Improving end joint force sensing for robotic grasping (Using IMUs to estimate contact force)

Nao Ouyang

Rotation in Harvard Biorobotics Lab

This is my final report for my 299r rotation during Spring 2018 in Prof. Robert Howe's lab.

Abstract. *Despite decades of research, robots remain unable to reliably grasp objects in environments not designed for robots. Thus, they are not able to work alongside us in human environments or outside buildings. One approach is to use tactile sensing to predict whether a grasp will succeed or not as well as build simulations to evaluate potential grasps. In this lab, we use an underactuated three-fingered gripper, which makes it difficult to estimate the location of the fingers and and the location of the force with respect to the robot frame of reference. I investigated the use of an inertial measurement unit to provide such information. By characterizing the stiffness matrix K of the finger, I could investigate whether using an inexpensive inertial measurement unit (IMU) to estimate θ deflection with the linear equation $\tau = K\theta$, I could discover information about the force applied. I did very preliminary work, limiting my experiments to the case where we only desire to estimate the change from directly before to second or two after contact. Additionally, I looked at using of off-the-shelf machine learning approaches to reduce the errors in our model. Finally, I investigated adding more sensors to the fingertip in order to better estimate contact location.*

1. Introduction

State-of-the art robots are bad at grasping in unstructured environmnents, limiting their use to structured environments like factory floors. An active area of research in order for robots to expand to unstructured environments (such as kitchens or hospital environments), is the ability to grasp novel objects (objects the robot has not encountered before and may not have any prior information about). Grasp stability analysis plays a critical role in enabling robots to achieve high grasping success rates (in terms of not dropping objects). Through sensing, for instance tactile sensing, we can predict whether or not we will drop the object after grasping and prior to moving. If we find our grasp to be unstable, and assuming the object has not moved during grasping such that releasing our grip might e.g. cause it to tumble, we can simply release our grasp and try again.

Physical model approaches involve modeling the contact forces, contact location, and surface normals (which will differ from the force XYZ if the surface(s) are squishy). However, over the last few decades physical models have proven insufficient. This in part due to complex gripper-object interactions during grasping which are both difficult to measure and hard to model. Thus, one of the aspects of this investigation is to evaluate whether machine learning can be used to improve physical models. We can combine the good aspects of physical models (good generalizability, requiring significantly

less data than in pure computer vision machine learning approaches) with (not having to have extremely precise models). I tried to charactize the residuals in the physical models and determine if these residuals could be more accurately modelled by a black box machine learning approach rather than making the physical model more complex. This is an increasingly popular approach, which makes intuitive sense. We can leverage domain knowledge, instead of using an end-to-end approach for sensor data to stability prediciton.

Although there are a few compact 6 axis force torque sensors, they cost multiple thousands of dollars and are finicky and fragile. Thus, we would like to use an inexpensive nine degree-of-freedom (three-axis acceleromter, gyrometer, and magnetic heading) IMU instead.

During the course of this rotation, senior graduate student Qian Wan discoverd that the current tactile sensors are insufficent for estimating contact location accurately enough to develop a good grasp stability estimate. Thus, this project includes a brief exploration of a new sensor design using the same elements (barometric pressure sensors covered by a layer of elastomer, such as urethane), but more of them.

2. Theory

2.1. Model

For the model, we are making a few simplifying assumptions:

1. There is a linear relationship between force and deflection, or alternatively torque and angle of deflection 2. The center of the axis of rotation was at the tip of the proximal joint (in reality, likely it is two or three mm shifted outward, and changes as the flex increases) 3. We consider the z axis to be defined respect to the surface of the finger, so that z=0 is always at the tip of the finger

The consequence of #3 is that we cannot estimate z-axis torque.

2.2. Reference Frame

```
Coordinates
----> + x (roll)
|
|
v
+ y (pitch)

+ z (up out of page) (yaw)


Finger Positions
=======      ====================.
 {CL}  || --- || 15  12  9  6  3 ||
 {A} [xy=0]-- || 14  11  8  5  2 ||
 {MP}  || --- || 13  10  7  4  1 ||
=======      ====================.


  [[---]]
||  WEB  ||
```

```
||  CAM  ||
  ||--- ||

Points 13 through 15 are at x=2.6 cm, and each x is spaced 5 mm
    apart.
```

2.3. Linear Model, 1D case

In the one-axis case, the math is straightforward.

$$\tau = k \times F \tag{1}$$

For example, one datapoint might be

$$k = F/\tau \tag{2}$$
$$k = (20g \times 9.8m/s^2)/0.1 \text{ degrees} \tag{3}$$

where ‘k‘ represents the stiffness of the finger. Let c represent the inverse of k.

Using least squares error, we may fit a line to find $\hat{c}$.

$$\hat{c} = \frac{1}{\hat{k}} \tag{4}$$
$$\theta = \tau\hat{c} + b \tag{5}$$

where ‘b‘ be a constant determined by the line of fit.

2.4. Residuals

From the above, we may calculate the residuals of any of our estimated variables. For instance, from our actual data we may obtain an estimate for ‘k‘.

$$\tau_{data} = \hat{k} \cdot \theta_{data} \tag{6}$$
$$\tag{7}$$

Using this k we can go back and calculate estimates for the ”true”torque, assuming our linear model was correct.

$$\hat{\tau} = \hat{k} \cdot \theta_{data} \tag{8}$$

We would then calculate our torque residuals as

$$\varepsilon_\tau = \hat{\tau} - \tau \tag{9}$$

If we plot a graph of (torque residuals) vs (estimate residuals) and find that our points are randomly scattered around a straight line, then our model well-approximates reality as sensed by a noisy sensor.

However, our residuals may instead follow a parabola, in which case we would want to amend our model to have higher-order terms.

$$\hat{\tau} = \hat{k}\theta_{data} + c_1\theta_{data} + c_2\theta_{data} \tag{10}$$

and so forth.

We may eventually use machine learning techniques to fit higher-order terms.

2.5. 3D case

The 3d case is exactly the same, except now each of the variables are vectors or matrices.

$$\left[\vec{\tau}\right]_{3xn} = \left[K\right]_{3x3} \cdot \left[\vec{\theta}\right]_{3xn} \tag{11}$$

where

$$\vec{\tau} = \begin{bmatrix} \overset{3x1}{\tau_1} \mid \overset{3x1}{\tau_2} \mid ... \mid \overset{3x1}{\tau_n} \end{bmatrix} \tag{12}$$

$$\tag{13}$$

$$\vec{\theta} = \begin{bmatrix} \overset{3x1}{\theta_1} \mid \overset{3x1}{\theta_2} \mid ... \mid \overset{3x1}{\theta_n} \end{bmatrix} \tag{14}$$

2.6. Sanity Check: Simplify to 2D case

We know that $\tau = r \times F$

$$\left[\tau_x \ \tau_y \ \tau_z\right] = \left[K \right] 3x3 \ \left[\theta x \ \theta_y \ \theta_z\right] \tag{15}$$

Further, we can use some of the simplifying assumptions we made about to model our system, in order to have an idea of what values our math should result in for k.

We may also see that we will only ever have x and y components for r; z components for F; and thereforce (by how cross products work) only ever have x and y components for τ.

Off-axis

$$r \approx \begin{bmatrix} r_x \\ r_y \\ 0 \end{bmatrix} \times f \approx \begin{bmatrix} 0 \\ 0 \\ f_z \end{bmatrix} = \tau = \begin{bmatrix} \tau_x \\ \tau_y \\ 0 \end{bmatrix} \tag{16}$$

$$\tag{17}$$

On-Axis

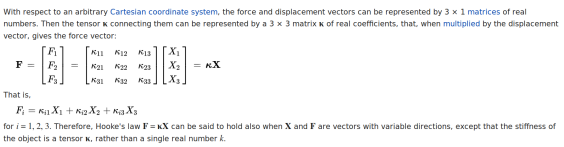
In the even more simplified case, if we do not apply the force off-axis and there is only a pitch deflection

$$r \approx \begin{bmatrix} r_x \\ 0 \\ 0 \end{bmatrix} \times f \approx \begin{bmatrix} 0 \\ 0 \\ f_z \end{bmatrix} = \tau = \begin{bmatrix} 0 \\ \tau_y \\ 0 \end{bmatrix} \tag{18}$$

$$\tag{19}$$

We **could** have enough nonzero values for the xyz components of θ that we will be able to calculate a nondegenerate estimate for k. This is thanks to applying the force off-axis, causing the finger to roll and create non-zero elements for θ_y (our roll) and θ_z (our yaw). However, after solving for K, I found that I ended up with a bottom row consisting entirely of zeroes.

There is a good summary on wikipedia of the relevant linear model, which I enjoyed.



With respect to an arbitrary Cartesian coordinate system, the force and displacement vectors can be represented by 3 × 1 matrices of real numbers. Then the tensor $\boldsymbol{\kappa}$ connecting them can be represented by a 3 × 3 matrix $\boldsymbol{\kappa}$ of real coefficients, that, when multiplied by the displacement vector, gives the force vector:

$$\mathbf{F} = \begin{bmatrix} F_1 \\ F_2 \\ F_3 \end{bmatrix} = \begin{bmatrix} \kappa_{11} & \kappa_{12} & \kappa_{13} \\ \kappa_{21} & \kappa_{22} & \kappa_{23} \\ \kappa_{31} & \kappa_{32} & \kappa_{33} \end{bmatrix} \begin{bmatrix} X_1 \\ X_2 \\ X_3 \end{bmatrix} = \boldsymbol{\kappa}\mathbf{X}$$

That is,

$$F_i = \kappa_{i1}X_1 + \kappa_{i2}X_2 + \kappa_{i3}X_3$$

for $i = 1, 2, 3$. Therefore, Hooke's law $\mathbf{F} = \boldsymbol{\kappa}\mathbf{X}$ can be said to hold also when $\mathbf{X}$ and $\mathbf{F}$ are vectors with variable directions, except that the stiffness of the object is a tensor $\boldsymbol{\kappa}$, rather than a single real number k.

3. Methods

3.1. Setup

For the setup, I used a triple-beam balance as a way to apply force while retaining a vertical angle. I hot-glued a pumpkin awl to the bottom of the plate and put standard weights on top of of the scale. This caused the awl to contact the surface of the finger, and the finger would then deflect.

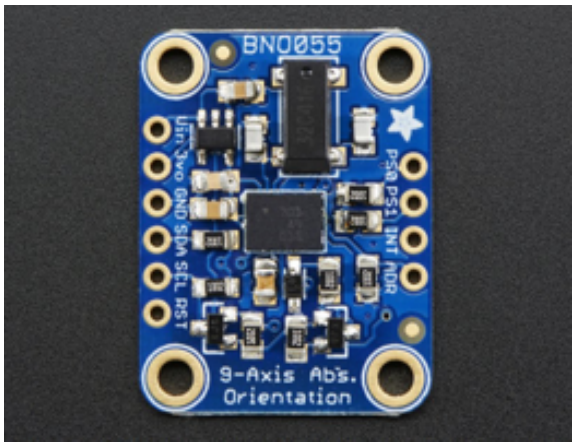
In the 1D case, I simply placed a ruler behind the finger and used a webcam to take pictures. By mapping pixel counts to millimeters, I could then estimate the deflection of the finger. The linear fit from these very imprecise measurements proved linear enough that I then relied on the IMU going forward and skipped the step (which should be completed eventually) of characterizing the accuracy of the IMU versus a gold standard, such as a stereo camera (the Microntracker available in lab) or other methods (such as using Apriltags with a calibrated webcam).

I then created a grid of 15 points on the finger. For each of the 15 points, I applied several known amounts of force, with three samples at each force-point combination. Each sample consisted of a ”zero”measurement pre-contact and a measurement post-contact. In post-processing, I then subtracted the two to obtain my final sensor data.

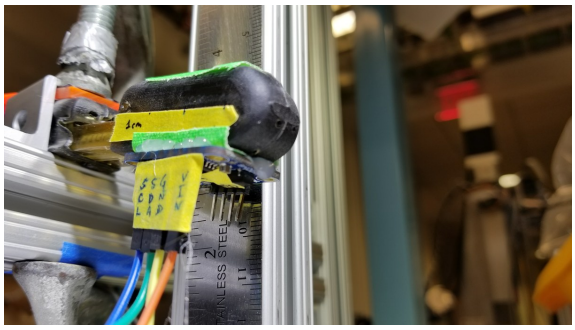
Complications arose from the limited travel of the triple beam balance, which meant I could only measure small deflections (initially on the order of 4 mm; after disco-

vering that I could take the end stop off the back edge of the triple beam balance, about 10mm). At the tip of the finger, I could only less than a hundred grams of "force"(about 1 N), while 4N is a normal amount of force applied by humans to grasp things. This remained a source of frustration throughout the data collecting and ate up many hours of adjusting the experimental setup.

I glued an IMU to the undersid of the finger. The IMU I used was the Bosch BNO055 sensor. A breakout was available from adafruit.com for $35, which is relatively inexpensive. I chose this sensor not only for the built-in processing, and also because of the extensive beginner-friendly documentation developed by Adafruit for all of its products. It was recommended to me by at Right Hand Robotics, which is a spinoff of this lab.

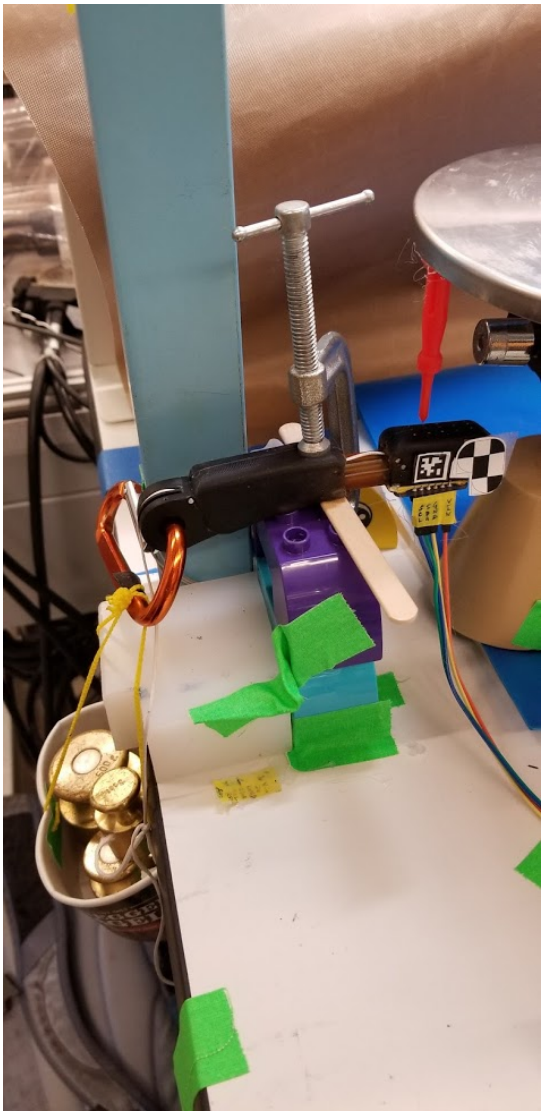


"Bosch is the first company to get this right by taking a MEMS accelerometer, magnetometer and gyroscope and putting them on a single die with a high speed ARM Cortex-M0 based processor to digest all the sensor data, abstract the sensor fusion and real time requirements away, and spit out data you can use in quaternions, Euler angles or vectors."(source: adafruit.com)

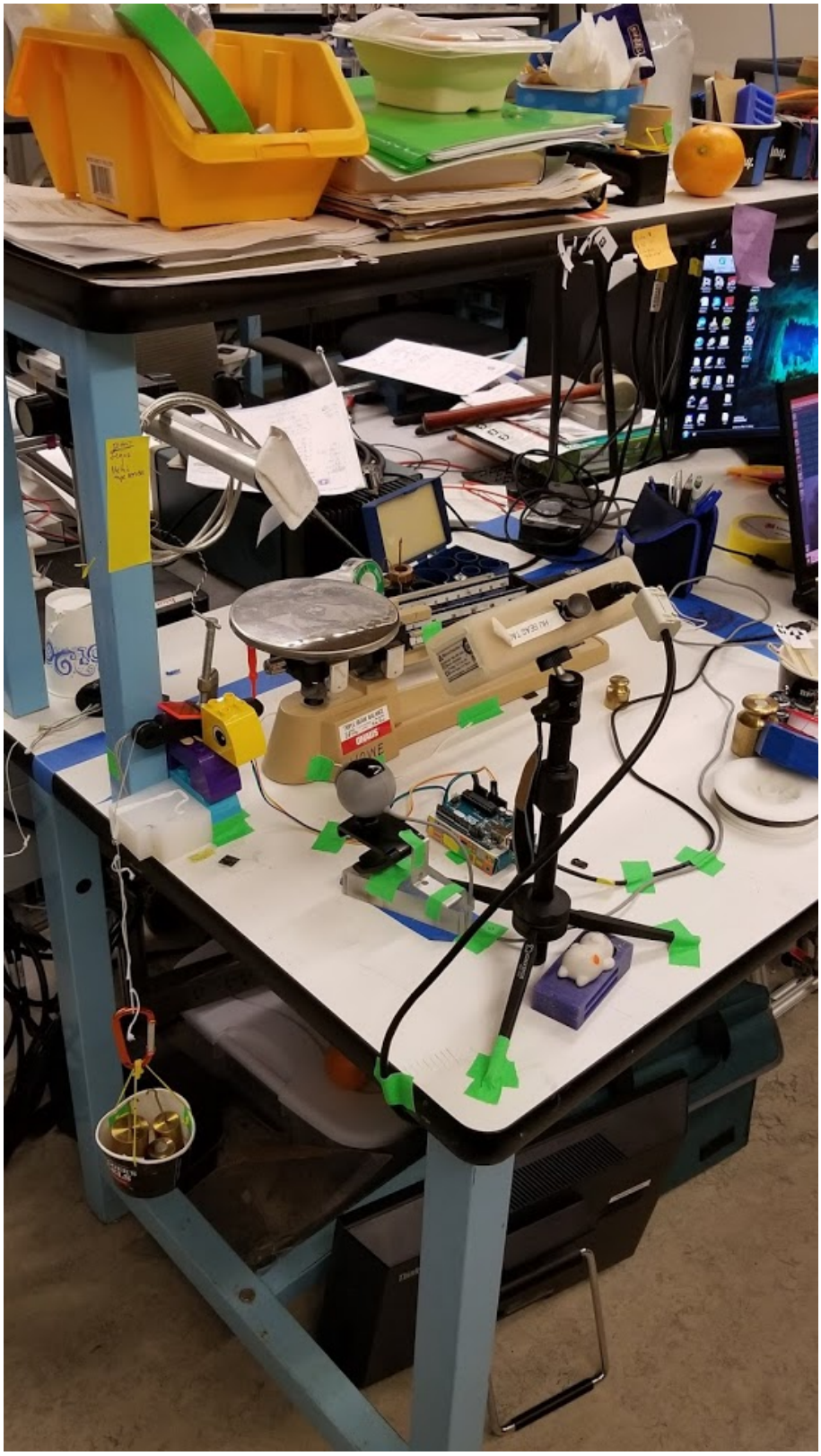


I also attached an apriltag (which looks like a QR code) and a micontracker tag (which has a checkerboard like cross), both of which can simply be printed out on paper, and calibrated the webcam for use with the Apriltag C++ script. Although I have collected mictrontracker and apriltag data for comparing to the IMU deflection data, I have not had a chance to analyze this data yet,

Additionally, the IMU acclerometer would consistently refuse to calibrate if the sensor did not start parallel to the ground. The significant slope of the finger versus the clamp and 90 degree angle of the 80/20 meant that I had to approximate a slanted 80/20 setup by increasing the degrees of freedom on the L brackets (removing t-nuts). This led to a lot of worry on my part about the creakiness and variability in the data collection as I could not guarantee angle consistency, although in retrospect this is almost completely mitigated by the fact that I am zeroing before every reading.



In order to apply load to pull on the tendon and change the stiffness of the finger, I decided on a simple setup where I hung a weight from the tendon.



As I progressed, I trended toward collecting coarser and coarser data as I grew confident that measurements still provided enough data for me to see overall trends in the data (such as the linearity of the data). I shifted from 2g increments, to 20g, and in the end

50g increments. This meant that for the finger when stiffness was applied, I had to change the setup so that the finger started at an angle and became parallel to the floor when load was applied to the tendon. After my deep grievances with producing angles on the 80/20 setup, I decided to use legos with popsicle stick props on one end in order to produce the correct angles.

Finally, I discovered at the very end that I could calibrate the scale to have a constant force offset. Thus, instead of collecting a zero measurement with the tip starting just above the surface, which meant that I had to wait for the oscillations to settle, I could put an offset which made the scale keel all the way to up when the weight was removed. This made zero measurements 2-3x faster, which reduced the cognitive load to remember which measurement I was at, which both reduced data cleanup and needing to recollecting data. This led to significant time savings.

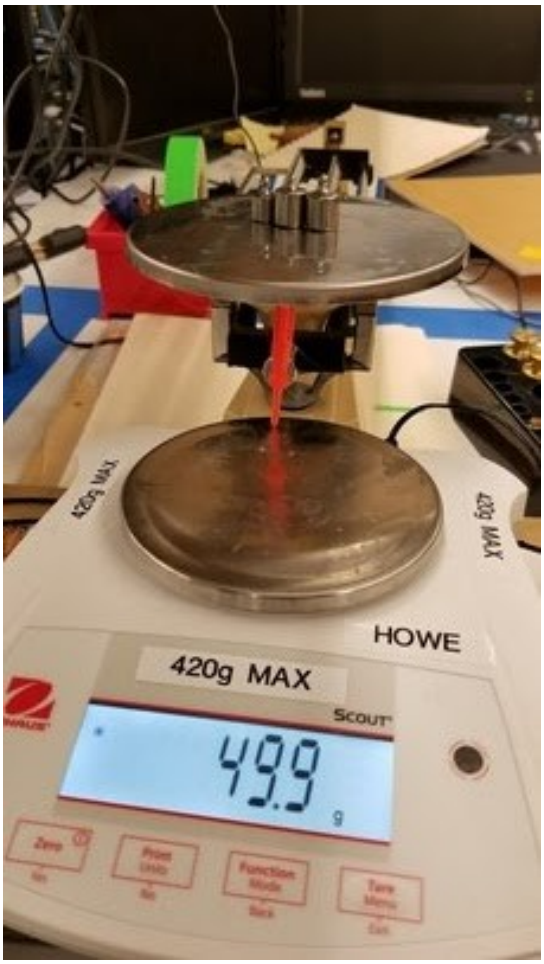


Figura 1. Scales and more scales: calibrating the weight applied with the triple beam balance using an electronic scale

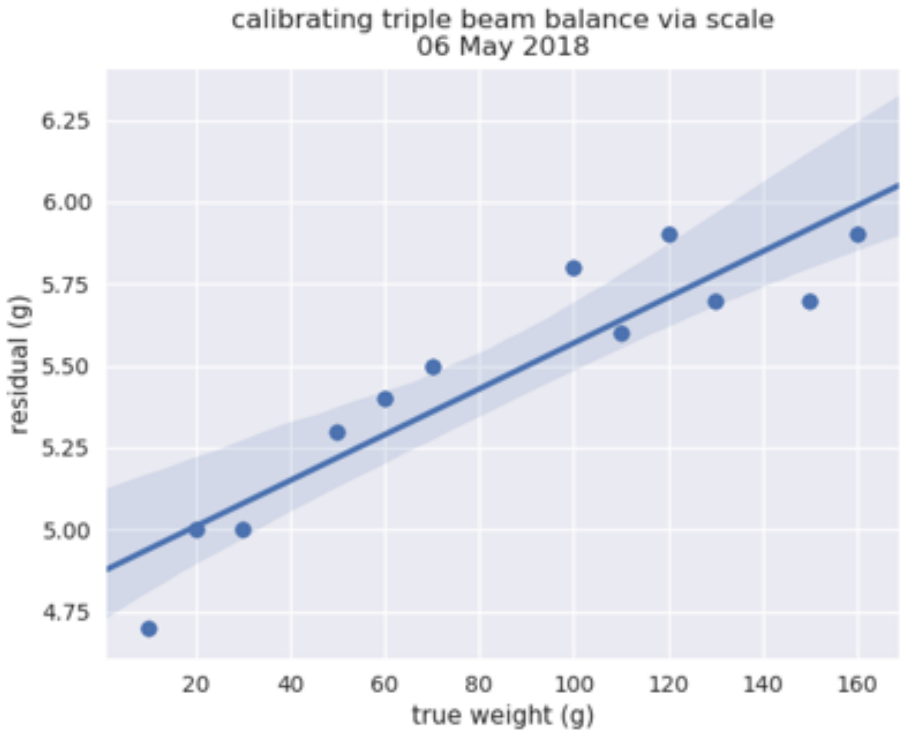


Figura 2. Scale calibration graph, with nominal offset of 5g. I observed that the offset is not constant and increases with increasing load.

4. Data Analysis

4.1. Loaded tendon data

For the final dataset, I collected data in 50g intervals at 2 y locations and 3 x locations on the finger (total of 6 positions).

First, a quick sanity check. By plotting torque versus measured theta, we see that the data follows a linear pattern, as we would expect.

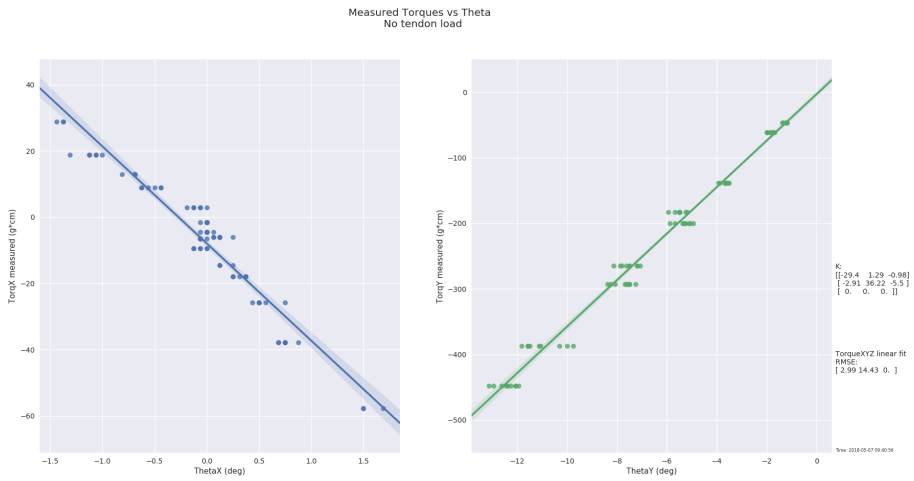


Figura 3. Relaxed tendon

Figura 4. Loaded tendon

Additionally, we may see that we have properly fitted our data with a K matrix by comparing our newly created torque estimates, versus the original measurements. The datapoints do not fall precisely on a line because the K matrix and fit is in three-dimension. In picking a single dimension to plot, the projection results in datapoints that are not linear in one dimension despite a linear fit overall.

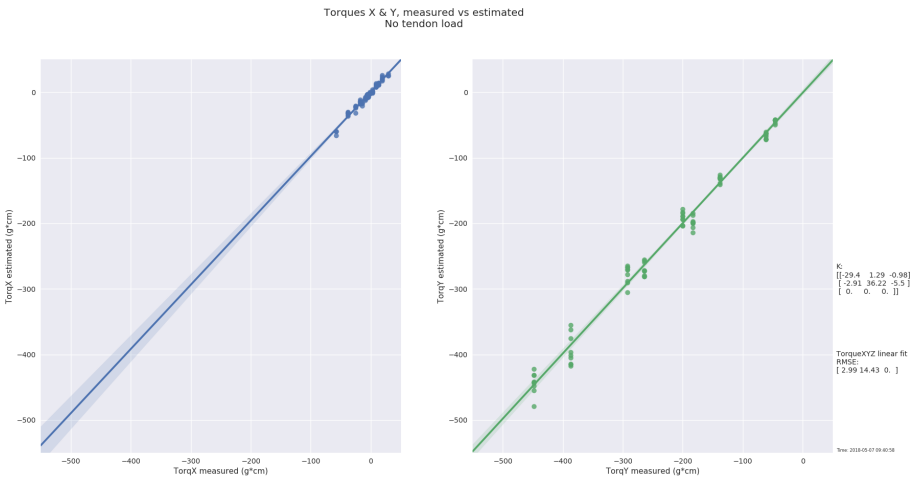


Figura 5. Relaxed tendon

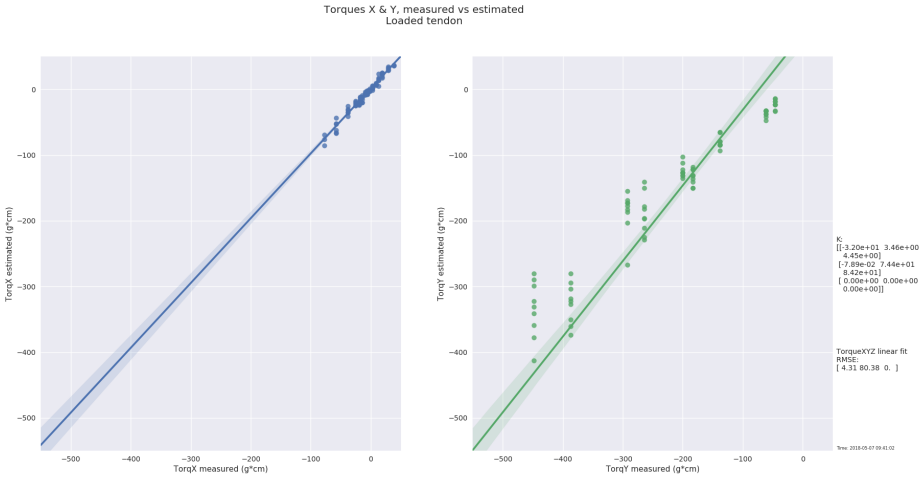


Figura 6. Loaded tendon

Next we can look at the residuals. Based on the previous experiment, I was expecting to see a linear relationship in the residuals between ThetaY and TorqueY. However, I did not see any such relationship in the relaxed case, which should have most closely matched the data in the previous experiment.

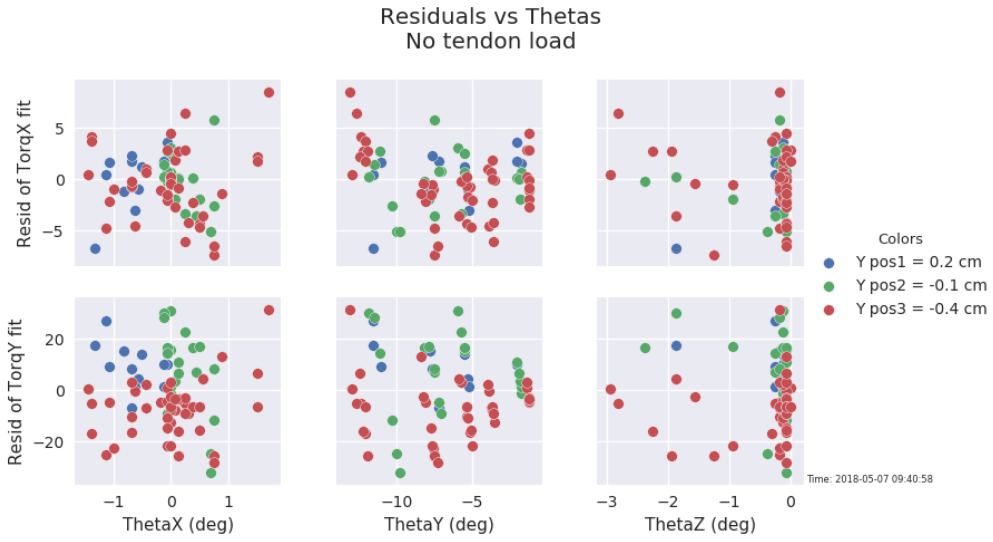


Figura 7. Relaxed :tendon

A pattern did appear in the loaded case. Further investigation will be required.

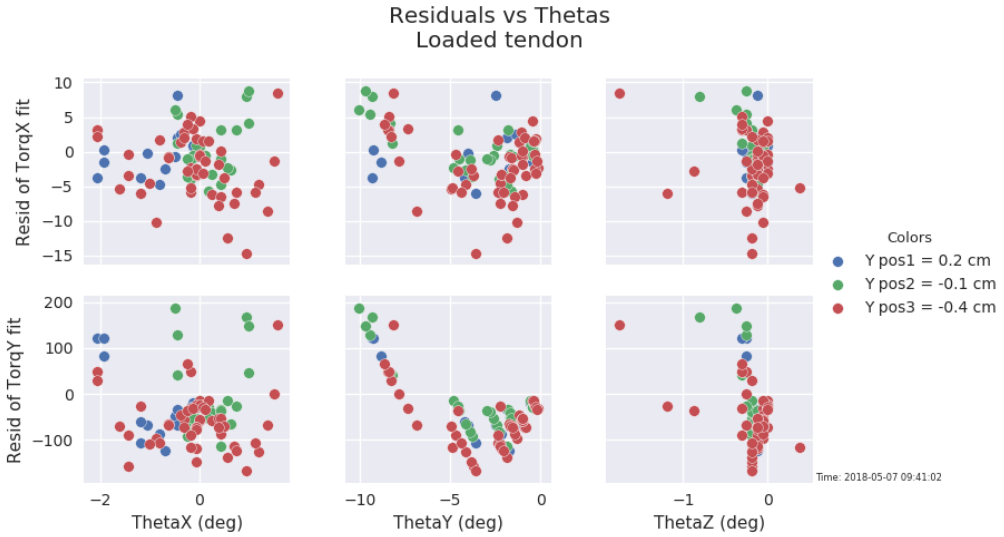


Figura 8. Loaded tendon

Patterns in the thetaZ may be ignored as it was not used in the fit (zeroed out by zero for Force)

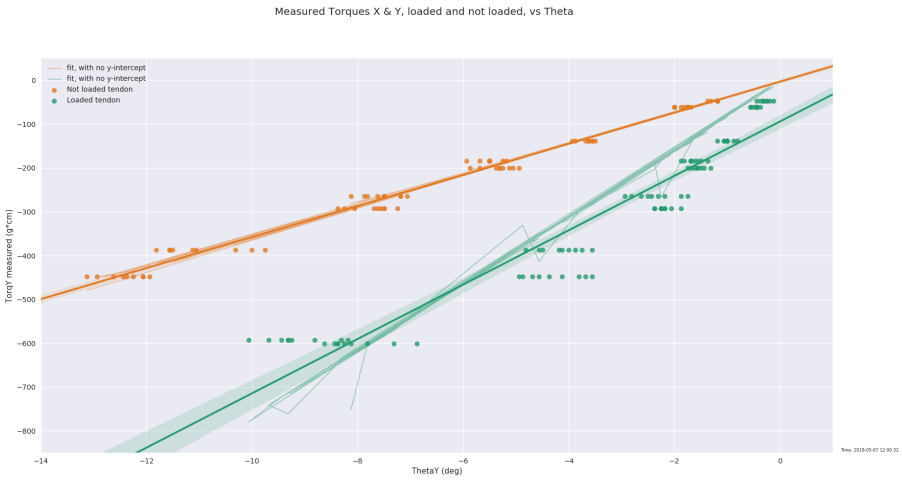
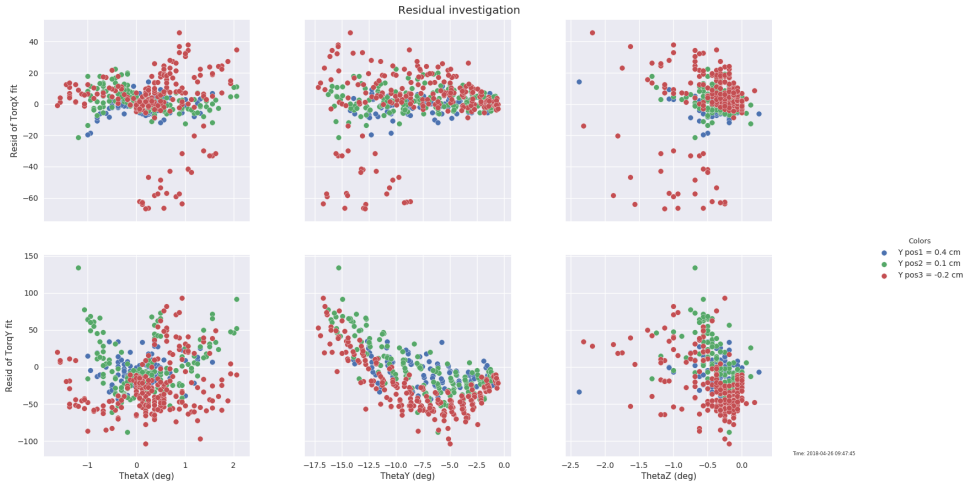
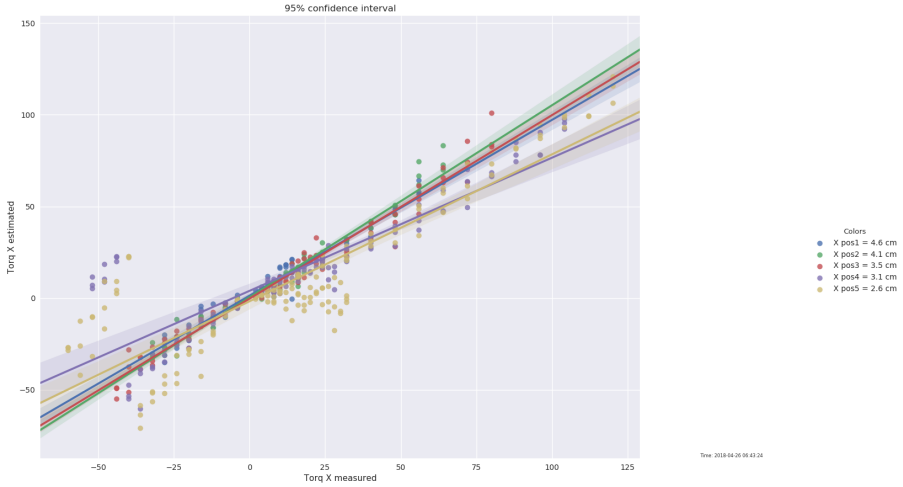
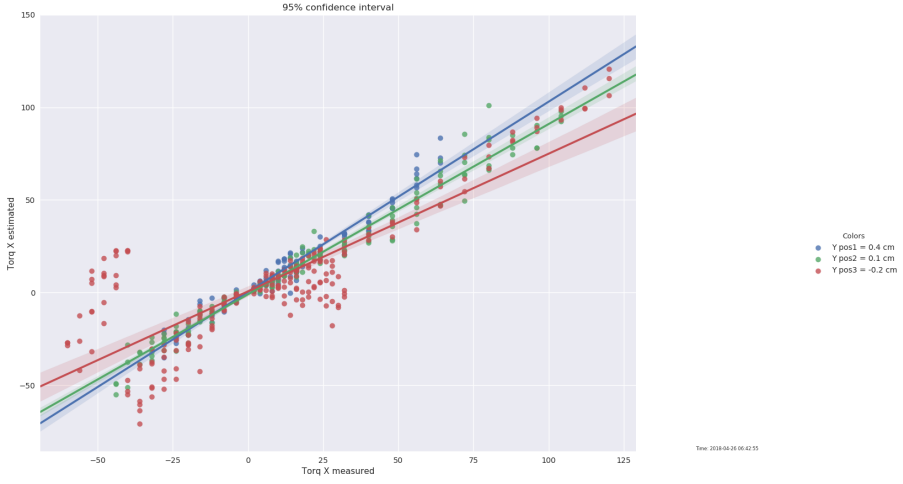


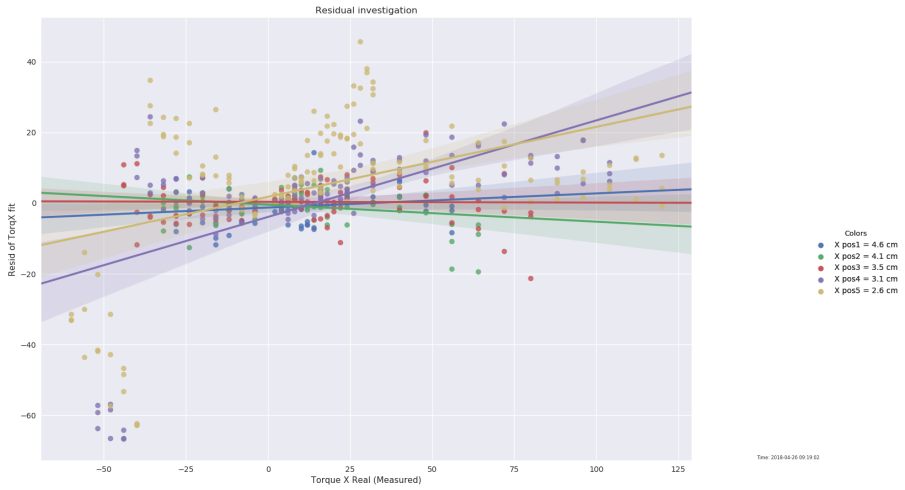
Figura 9. Loaded case shown in green, relaxed tendon shown in orange. This graph shows that the loaded tendon is stiffer (slope is steeper), as we expect. Both models seem mostly linear, but it's possible that in the loaded case, the fit is more exponential. Note that physically the model must go through the y-intercept, and I have graphed the actual fit used to calculated residuals in light green (albeit for a nonlinear fit). The dark green line is plotted due to limitations in my plotting library

4.2. Initial data (relaxed tendon only)









4.3. 1D case

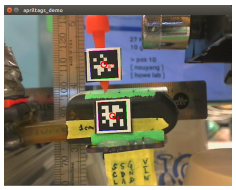


Figura 10. Zeroed

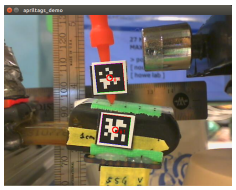


Figura 11. Weighted Loaded tendon

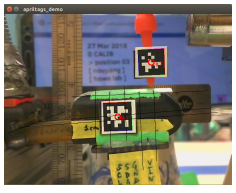
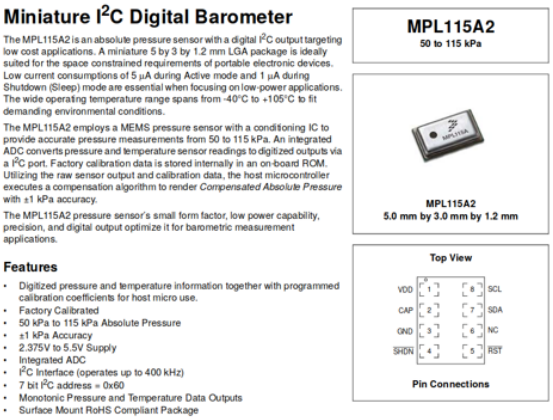


Figura 12. Referencing the ruler in order to estimate location of force applied, as well as angle of displacement.

4.4. Attempted fit improvements

5. Sensor Design

MLP



Miniature I²C Digital Barometer

The MPL115A2 is an absolute pressure sensor with a digital I²C output targeting low cost applications. A miniature 5 by 3 by 1.2 mm LGA package is ideally suited for the space constrained requirements of portable electronic devices. Low current consumptions of 5 μA during Active mode and 1 μA during Shutdown (Sleep) mode are essential when focusing on low-power applications. The wide operating temperature range spans from -40°C to +105°C to fit demanding environmental conditions.

The MPL115A2 employs a MEMS pressure sensor with a conditioning IC to provide accurate pressure measurements from 50 to 115 kPa. An integrated ADC converts pressure and temperature sensor readings to digitized outputs via a I²C port. Factory calibration data is stored internally in an on-board ROM. Utilizing the raw sensor output and calibration data, the host microcontroller executes a compensation algorithm to render *Compensated Absolute Pressure* with ±1 kPa accuracy.

The MPL115A2 pressure sensor's small form factor, low power capability, precision, and digital output optimize it for barometric measurement applications.

Features

- Digitized pressure and temperature information together with programmed calibration coefficients for host micro use.
- Factory Calibrated
- 50 kPa to 115 kPa Absolute Pressure
- ±1 kPa Accuracy
- 2.375V to 5.5V Supply
- Integrated ADC
- I²C Interface (operates up to 400 kHz)
- 7 bit I²C address = 0x60
- Monotonic Pressure and Temperature Data Outputs
- Surface Mount RoHS Compliant Package

MPL115A2
50 to 115 kPa

MPL115A2
5.0 mm by 3.0 mm by 1.2 mm

Top View

VDD 1 8 SCL
CAP 2 7 SDA
GND 3 6 NC
SHDN 4 5 RST

Pin Connections

Newer (smaller footprint)

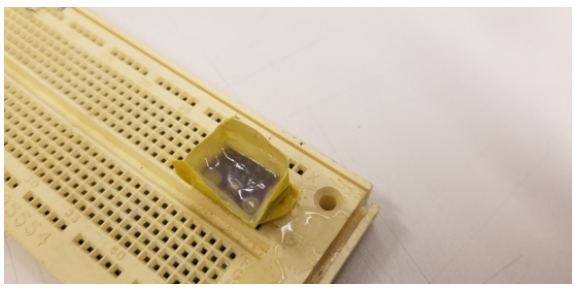
BMP280

Digital Pressure Sensor

Key parameters

Pressure range	300 ... 1100 hPa (equiv. to +9000...-500 m above/below sea level)
Package	8-pin LGA metal-lid Footprint : 2.0 × 2.5 mm², height: 0.95 mm
Relative accuracy (950 ... 1050hPa @25°C)	±0.12 hPa, equiv. to ±1 m
Absolute accuracy (950 ...1050 hPa, 0 ...+40 °C)	typ. ±1 hPa
Temperature coefficient offset (25 ... 40°C @900hPa)	1.5 Pa/K, equiv. to 12.6 cm/K
Digital interfaces	I²C (up to 3.4 MHz) SPI (3 and 4 wire, up to 10 MHz)
Current consumption	2.7μA @ 1 Hz sampling rate
Temperature range	-40 ... +85 °C
RoHS compliant, halogen-free	
MSL 1	

Prototype



I produced Arduino and python code that allowed me to view a plot in real time of the sensor data.

Additionally, I integrated the Arduino with an RGB LED strip, where if you pressed on the sensor, more LEDs would light up.

The sensor displayed a large amount of hysteresis, which should be investigated and addressed.

This required use of a separate power supply, otherwise the Arduino would refuse to talk to the computer (for real-time plotting) when the LED strip drew load on the 5V line.

6. Miscellaneous

6.1. Online documentation

I documented my work on a blog, http://orangenarwhals.github.io/, using the Hexo flatfile content management system. I chose this because it allowed me to host for free on github, allowed for easy version control of content, and also had an admin plugin that allowed me to drag-and-drop images, which is crucial for documenting experiments where I have a lot of graphs. I was also able to get Latex to work on the site so that I could have equations rendered on there.

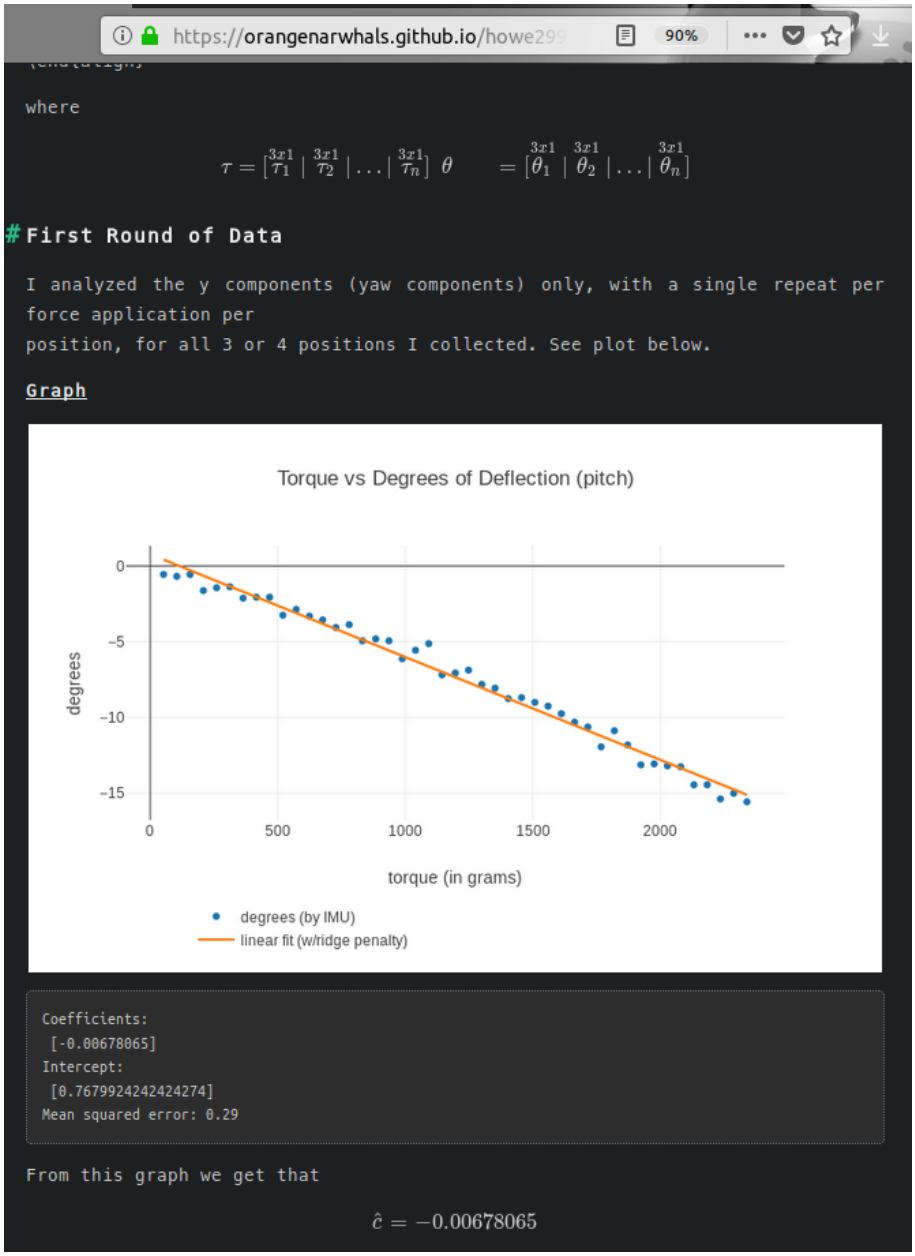


Figura 13. Getting latex to work was a major source of frustration.

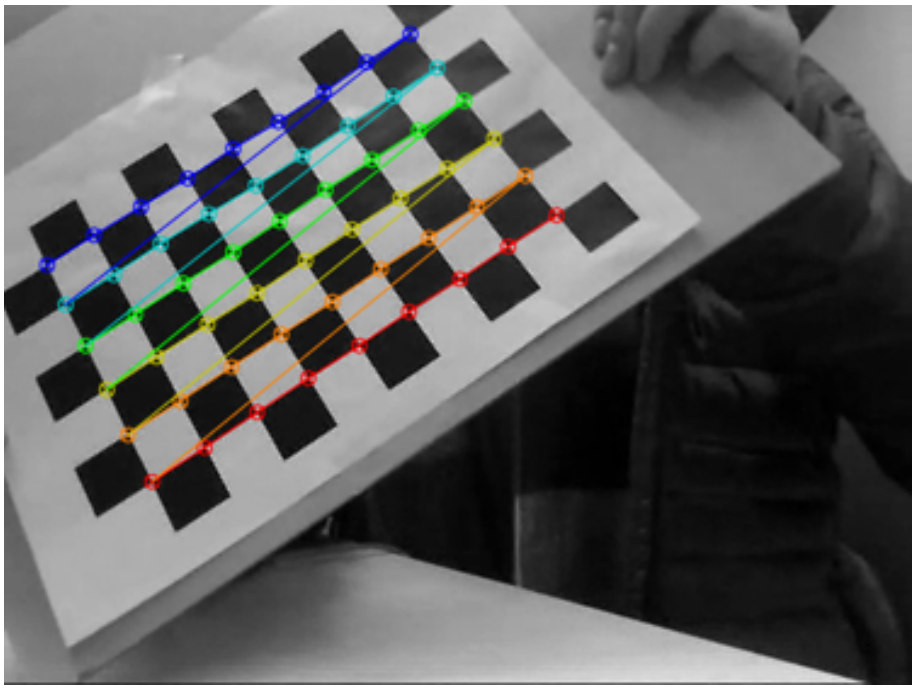
6.2. Apriltags

I decided to investigate apriltags, which helped me refresh on my knowledge of C++. I struggled to find an appropriate ROS driver, and ultimately was able to get the C++ working (despite some issues with the openCV version on my laptop) with the help of Patrick from Prof. Kuindersma's lab.

```
nrw@earlgrey:~/projects/apriltags$ vi CMakeLists.txt
    (line 14)
        find_package(OpenCV 2.4.9.1 EXACT REQUIRED)
```

Fig. 1: Applications and users of AprilTags. From top left, clockwise: robot-to-robot localization and identification on MAGIC robots, object localization for Boston Dynamics' Atlas robot, testing of virtual reality headsets at Valve, and tracking individual ants to study their social organization [4].

Figura 14. Loaded tendon



This gave estimates

```
public:

  // default constructor
  Demo() :
    // default settiwgs, most can be modified through command
       line options (see below)
 [...excerpted section...]
    m_width(640),
    m_height(480),
    m_tagSize(0.00944), // in meters
    m_fx(667), // in pixels
    m_fy(666), //
    m_px(344), // principal point
    m_py(227),
```

6.3. Time estimates

An additional yet important thing in any project is measuring project slip in the hopes of developing better time estimates.

299r Rotation: Harvard Biorobotics Lab
Spring 2018

Force Estimation with Inertial Measurement Units and with Applications of Machine Learning

Milestones

Week starting on Monday

1. **12 Feb**
   1D – Literature search, setup, collect preliminary data for F = kx (1st order model), in one axis of reflection (no twist of finger – directly up and down)
2. **19 Feb**
   1D – Analyze data, improve with machine learning
3. **26 Feb**
   3D – Collect off-axis deflection data (with twist). Test assumptions of linearity – A. Fix spot and change force, see if deflection angle changes B. Change spot, given same force, how much does the finger deflect at different points along the finger?
4. **05 Mar**
   3D – Collect data and analyze (with lower order model, then higher model with machine learning)
5. **12 Mar**
   3D – Analyze and write up conclusions
6. **19 Mar**
   Tendon – Setup for tendon measurements, where finger is curled
7. **26 Mar**
   Tendon – Collect tendon data
8. **02 Apr**
   Tendon – Analyze data, iterate on machine learning
9. **09 Apr**
   Tendon – continue analysis
10. **16 Apr**
    Spacer week
11. **23 Apr**
    Writeup
12. **Monday, 30 April 2018**
    Group presentation

7. Knowledge

In the process of this research, I learned about how accelerometers are calibrated, which I found very interesting (although ultimately, thanks to the built-in sensor fusion and

calibration routines of the BNO055 sensor, I did not need to write my own calibration algorithms). The common presence of the gravity vector across meaurements at diffent angles makes this calibration possible.

I learned the most about exploratory statistical data analysis. Where previously I had not even heard of residuals (only scalar measurements such as mean-square-error), I learned about how they could be compared to a horizontal basis against which they would be randomly distributed assuming random noise around a proper physical model.

Finally, I learned more about the I2C and SPI protocols for receiving (and writing) to multiple sensors. I began work on creating a custom printed circuit board to create a 3x4 array of sensors, and decided to switch and learn to use a newcomer in the PCB design space, which has CERN support: the free and open-source KiCAD tool.

8. Conclusions

The stiffness of the finger was remarkably linear, which meant that machine learning and higher order terms had little to contribute to improving the fit between measure theta deflections and force estimates.

However, I was only able to apply relatively small loads to the finger. It's possible that with large loads, the stiffness becomes less linear, in which case linear fit residuals may become more prominent or structured. If this were the case, machine learning approaches might have more to contribute to the force estimates. As it were, the data I collected did not benefit much from the addition of non-linear models fitted with machine learning techniques such as random forests.

Most crucially, Qian demonstrated to me the process by which grasps fail. By observing the grasping, I discovered that when the tendon was pulled taught (as needed to grasp anything), the y-axis was now now the stiffest axis! Thus, characterizing the finger with no tendon load applied and primarily in the y-axis direction, while a great learning experience, does not generalize at all to practice and becomes a toy example.

Additionally, the finger relied extensively on mechanical mechanism intelligence (as opposed to full actuated parallel grippers, commonly used in computer science labs). I saw that complete failures tended to occur because the finger had failed to detect contact when contact had already occured. This suggests the better instrumentation of the finger is critical. The IMU by itself can only suggest contact force, and the tactile sensors are needed to tell contact location. To evaluate the contribution of the IMU to force estimates and not just torque estimates, we must build better location sensors than the current 1x4 array.

8.1. Future Work

The microntracker should be used to evaluate the IMU. Evaluating the use of Apriltags with the microntracker would help reproducibility by others (or future grad students), as webcams are cheap and readily available, and documentation for apriltags is freely available online.

Additionally, a full sensor array on a custom PCB, manufactured into a finger unit, should be developed and characterized. By collecting data and then fusing this with the IMU data, we could physically evaluate the grasp stability preductions, producing an

important tie between theory and practice to ensure the two complement each other. The important of this has been mentioned above. I