driven actuator through a 1-mm move using a combination of velocity feedback control and position feedback control. More specifically, the velocity feedback control will be utilized to provide a smooth motion as the controller drives the actuator through a pre-planned motion profile. Once the controller has positioned the actuator within a certain distance of the target position, the controller will transition to position-based feedback control, bringing the actuator to its target position and completing the move. Position and velocity data is presented detailing how the actuator performed relative to its commanded movement. Finally, the layout of, and algorithms employed by the wire scanner control system are presented.

INTRODUCTION

A goal of the LANSCE-R wire scanner design is that it possess the ability to utilize closed-loop feedback for the positioning of the wire scanner actuator. The focus of this study is to describe a controller that utilizes a velocity controller to provide smooth acceleration, cruise, and deceleration motions until finally handing-over control to the position controller to bring the actuator position to within some distance of the desired position.

For this evaluation, a LEDA wire scanner was used. This wire scanner is driven by a Parker OS22B stepper motor coupled to a Moog resolver. A National Instruments compact RIO 9074 was used as the controller. As is typical in general for closed-loop control systems; in order to position the actuator, the controller

reads the sensor data of the resolver, compares the resolver data with a commanded set point, and then acts on the drive mechanism (i.e. motor) to drive the actuator to the commanded set point. The drive pulse rate was limited to any frequency within the range of 15.2 Hz to 2.5 kHz.

POSITION AND VELOCITY CONTROLLER

Position Controller

The position controller (algorithm shown in figure 1) reads the actuator position data from the RDK 9314 (resolver-to-digital converter) module and subtracts that position from the commanded position resulting in a position error $e(t)$. This error is then fed into a proportional, integral, and derivative controller. The proportional controller simply multiplies the position error by a constant known as the proportional control constant K_p (unitless). The proportional control constant has the effect of driving the actuator at a higher rate of speed if the position error is large, and a lower rate of speed if the position error is small. The derivative portion of the controller subtracts the present value of the position by the previous position value. This result is then multiplied by the derivative control constant K_d (μ seconds).

Generally, the derivative controller is beneficial for command tracking since it detects deviations in the rate of change between the commanded and measured positions (a.k.a. position error). The larger the deviation in the rate of change of the error, the more severely it acts to correct the error. Finally, an integral controller simply sums the position error over time.

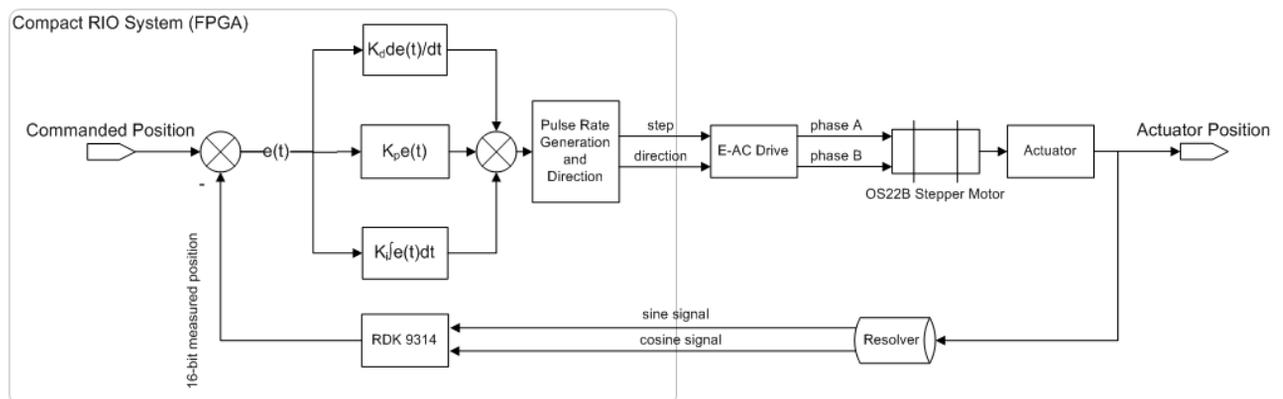


Figure 1: Position Controller Algorithm.

*Work supported by U.S. Department of Energy

As with proportional and derivative control, the effect of the integral error is reduced or magnified by the integral constant K_i ($\mu\text{seconds}^{-1}$). Integral control has the benefit of driving the system to zero steady-state error.

The results of the proportional (P), integral (I), and derivative (D) controllers are then summed to form the control output. In the case of the system implemented here, the results of the PID controller act upon the step-rate of the stepper motor. Positive errors drive the stepper motor counter-clockwise (actuator forward) while negative errors drive the stepper motor clockwise (actuator backward). The larger the error, the faster the motor turns. The summed result of each controller (P, I, and D) is input to the pulse rate generator.

In this case, the summed result passes through the following relations:

$$PRF_{\text{stepper}} = K_{\text{clock}} \times \alpha \tag{1}$$

$$Direction = \begin{cases} \text{forward} & a \geq 0 \\ \text{backward} & a < 0 \end{cases} \tag{2}$$

where PRF_{stepper} is the pulse rate applied to the stepper driver, K_{clock} is the clock rate of the FPGA, and α is the resolver angular measurement of position in radians. The summed resolver angle error value may vary anywhere from 380×10^{-9} to 62.5×10^{-6} radians; corresponding to a controller-induced pulse rate from 15.2 Hz to 2.5 kHz. The error's effect on step rate keeps the system operation equivalent to servo motor systems since the step-rate is linearly related to the angular velocity of the stepper motor and is a parameter which can be changed rapidly.

Velocity Controller

The velocity controller (algorithm depicted in figure 2) behaves in much the same way as the position controller. However, in this case, the velocity controller acts upon the velocity error; not the position error. In this case, proportional-only control is utilized since the velocity data attained from the stepper motor motion has a high degree of high-frequency noise.

In order to provide smooth motion, the velocity controller must first have a filtered velocity signal obtained from the resolver. This filtering is necessary because the step-like nature of the stepper motor introduces high frequency noise into the position measurement. The effect of this noise is exacerbated once its derivative is taken to compute the velocity. This noise is reduced through the use of an 8-point running average filter. The second method utilized to smooth the motion is the use of a pulse-code modulation technique to modulate the stepper pulse frequency. This technique causes positive error to increase the pulse rate and negative error to decrease the pulse rate. It prevents the motor from being driven in reverse in order to slow it down.

The overall operation of the position and velocity controller is summarized in the flow diagram of Figure 3. It summarizes how the two controllers (velocity and position) are coordinated in order to accomplish a move. In this scheme, the velocity controller acts first, following a motion profile until the actuator's position resides within a specified distance of the target position. Once within this target position, the position controller takes over and positions the actuator to within a smaller distance of the target position. Finally, when the actuator has been adequately positioned, both controllers quit operating on the actuator in order to prevent dithering and lock the actuator position in place.

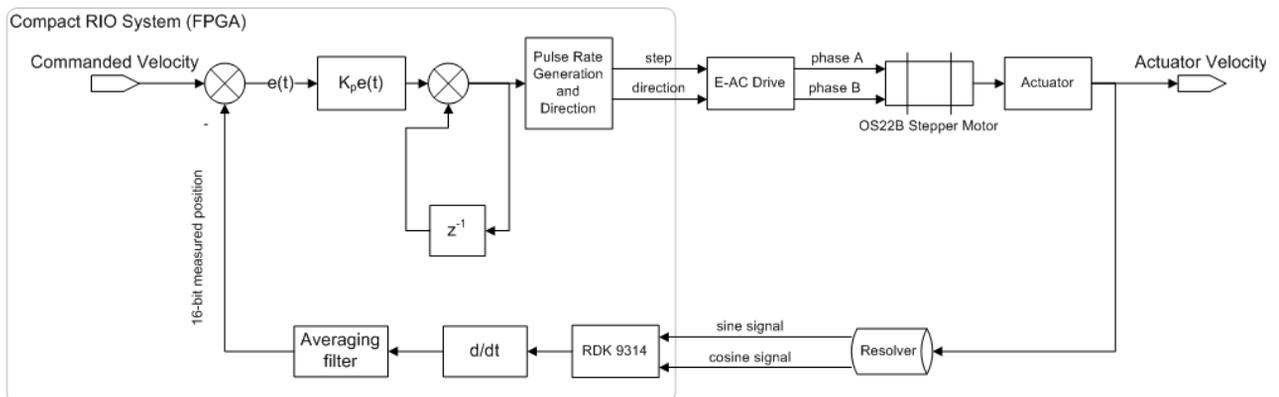


Figure 2: Velocity Controller Algorithm

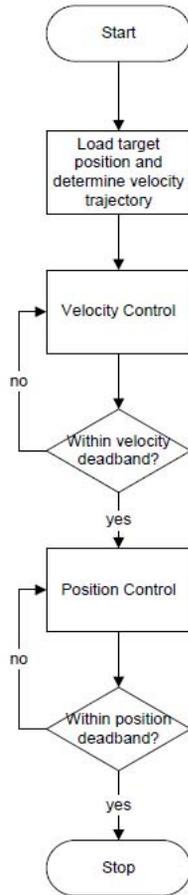


Figure 3: Velocity and Position Controller Switching.

POSITION AND VELOCITY CONTROLLER PERFORMANCE

The following plots detail the motion of this control system for 1-, 3-, and 5-mm moves. The resolver resolution was set for 12 bits with a 16-bit readback resulting in a position resolution of 1.9 micrometers.



Figure 4: Plot of position of 1-mm move.

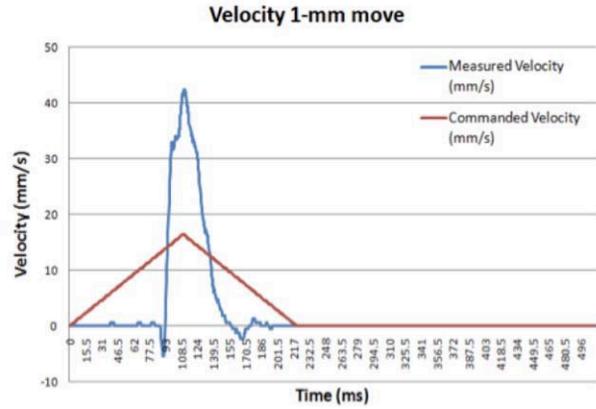


Figure 5: Plot of velocity of 1mm move.



Figure 6: Plot of position of 3mm move.

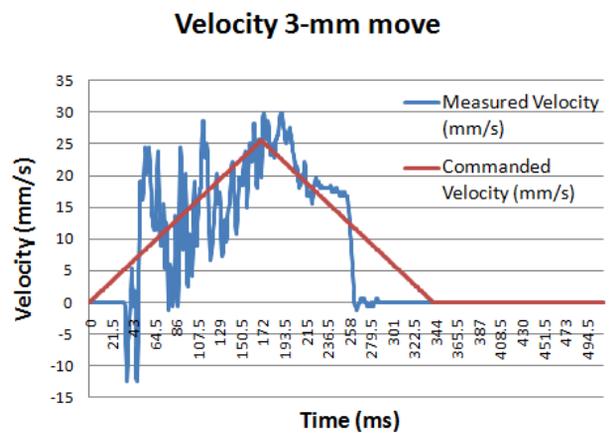


Figure 7: Plot of velocity of 3mm move.



Figure 8: Plot of position of 5mm move.

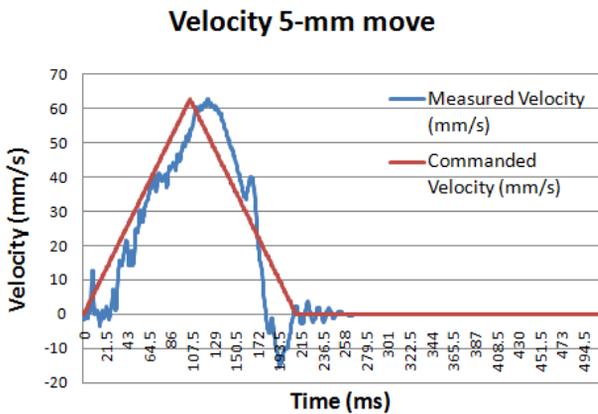


Figure 9: Plot of velocity of 5mm move.

The data suggests that the combined velocity and position controller is quite effective for larger moves, while performance is decreased for smaller moves. This may be due to a few factors. First of all, movement resolution (i.e. motor steps per move) is reduced for smaller moves, thus the ability to make these moves “smooth” is reduced. This is evidenced by the granularity of figure 4. Secondly, smaller moves either require higher proportional gain, or increased velocity setpoint values to alleviate the initial lag in response indicated by figures 5 and 7. The 5-mm move of figure 8 shows the smoothest motion in position.

SUMMARY

All in all, it is evident that a reasonable degree of position smoothness can be attained by the use of this velocity control method generally in moves that are larger than a millimeter.

Improvements in performance may be attained with decreased step size, improved PID tuning, and improved low-pass sensor noise filtering. However, the tradeoff associated with decreased step size will be a decrease in applied motor torque whereas the tradeoff associated with lower-bandwidth sensor noise filtering will be decreased controller bandwidth. Future studies of brushless and dc servo motor systems may improve the tracking capabilities of wire scanner actuator systems. Furthermore, others [1] have shown that full-state estimation and PID control techniques of stepper-based systems are possible through the use of a linearization controller and a stepper motor driver capable to driving the individual stepper phase windings in an analog fashion.

REFERENCES

[1] R. Rekowski, “Implementation of an Exact Linearization Controller for a Permanent Magnet Stepping Motor Using Only Optical Encoder Feedback.” Master’s thesis, Univ. Pittsburg, 1991.