Elecanisms MiniProject 1 Report

Haptic Joystick

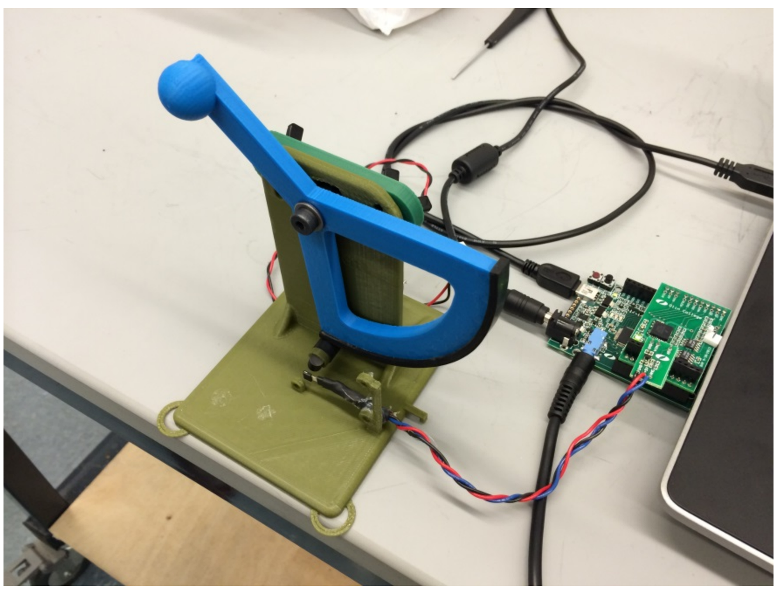
2/17/2015/I

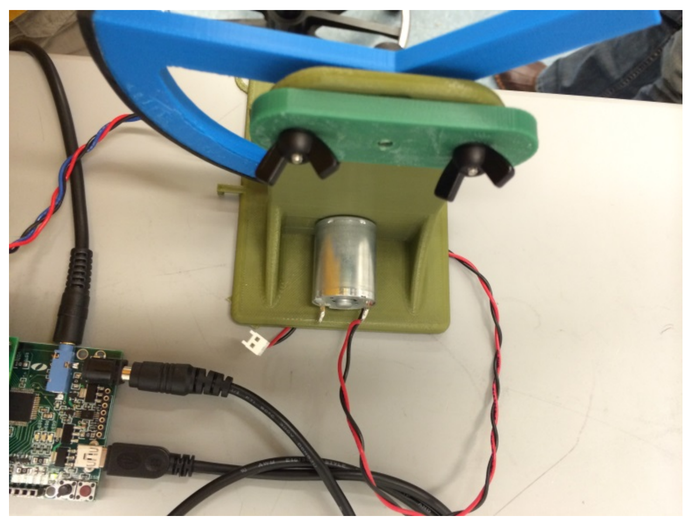
Kyle Mayer and Caitlin Riley

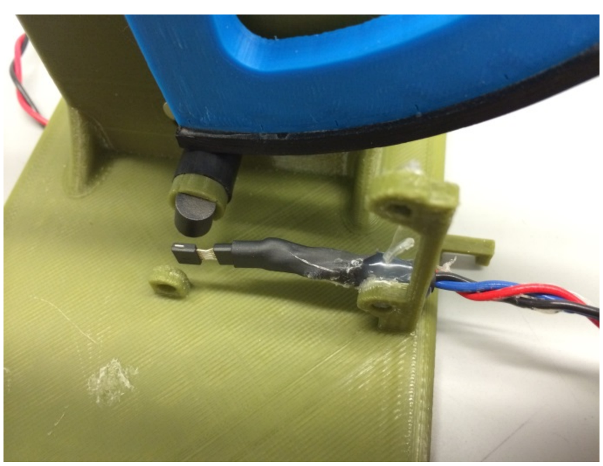
We built a haptic joystick that is capable of giving a user feedback by measuring the position of the joystick and manipulating the joystick's position via a friction drive.

The mechanical system is based off of the Stanford hapkit joystick. We considered modifying the kit, but decided not to because neither of us are mechanically interested or inclined, the current kit is known to work sufficiently, and we are not certain that the joystick will be used in our final project. We may consider modifying the mechanical system if it becomes a part of the final project. The main modification we considered was to turn the magnet shaft on a lathe to improve the roundness, and therefore the evenness of the friction drive as it rotates.

The main difficulties we ran into included too long of screws that stalled the motor (by locking up the armature), and 3d printed parts warping/ not being round. Below are photos of the completed joystick.



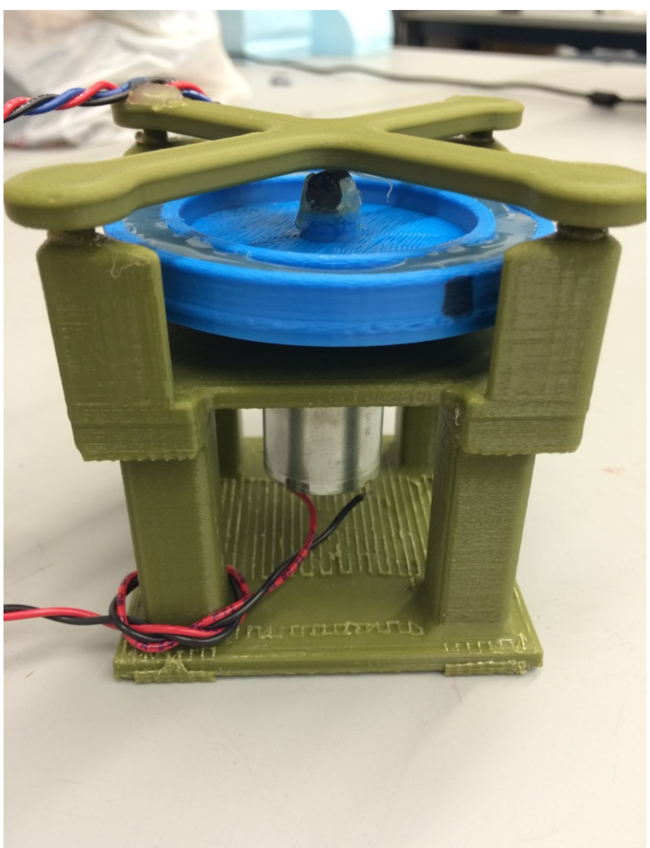


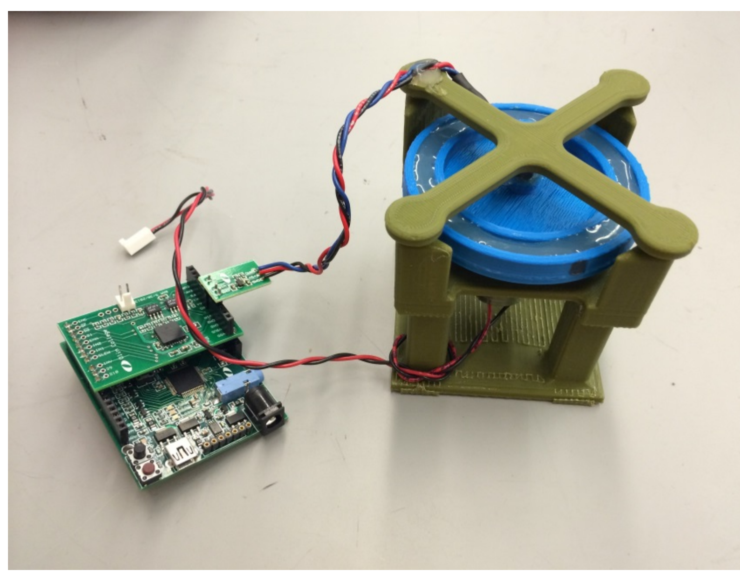


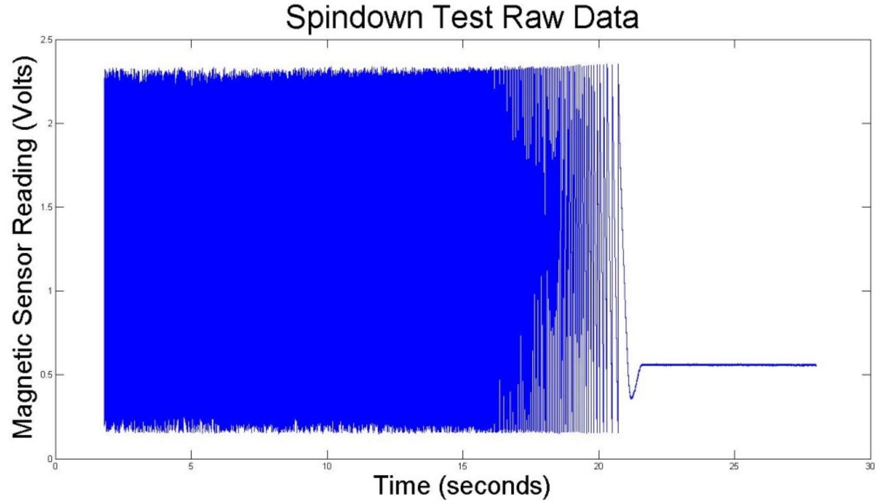
For the electrical system, we used a magnetoresistive sensor, a motor driver shield, and the PIC board designed for the class. The hardware worked sufficiently as designed, so no additional modifications were needed. We had difficulties getting the microcontroller to read the magnetoresistive sensor for a while due to the fact that we were plugged into the wrong analog pin.

For a spindown test, we quickly found out that our microcontroller couldn't keep up. A quick estimation says that a standard off the shelf DC brushed can motor probably doesn't exceed 20,000 RPM, or 333 revolutions per second. This is ~ 600 sensor flips per second (every 180 degrees). We said we probably want ~20 data points per sensor flip, so that the flip is sufficiently distinct from the datapoints. This means 12,000 analog reads per second. Although the microcontroller is capable of much faster than this, we were having timing issues when we tried to send this data back to the computer in real time. We tried using a buffer system, but found the microcontroller had nowhere near enough RAM. We considered using a different measuring system (such as a tachometer), but after probing the magnetoresistive sensor, we determined that the sensor was plenty fast enough. Instead, we ended up using an oscilloscope that could buffer 14,000,000 data points at an arbitrary rate, downloaded the scope grab and manipulated the data in python and matlab.

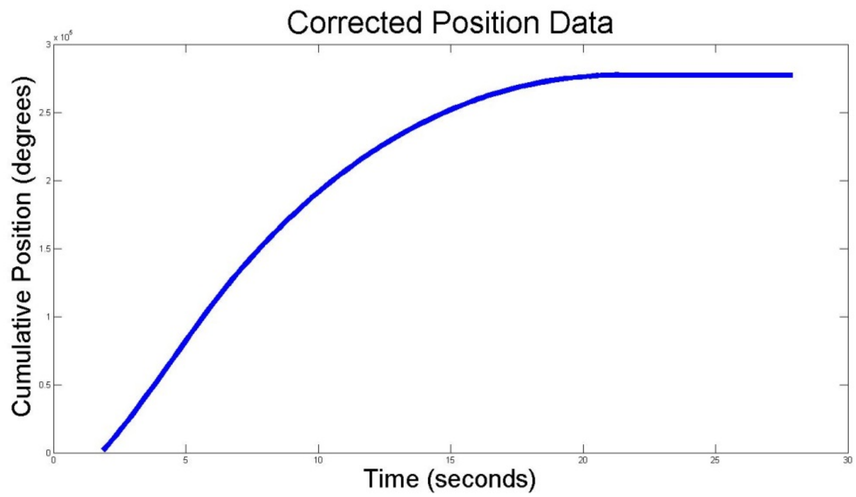
We also made additional hardware for the spindown test- the haptic joystick has the motor mounted so close to the ground that there is no room to add a significant flywheel. We designed and printed a stand that holds the motor vertically (such that nothing obstructs a flywheel), then added a larger disk with a weight near the edge to make the spindown last longer. Below are photos of the spindown jig. The flywheel is in blue. The top green bar holds the magnetoresistive sensor.



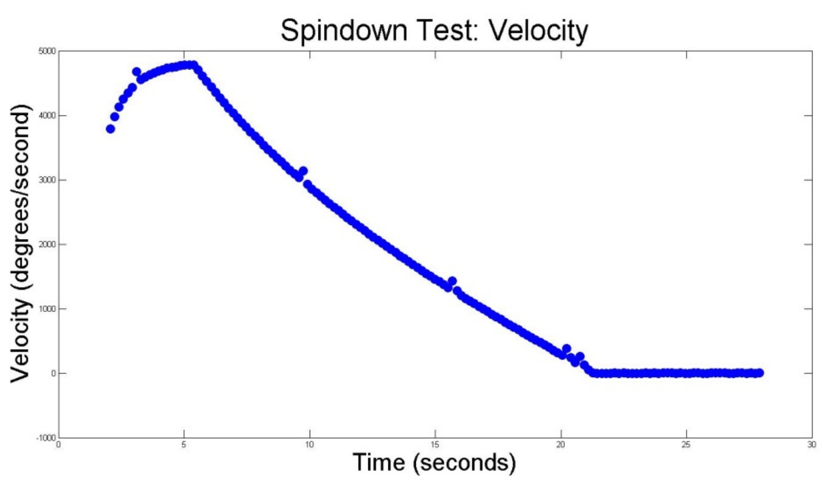




This graph shows the raw scope grab from the magnetic sensor. It is difficult to read because of the flips every 180 degrees.

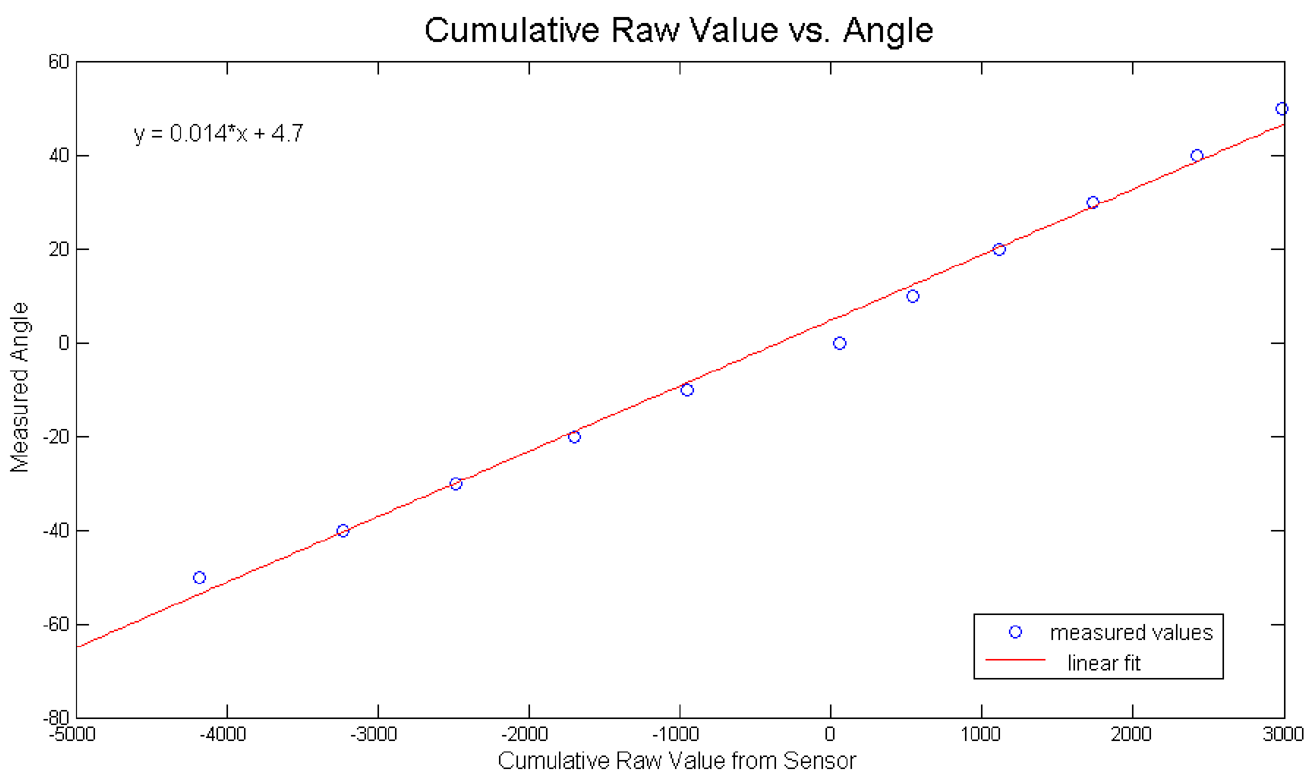


This is the positional data after the 180 flips have been accounted for.



This graph shows the spindown velocity. You can see the exponential decay rise at the beginning as the motor reaches full speed, then the slower exponential decay as the motor spins down. Note that the exponential decay is cut off before it reaches its asymptote. With a larger flywheel (more kinetic energy), it would get closer before flatlining at 0. Also note the slight error in a few places. Due to the periodic nature of this disturbance, and the lack of correlation between motor events and graph error, we believe this to be an artifact from the scope's buffering system.

Finally, we made a calibration plot that allows us to correlate raw readings from the microcontroller's ADC with actual angles of the joystick. The sensor readings natively range from 0 to 1024 (although our sensor only uses ~80% of this range), and increment every time the motor shaft flips 180 degrees (not the joystick). Therefore, it makes sense to have a mathematical formula for converting from raw reading to degrees.



We also wrote code to periodically measure the sensor, keep track of 180 degree flips in both directions, and control the motor using PWM and the motor shield. Below is a link to our github repository with all of our source code. Most of the code (and raw data/ data processing scripts) for this project can be found under the hapticPaddle or MiniLab1_Report subfolders

GitHub repository: https://github.com/cariley/elecanisms