Design and Implementation of a Motion Control Program to Assess the Consistency of the Flying Wire‟s Feedthrough and Coupling

Kayla Malone
School of Engineering & Technology
Alabama Agricultural and Mechanical University
Normal, AL 35762

Supervisors: James A. Galloway and James Zagel
Accelerator Division
Instrumentation Department
Fermi National Accelerator Laboratory
Batavia, IL 60510

Table of Contents

Abstract

The reliability of the hi-flex bellow coupling and the magnetic feedthrough utilized in the Flying Wire systems are two prime components that require research to increase dependability. The current design of the Flying Wire system has an issue with the coupling and the feedthrough being overwhelmed from prolonged use. In order to improve the dependability of these two components, a motion control system was designed to test these specific components. Using a test setup, a motion control program was written to analyze the consistency of the coupling and feedthrough. The motion control program was created by programming the Elmo Solo Whistle digital servo drive, which stores and executes the designed program. The system had to be tuned so that the Elmo Solo Whistle digital servo drive can provide the most suitable parameters while also reducing error. Once the setup has been tuned, the system operated continually for about 1.6 days to gather data about the coupling and feedthrough. This paper reviews the hardware and software format, proficiencies, and the results from the test system.

1. Introduction

Fermilab‟s particle accelerator, the Tevatron, is the United States‟ highest energy particle accelerator. The Tevatron, a circular proton-antiproton collider known as a synchrotron, can accelerate protons and antiprotons in opposite directions to a maximum energy of 1 TeV. The proton beam and antiproton beam trek the four mile Tevatron close to the speed of light. The Flying Wire system is utilized to detect and measure the size of these beams. The Flying Wire system determines the profile of the beam by passing a moving wire through the beam. The wire is actually a carbon fiber filament, which has a diameter of 7 microns, drawn taut between the prongs of a fork. From a relative perspective, the carbon filament is finer than strand of human hair. The size of the beam is around 5 mm, which makes the wire much smaller than the beam. This wire flies at a speed of 6.91 m/s. Figure 1 shows a representation of the flying wire system.

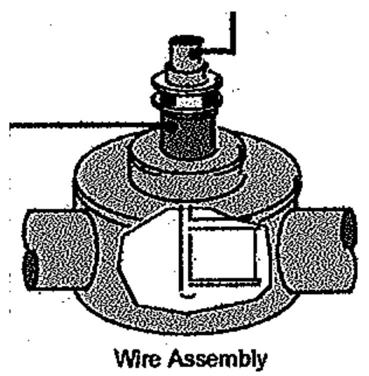

Figure 1 Flying Wire test system

In the accelerators, the position of the wire is determined using a resolver. When the flying wire intercepts the beam, secondary particles from the beam particles scatter. These secondary particles that have scattered have an intensity proportional to the number of beam particles present at the position of the wire (Blokland). The secondary particles are detected with an adjacent scintillator paddle that will produce light. A photomultiplier tube transforms the light into an electrical signal, which is charted to provide a profile of the beam.

The Flying Wire has to complete a path that totals 540 degrees. During the fly, the wire accelerates fast in order to pass through the beam twice, at a constant speed, and then it decelerates. Once the Flying Wire rotates 540 degrees, the next fly will rotate 540 degrees in the opposite direction which places the Flying Wire back at the starting position. Figure 2 shows a diagram of the fly path. In Figure 2, Park 1 represents the starting position and Park 2 represents the ending position for the complete fly. The same speed, acceleration, and deceleration parameters are employed for both flying directions. The Flying Wire system for this project will only fly clockwise and counter-clockwise on a horizontal plane.

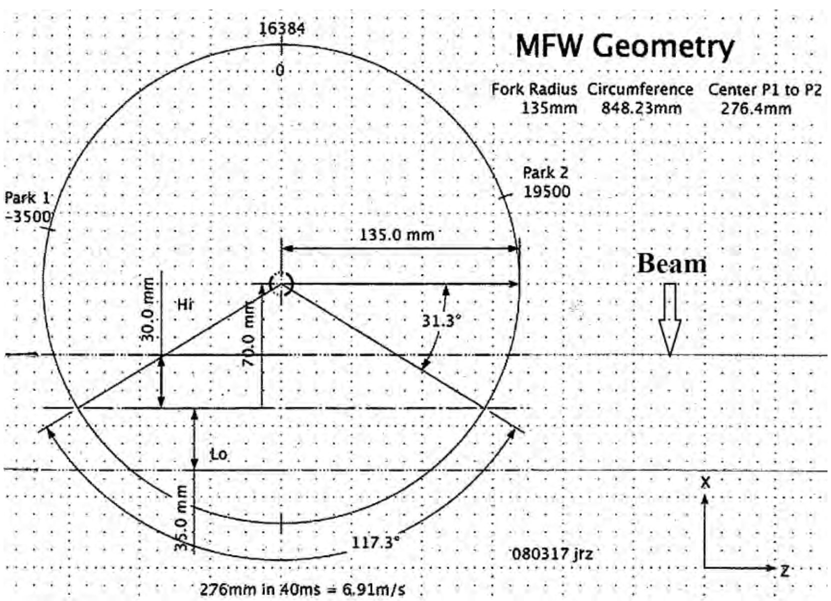

Figure 2 Fly path of wire

1.1 The Current Flying Wire System

The current system uses a resolver which is immune to radiation damage. The current system works, but it has experienced premature bearing failure in the feedthrough. A new feedthrough was chosen to test another vendor‟s product under some conditions to see if it would last longer and if it was easier to maintain than the current design. The Flying Wire system used in this project uses an optical incremental encoder because that is the type of drive that was available in time to test. The Elmo drive was chosen to make the development of the test stand easier for use and as an experimental platform for future tests. The purpose of this project is to see if the Flying Wire feedthrough setup and the bellow coupling can withstand a year‟s equivalent of rotations, which is 36,500. In order to perform this, control loops for the motor controller must be designed.

1.2 Elmo Solo Whistle Digital Servo Drive

The Solo Whistle is a lightweight and highly compact solution which can be used whenever reduced size and weight are essential to the application (Elmo manual). The *Elmo* Solo Whistle digital servo drive is a motor control drive that contains a high level programming environment and supports a variety of feedback, like an incremental encoder, resolver, or potentiometer just to name a few. The test system in this project will operate in conjunction with an optical incremental encoder and a DC brushed motor. A DC power source, 12 ~ 95 VDC, is used to operate the servo driver in current, velocity, and advanced position modes. The Solo Whistle can be used by itself or with other drives. Elmo‟s *Composer* software, which is Windows-based, is exercised to tune the drive. *Composer* can also be used to quickly alter any parameter of the digital servo drive to analyze or enhance the use of the drive with the motor. The Solo Whistle can be fully programmed using Elmo‟s *Metronome* motion control language, which is very similar to the programming language *C/C++*. This drive can store a motion control parameter set from a designed program that has been uploaded to the drive.

1.3 Incremental Optical Encoder

The optical incremental encoder being utilized in the project contains ABEC class 7 bearings and a chrome-on-glass disc. It supplies information on where the wire is positioned. On the glass disc, there are three special paths, which are A, B, and Z. The paths of A and B have 4,096 markings on them, while the Z path contains the index. The index, which is a single marking, indicates a known reference position of the wire. An LED illuminates through the index, A and B tracks, and is recognized through optical sensors.

2. Methods

2.1 Wiring the Motor Control/Power Supply

When beginning the project, the motor controller/power supply chassis consisted of an unwired start and stop button, the Solo Whistle digital servo drive, one power supply, and an unwired LCD counter. The Solo Whistle digital servo driver was already installed in the motor controller chassis however, the components of the motion controller were not wired in a circuit. An electrical schematic had to be designed. National Instruments Multisim 11.0.2 was used to design the circuits. The system requirements consisted of the start/stop buttons working correctly, the counter increasing with each fly, and a working LED program status indicator. The first circuit schematic consisted of only the start and stop buttons which are momentary switches connected to the Solo Whistle servo drive.

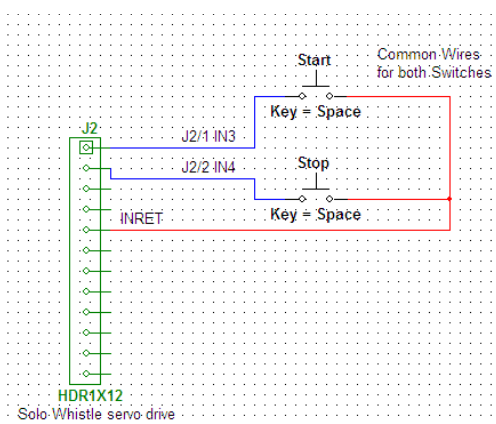

Figure 3 electrical schematic of the first iteration of the Start/Stop circuit

In Figure 3, the HDR1X12 connector represents the terminal connections that are attached to the Solo Whistle servo drive. The start button is wired to digital Input 3 (IN3) and the stop button is wired to digital Input 4 (IN4). The common wires of both switches were connected to the Input Return (INRET). With this circuit design, the buttons would not work. When the start/stop button is pressed, the circuit would close and be completed, as shown in Figure 4. If either of the buttons were pressed, digital Input 3 or 4 would turn green, indicating if the start/stop button was working correctly. Since the start/stop button is a momentary switch, digital Input 3 or Input 4 would turn green when pressed and then turn back to gray when the button has been released. The momentary switches were wired to the normally open position.

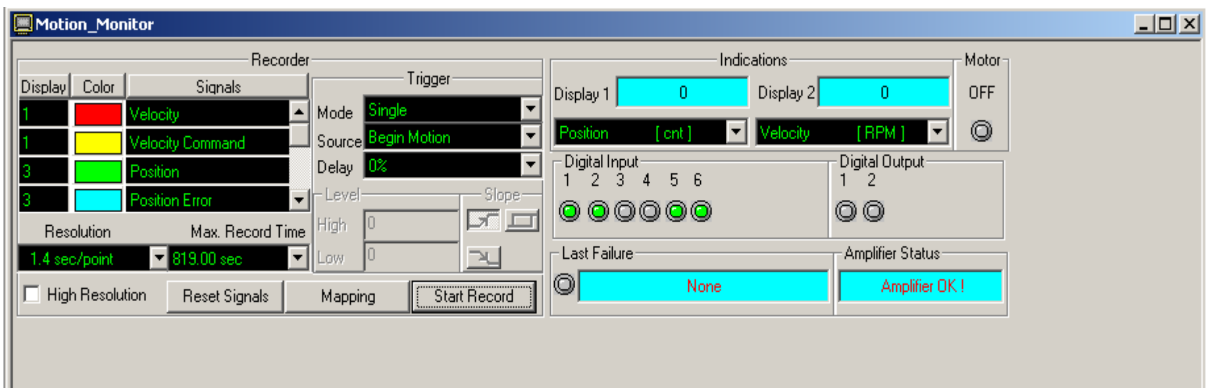

Figure 4 Motion Monitor from the Elmo Composer Software.

Figure 4 shows the Motion Monitor from the Elmo *Composer* application. Inputs 3 (Red Arrow) and 4 (Yellow Arrow) are off due to a problem with the circuit design. It was discovered that the switches were not receiving enough voltage so that the switches could work. The initial assumption was that the Solo Whistle servo drive would produce the voltage so that it would recognize that the buttons have been pressed, however, that was not so. The inputs required their own power supply, which was 5 V due to the model of the Solo Whistle. For the digital inputs, the internal resistance, R_{in}, is 1.43 K and the digital inputs had an internal current, I_{in}, of 2.8 mA at 5 V, which is depicted in Figure 5.

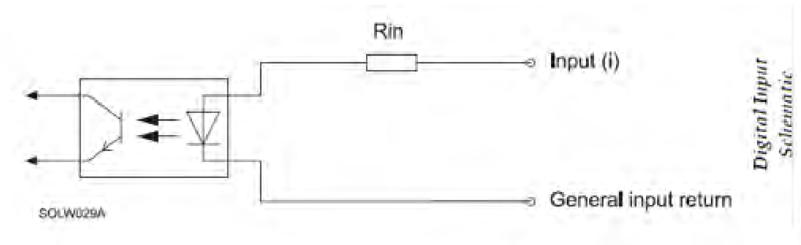

Figure 5 Schematic of Solo Whistle inputs

The highest-level input voltage of the Solo Whistle servo drive is 24 V; however, we utilized the ZWS10-5 single output power supply. Its voltage ranged from 4.5-5.5 V with a max current of 2 A. The 5 V power supply was then secured into the motor controller with the other devices and the start/stop wires were connected to it. The negative terminal of the 5 V power supply was utilized to ground all of the digital inputs through the Input Return (INRET) connection.

When the 5 V power supply was installed, the start/stop buttons were tested to observe if Solo Whistle digital servo drive would detect if the switches have been pressed. At the first attempt, the Solo Whistle only acknowledged the start button when it was depressed, not the stop button. A voltmeter was used to check if there was actually 5 V being applied to the two switches. There was only 4.8 V being directed to the switches instead of the desired 5 V. 5 V is needed in order for both switches to work. The potentiometer on the 5 V power supply was adjusted and 5 V finally flowed through the circuit. At the second attempt, the start/stop buttons worked perfectly and the Solo Whistle software recognized that the buttons were being pressed as the circuit closed.

Once the start/stop switches were wired correctly and working, wiring the LCD 8-digit counter to the Solo Whistle was next. The counter has its own internal battery, so there was not a need to connect it to a power supply. The value on the counter indicates the number of motion cycles completed and is an important value that we do not want to be lost. If this data was lost, then the test would have been a complete waste and it would have to be restarted. The counter is wired to one of the digital output terminal slots. Since the counter is wired in this fashion, it can be controlled through the Solo Whistle software. By turning Output 1 (OUT1) on and then off through the software, the numerical display on the counter will increment. Output 1 has to be turned on and off, not just on because the display on the counter will not increment. This can be verified through the Solo Whistle application, *Composer*. The command for output 1 is OB[1]. By setting OB[1] equal to 1, output 1 will be on. Vice versa, output 1 will be off by setting OB[1] equal to 0. Figure 6 shows a schematic of the counter‟s wiring.

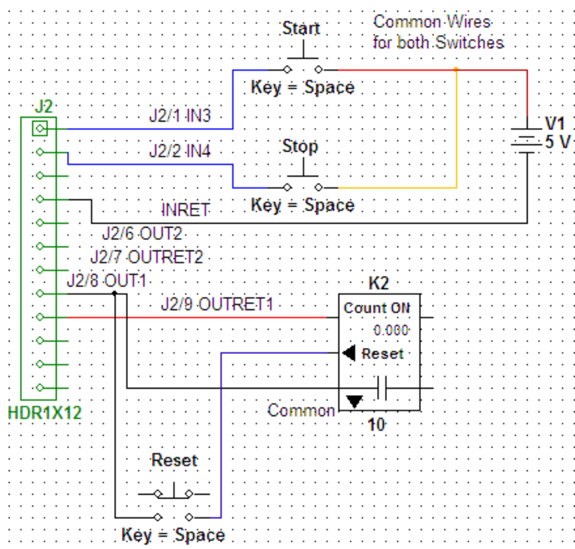

Figure 6 Counter added to the electrical schematic

The reset button on the counter was disabled by flipping the reset dip switch on the counter, because it can be pressed very easily. This helps to prevent someone from resetting the value on the counter accidently. The reset wire was connected to a momentary switch with a guard on it to remove the possibility of the counter being reset accidentally. When this switch is closed, the counter immediately resets and displays „0‟. The last hardware installation to be completed was the LED status indicator. The LED also required its own power supply in order to function. In the first attempt, the 1 kΩ was attached to the power supply. Then, a wire led from that resistor to the LED. The LED did not light up when it was tested. The wires attached to output 2 were switched to see if that would make a difference. However, that did not, so the 1 kΩ resistor was soldered to the LED. The first attempt did not work because the resistor"s leads did not stay secured in the power supply slot. The LED began to operate once it was tested with the second configuration. Figure 7 shows the complete circuit.

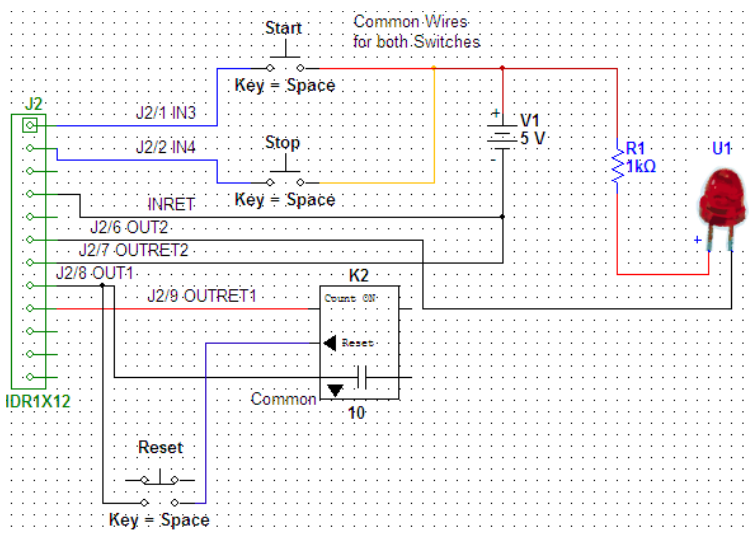

Figure 7 Complete electrical schematic

2.2 Tuning the First Flying Wire System

For the Flying Wire system, there is a desired velocity profile. Motor control systems basically have a velocity profile similar to the Flying Wire"s velocity profile. The motor has to accelerate to a set velocity, hold that velocity steady within a window (Θ_{SS}), and then decelerate in time. All of this motion has to occur within a window. Figure 8 shows the ideal velocity profile. In order to achieve the profile shown in Figure 5, the Flying Wire system has to be tuned.

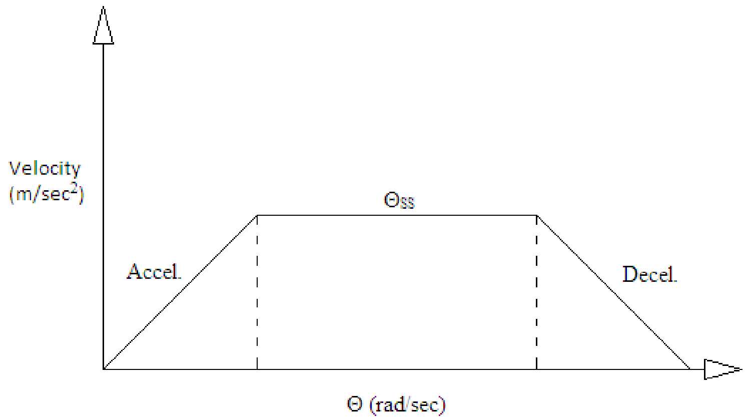

Figure 8 Ideal Velocity Profile

The current loop is the first process to tune because it is the most basic control loop. This step energizes the motor winding with a high-frequency current, in order to identify the dynamic response for resistance and inductance (Elmo manual). When this test is complete, an array of auto-tuned current controller factors is created. The results of the test are depicted in graphs that show the Reference, Response and Controller Out vectors. This test takes about 5 minutes to complete.

Tuning the velocity loop was the next process. This step adjusts the velocity loop and sets an optimal balance between control gains and precise motion on the one hand; and higher stress, measurement and quantization noise on the other (Elmo manual). The *Composer* wizard allows manual tuning, automatic tuning, and advanced manual tuning. Automatic tuning was chosen to tune the loops because the software determines the needed parameters automatically. Figure 9 shows a sample of the Velocity Tuning window in *Composer*.

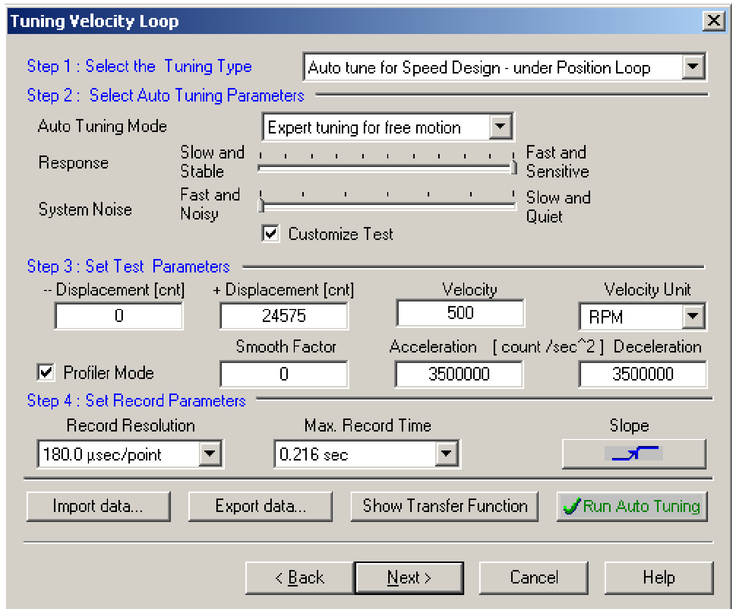

Figure 9 *Composer* window of the Velocity Loop Auto tuning.

In Figure 9, Auto tune for „Speed Design—under Position Loop" is chosen for the tuning type because this choice would provide more precise results when performing the actual tests. Under Step 2 in Figure 9, the auto tuning mode is chosen to be „Expert tuning for free motion" because the option is for the motion of the motor. The motor travels in a circular motion and is unbounded. The other option is „Expert tuning for bounded", which is necessary for linear motors or rotary systems with limits and angles. The response of the motor needs to be fast and sensitive because the Flying Wire has to accelerate to a certain velocity, hold that velocity steady, and then decelerate within a small window. If the speed is too slow, then the carbon fiber filament could burn up in the beam. The same goes to the system noise selection. The system can be noisy, but it needs to be as fast as possible.

Step 3 in Figure 9 is the custom parameters set for the test. Since the Flying Wire has to rotate 540°, the displacement of the encoder was set to 24,576 counts. There are 16,384 counts in one revolution, which has 360 degrees. The Flying Wire completes one revolution, which is 360 degrees, plus 180 degrees. Half of 16, 384 is equal to 8,192; therefore, when 16,384 and 8,192 are added, 24,576 counts is the result. The velocity was set to 500 RPM and the acceleration/deceleration was set between 3,500,000—4,000,000 counts/sec^2 because these parameters presented the best graphical results. In the graphs, a steady state after the acceleration is desired. With these considerations, the auto tuning returned its results in a graph, as shown in Figure 10.

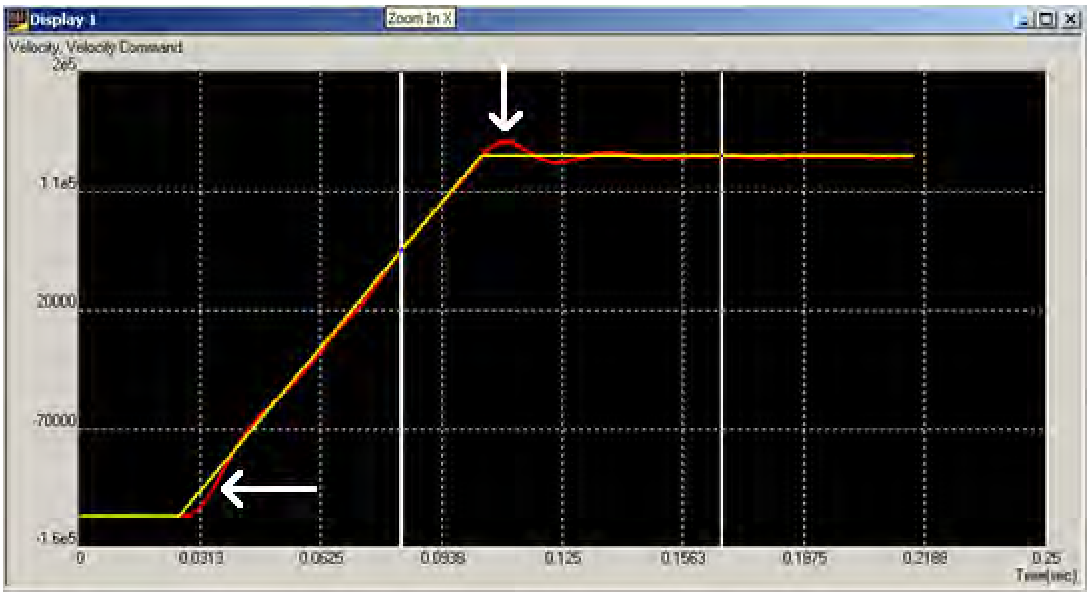

Figure 10 Auto tune result of the ideal velocity loop profile

The straight line in the graph represents the commanded velocity and the red line, where the white arrows are pointing, represents the velocity of the motor controller. Following the commanded velocity is desired; however the graph in Figure 10 doesn‟t show that. The motor controller over shoots the command velocity around 0.1 seconds, but then begins to stabilize around 0.14 seconds. This motor controller seems to always overshoot, even if it were manually tuned because the motor controller attempts to accelerate to the commanded velocity as quickly as possible. Since it is accelerating so quickly, the motor controller will continue until it notices that it has overshot the desired steady velocity. The current of the system will then drop just enough so that the motor can slow down to the steady velocity. Figure 11 shows the current graph in same scale as Figure 10. The larger the overshoot, the more the current will be decreased or reversed to arrive at the commanded velocity.

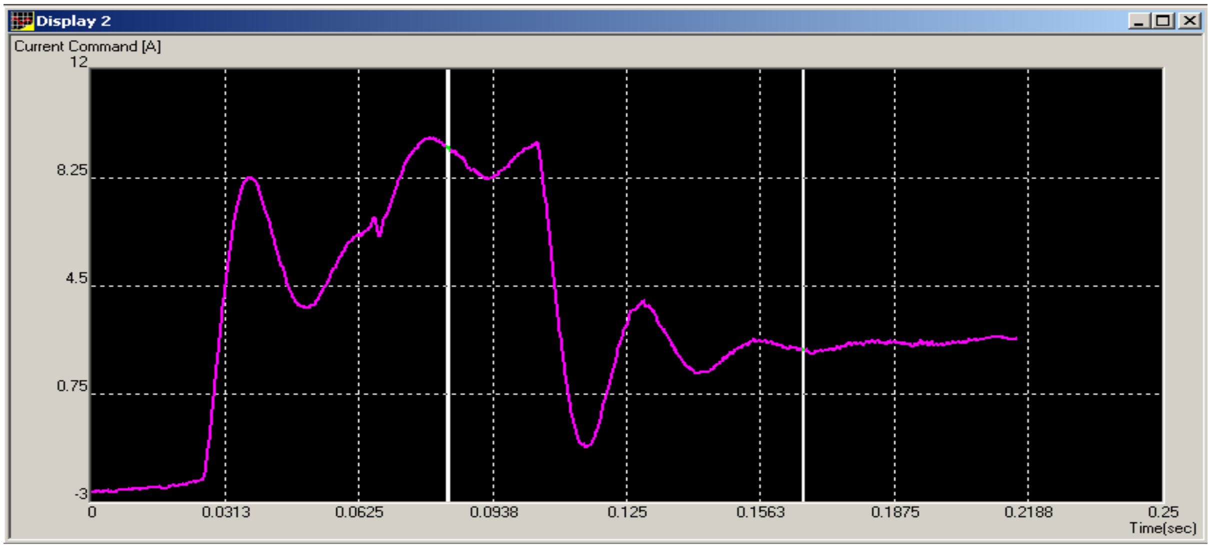

Figure 11 Auto tune result of current command for the velocity loop.

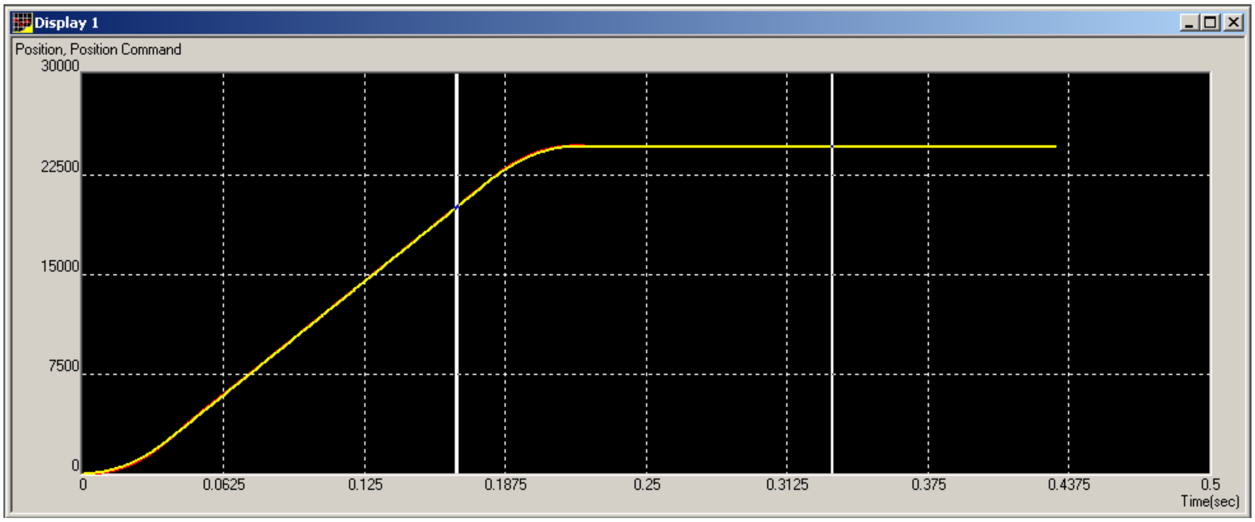

Figure 12 Auto tune result of position loop.

The next process was tuning the position loop. The steps required for this process is similar to tuning the velocity loop. This process tunes the motor to make sure it starts and stops in the correct position with minimal error. Figure 12 shows the results of the auto tuning of the position loop with the same parameters as the velocity loop. The graph shows that the motor performed very well. It means that the motor is doing what it was commanded to do, since the motor‟s position is in sync with the commanded position. The motor controller began in one position and swiftly progressed to the commanded position with minimum error. The results in Figure 12 are desired for the system.

2.3 Programming the Test System

Now that the hardware portion of the project was basically finished, programming the test system to operate ensued. The system needs to rotate 540° forwards and backwards, to stimulate normal Flying Wire usage. The motor controller can be tested through the Elmo *Composer* application; however, the program must be written in the Elmo *Studio* application. The program can be uploaded to the Solo Whistle digital servo drive from the Elmo Studio application. The Elmo Solo Whistle digital servo drive requires its own programming language, along with C as a foundation of program language. The only difference is that the Solo Whistle servo drive contains commands that only the Solo Whistle understands.

The maximum number of bytes the overall user program size is around 2,000 KB. This means that the program will not compile if the maximum bytes are exceeded. Every character counts towards the byte count; so, extra spaces, comments, and characters have to be limited. This proposed a predicament because the motor controller had to perform the required actions due to the programming of the Solo Whistle servo driver. It was assumed that the program would be close to 2,000 KB. Since comments for the program were needed for the next worker on the project, the program was edited through another word processing application.

In order for the test system to operate without the use of a computer, the uploaded program required an automatic subroutine. This command, "@#AUTOEXEC", is applied at the beginning of the program so that the program can be executed automatically upon power on. The acceleration, deceleration, and speed values of the motor controller system are located after the automatic execution subroutine. Global variables are also declared before the main function. The remainder of the program consisted of the main function and four sub functions, which are the start, stop, LED, and counter functions. Each of the sub functions are called within in the main function to do a specific job.

The motor of the test system is always off when the program begins, so a command must turn the motor on. When going through the first while loop of the main function, the motor variable, MO, must be on. The start function is continuously called until the start button is pressed. Once the start button is pressed, the start function will set MO = 1, which will turn the motor on. Once the motor is on, the program goes into the next while loop. Inside of this loop, the position relative, PR, is set to 24,575 counts which equal 540°. The begin command, BG, initiates the Flying Wire system to complete the 540° in the clockwise direction. The test system accelerates to the given velocity, holds that velocity steady for short window, and then decelerates to the provided relative position. After this, the program has the motor controller wait for 3 seconds before initiating the reverse motion.

As the test system stops at the destination, the value on the counter increments by one because the counter function is called before the wait command. Also, the LED on the front panel will blink on and off to let users recognize that the program is still progressing.Within the LED function, the LED can be turned on by setting OB[2] = 1 and it can be turned off by setting OB[2] = 0. It is similar to the counter function. After the 3 seconds, the motor rotates in the opposite direction with identical parameters. Again, the system waits 3 seconds and begins the cycle again until the maximum number of cycles has been reached or until the stop button has been pressed. Once the stop button has been pressed, MO is set to zero, which shuts off the motor and stops the program. While the program has stopped, the LED stays lit until the start button has been pressed again. If the program has completed the allotted number of cycles, the LED will flash faster than when the program was still going through the cycles. When that has happened, the LED will not stop flashing until someone comes to check the problem. The program can then be restarted from the Elmo Solo Whistle software.

2.4 Assembling the Setup/Second Test System

The test setup was constructed with the help of a mechanical engineer, technician, and a mechanical engineering co-op student. The co-op student had drawn the mechanical representation of the setup, shown in Figure 13, and then he came to assist in assembling the set up. While constructing the setup, nitrile gloves were worn to prevent fingerprints and dirt from getting on the inside of the feedthrough or on the inertial slug, since these parts will eventually be tested under ultra-high vacuum. Fingerprints and dirt will contaminate the vacuum. Ethyl alcohol was also used to clean the gloves in case they touched any other object.

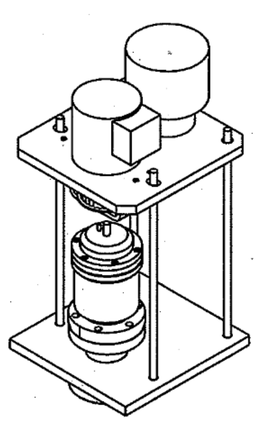

Figure 13 Mechanical drawing of Flying Wire setup

While finishing the construction, it was found out that the diameter of the inertial slug for the feedthrough was the exact diameter of the slot it was supposed to be in. This meant that the inertial slug would not fit into the provided slot. It was supposed to be smaller in diameter.

Another order was made for a new inertial slug, so the feedthrough was utilized without the inertial slug until the new one comes in. While waiting for the new inertial slug, the test setup was auto tuned to provide results with minimum error. There were two test setups used for the project. Figure 14 shows the first feedthrough that was used to set the auto tune parameters for the new feedthrough.

Figure 14 First Flying Wire setup

2.5 Tuning the Second Flying Wire System

The system had to be tuned to provide the best parameters for the test. The same steps from the first feedthrough were also performed on the second feedthrough. While the current loop was being completed, the system failed after testing the motor on various frequencies. It was thought that the reason for the failure was the size of the new feedthrough. The new feedthrough caused the system to have too much friction, which prevented the system from accelerating quickly. With this friction, the system required more torque and current to rotate the feedthrough faster. However, the motor could not provide the required torque because the load was greater than the load in the first setup. The motor would always max out the current, which reduced the acceleration and prevented the system from reaching the commanded velocity. Due to this friction, the system had to be manually tuned because the load was substantially larger. Figure 15 depicts a graphical result from the manual velocity loop test. The continuous straight line in the figure is the commanded velocity, while the erratic line is the result from the feedthrough. The system never reached the commanded velocity, so the setup was taken apart to see which components were the causes.

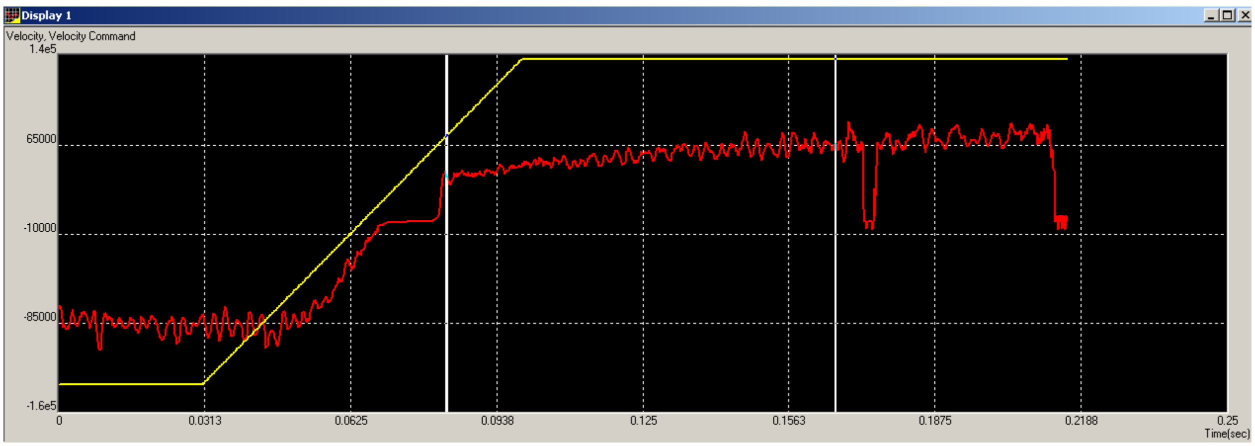

Figure 15 Velocity loop test for the second feedthrough.

The belt, which was drawn around the feedthrough and motor, was examined to see if it was being pulled too tight. The coupling was also checked and it was found to be slipping on the shaft due to improper fitting on the feedthrough. When these issues were corrected, the setup was assembled once again and the components were checked to see if they were tightly secured in place. The auto tuning tests were performed again. Although the results on the graph were better than the previous tests, the graphs still depicted some instability in the system. The system results were still erratic and the error was too large. When the position loops were performed, the system would overshoot the commanded position and then try to correct itself. The size of the load became a major problem to the system setup. Another problem of the system was the stability. When the setup was held down, the error seemed to be reduced on the graphs.

3. Results

Once the construction of the test system was complete, the destructive analysis began. The test system was started with the estimate to complete the total number of rotations in about 1.6 days. When beginning the trial, the test system was left overnight to operate so that the feedthrough and coupling could be analyzed the next day. On the next day, it was found that the test system had ceased its movement. The power switch was still on however, the test setup was not operating. The setup was examined to see why it had stopped operating. The power cord was checked to see if it were still in the electric outlet. It was thought that someone had turned off the system overnight. However, the coupling was found to be the problem. The coupling had broken below the bellows and it was no longer attached to the feedthrough. Figure 16 shows the separation.

Figure 16 The coupling has broken below the bellow because of misalignment.

The Flying Wire setup had a misalignment, which is the reason that the coupling had failed. The coupling was misaligned about 3 to 4 millimeters. The coupling was supposed to axially aligned between the motor and the feedthrough. Although it is impossible to line up multiple components exactly, the coupling provides the flexibility necessary. The coupling allows the system to have a small amount of variance in the alignment. Also, the coupling helps keep the system from being bonded. There can be a small offset in the coupling, but the analysis of the coupling proved that the offset was too great. In Figure 16, the misalignment can be seen clearly. Since only the top portion of the coupling was still attached to the motor, the bottom portion remained stationary. This presented grinding between the two parts of the coupling.

Through further analysis, the test setup shutdown after the separation occurred. The test system was tested through the *Composer* application to see when it stopped. The same program was executed; however, the application presented an over speed error after a few seconds of running the program. Since there was no load on the motor, the test system had over sped. There was no way of recognizing where the load was positioned because of the separation. After this over speed error, the test system powered off internally. The value on the counter at the power down was 7,706. The Flying Wire setup takes about 0.8 seconds to complete 540 degrees. A complete cycle is one rotation plus a wait of 3 seconds, which totals the cycle to be 3.8 seconds. The total time of the operation of the test system before shutdown was about 8.13 hours.

4. Conclusion

The feedthrough of the test setup endured throughout the analysis. The coupling did not survive to the end of the analysis. Due to misalignment of the test system, it was not able to complete the allotted number of cycles; therefore, the test system did not operate a year‟s equivalent. The test system operated for an equivalent of about 2.5 months. The components of the test system must be measured and setup correctly or a misalignment of the test system will happen once again. For future work, the components of the system will be analyzed to establish that they are correctly measured to the mechanical schematic to prevent misalignment. Also, obtaining the parts to the test system setup was an issue. Hopefully, the parts will be ordered on time so that the testing could be completed entirely. Also, more trials will have to completed to establish how long the coupling and feedthrough can last.

5. Acknowledgements

I would like to thank Fermilab and the SIST program for allowing me to participate in the amazing internship. Also, I would like to thank my supervisors, James A. Galloway VI, James Zagel, and Benjamin Vosmek, for all of their assistance and helpful knowledge on this project. Lastly, I would like to thank all the employees in the Instrumentation Department of the Accelerator Division that have aided me. Further thanks to all the people I have not mentioned and made this great experience possible. Thank you.

6. References

1. Blokland, W.; Vogel, G.; Dey, J.; , "A New Flying Wire System for the Tevatron," *Particle Accelerator Conference, 1997. Proceedings of the 1997* , vol.2, no., pp.2032-2034 vol.2, 12-16 May 1997

2. Elmo Motion Control. 2010. *SimpleIQ Software Manual.*

3. Elmo Motion Control. 2010. *Solo Whistle Digital Servo Drive Installation Guide.*

4. Fermilab. 2002, July 2. *Inquiring Minds*. Retrieved July 2011, from Fermilab: http://www.fnal.gov/pub/inquiring/physics/index.html