

# DOCUMENTATION OF PROGRESS

---

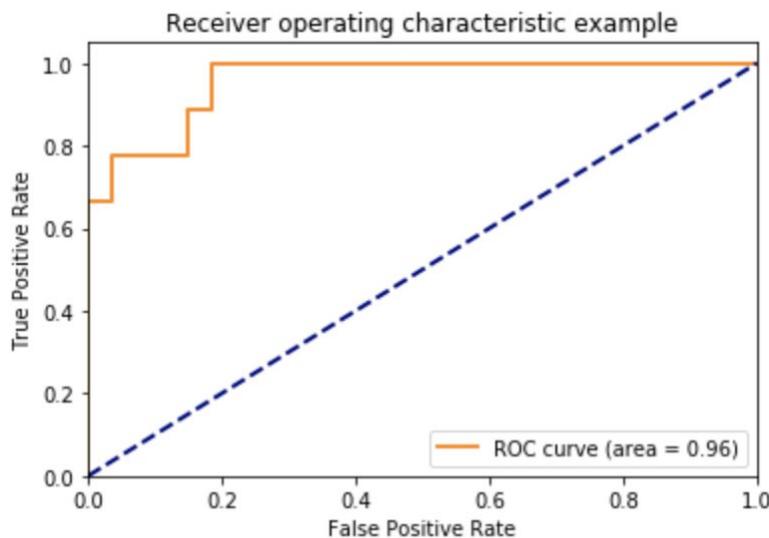
6.10.20

## Progress

- Trying to convert my current ML script into one that is formatted for ROC packages
- Taking data → normalize → “featurify” → take the max of each feature → make 2D array of featured data (ROC uses 2D data whereas I originally had it in 3D, but easy to convert to 2D)
- Following along with the sci kit learn example script:  
[https://scikit-learn.org/stable/auto\\_examples/model\\_selection/plot\\_roc.html](https://scikit-learn.org/stable/auto_examples/model_selection/plot_roc.html)

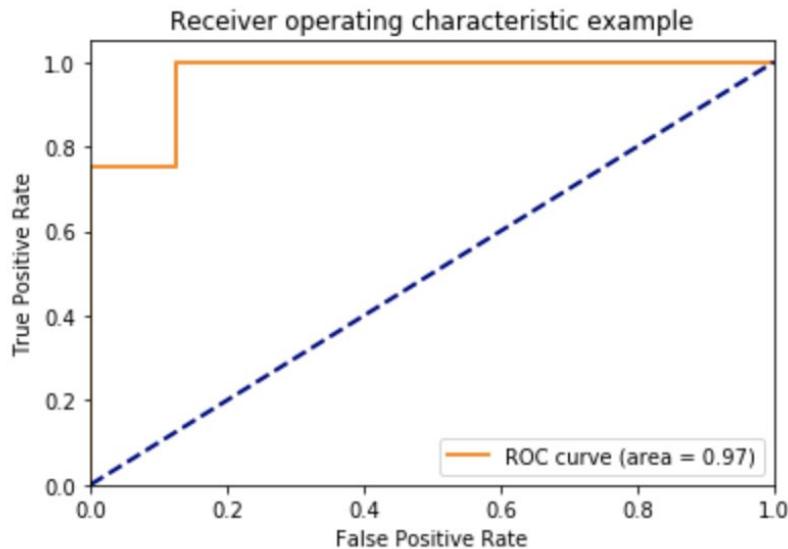
```
In [94]: # shuffle and split training and test sets
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=.5, random_state=0)
```

```
In [95]: # Learn to predict one class against the other
classifier = OneVsRestClassifier(svm.SVC(kernel='linear', probability=True,
                                         random_state=random_state))
y_score = classifier.fit(x_train, y_train).decision_function(x_test)
```

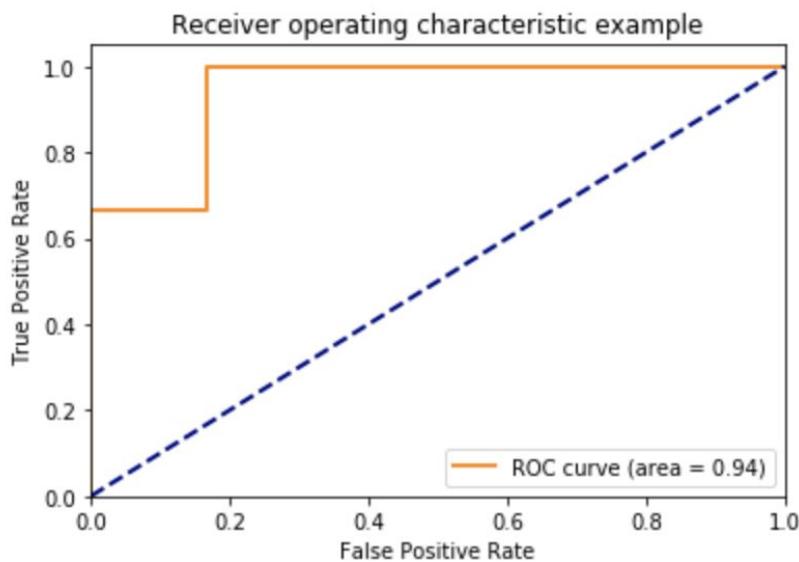


- Got good results, but not super sure what they mean honestly
  - Have Andrew explain more in detail?

- For 12 testing points (instead of half and half above):



- For a testing sample of 9:



- How does it know which are true positive and which are false positive? Does it just check the labels against the results of the model?
- Where can we see the model results? I don't understand the value of `y_score`
- Will probably continue on to try using probability models because those seem like they might be more intuitive to me

- Just kidding! I do understand (sort of) the value of `y_score`. This is for a sample size of 5 tests, and `y_score` is a measure of the probability of each label for each test

```
In [142]: y_score
```

```
Out[142]: array([[ 1.39051113, 16.61217209, -22.44062681, -7.19542744],
 [-36.89831605, 13.32412107, -0.62279804, -20.68490746],
 [-2.16851379, 19.38150573, -13.61203736, -14.72544425],
 [-29.34769553, 8.55962063, -1.18369171, -22.06859798],
 [-27.2132914 , 2.40230902, 6.32252924, -27.48814128]])
```

- Still next, going to look into scripts that return probabilities, not `y_score` because I'm not really sure what the exact values mean
  - I thought it was values closest to 0 but that doesn't make sense for some of them??
- Basic probability script, following along with  
[https://scikit-learn.org/stable/auto\\_examples/calibration/plot\\_calibration\\_multiclass.html#sphx-glr-auto-examples-calibration-plot-calibration-multiclass-py](https://scikit-learn.org/stable/auto_examples/calibration/plot_calibration_multiclass.html#sphx-glr-auto-examples-calibration-plot-calibration-multiclass-py)

```
In [140]: # Train uncalibrated random forest classifier on whole train and validation
# data and evaluate on test data
clf = RandomForestClassifier(n_estimators=25)
clf.fit(x_train_valid, y_train_valid)
clf_probs = clf.predict_proba(x_test)
score = log_loss(y_test, clf_probs)
```

```
In [141]: # Train random forest classifier, calibrate on validation data and evaluate
# on test data
clf = RandomForestClassifier(n_estimators=25)
clf.fit(x_train, y_train)
clf_probs = clf.predict_proba(x_test)
sig_clf = CalibratedClassifierCV(clf, method="sigmoid", cv="prefit")
sig_clf.fit(x_valid, y_valid)
sig_clf_probs = sig_clf.predict_proba(x_test)
sig_score = log_loss(y_test, sig_clf_probs)
```

```
In [142]: clf_probs
```

```
Out[142]: array([[0.4 , 0. , 0. , 0.6 ],
 [0.52, 0. , 0. , 0.48],
 [0.76, 0. , 0. , 0.24],
 [0.44, 0. , 0. , 0.56],
 [0.88, 0. , 0. , 0.12],
 [0. , 0.28, 0.72, 0. ],
 [0. , 0.28, 0.72, 0. ],
 [0. , 0.24, 0.76, 0. ],
 [0.92, 0. , 0.08, 0. ],
 [0.76, 0.04, 0. , 0.2 ],
 [0. , 0.48, 0.52, 0. ],
 [0.4 , 0.24, 0.36, 0. ],
 [0. , 0.32, 0.68, 0. ],
 [0. , 0.24, 0.76, 0. ],
 [0.88, 0. , 0. , 0.12],
 [0.68, 0. , 0. , 0.32]])
```

```
In [134]: y_test
```

```
Out[134]: [3, 3, 3, 3, 3, 2, 2, 2, 0, 0, 1, 1, 2, 2, 3, 3]
```

- `clf_probs` returns the probability of the test being classified as each label
- Super cool!! If you compare the labels with the highest probability, some are actually right! e.g. test 9 is 92% sure that it is a node test, which is correct!
- Can use these probabilities, looking at the plot of the raw data, to try to figure out why the classifier isn't working for some of the tests
  - e.g. really things most of the “nothing” tests are node tests -- should add more nothing data or figure out feature that would differentiate the node test from the nothing data, because the nothing data definitely sometimes has some random bumps that would look like a node test
- Figure out my own way of visually understanding the probabilities, or make my own metric for determining how accurate the classifier actually is
- Make improvements -- figure out how this stuff really works
- Maybe make an initial plot of how sure the classifier is about each label, on average for positive tests

### **My Update (Tensegrity HRI Meeting 6.12.20 cancelled)**

- Was able to use the receiver operating characteristic method on my machine learning model by slightly modifying the format of the data, but I used all the same data processing techniques and features from my existing model which proved to be fairly successful
- The curve I got for ROC looked promising but I'm not entirely sure what's going on
- I get that the top left corner is the “ideal point” because it's a false positive rate of 0 and a true positive rate of 1 but I don't know if I understand where the curve comes from
- Was able to binarize the output, so each of the 4 labels turned into an array of three 0's and one 1, depending on what class it was
- Made an ROC curve using binary prediction (which I also don't entirely understand)
- Not done trying to understand ROC, but I also tried using a probability calibration method where instead of the classifier outputting a guess for the label it outputs the probabilities of the sample being each class

- Learned that if the classifier is trained on all the data points designated for training then it is overly confident in its predictions which leads to a large log-loss
- So if you train the classifier on less data and use the “sigmoid” method on the remaining training data points to calibrate the classifier, this reduces the confidence of the predictions which results in a lower log-loss
- Increasing the number of base estimators also decreases log-loss
- The probabilities are really interesting to look at, e.g. the non-validated classifier is really sure about node tests and squeeze tests (60-90% sure) but isn't sure about drop tests (30-40%) and thinks nothing data is node tests
- I want to think about these probabilities more and try to come up with more features that would differentiate between these tests based on these observations

## Plan for this week

- Look at CS189!! Will definitely clarify a lot of stuff
  - [http://people.eecs.berkeley.edu/~jrs/189/?fbclid=IwAR2BH8K-mMfbEbBSepFumKQCgqV2y5B\\_b1L5mHlyBYXYx\\_RiDCpgaYZWJXE](http://people.eecs.berkeley.edu/~jrs/189/?fbclid=IwAR2BH8K-mMfbEbBSepFumKQCgqV2y5B_b1L5mHlyBYXYx_RiDCpgaYZWJXE)
- Understand ROC more
- Refresh on log-loss and sigmoid curves (3Blue1Brown intro to ML series)
  - <https://www.youtube.com/watch?v=aircAruvnKk&t=3s>
- Brainstorm more features based on probabilities
- Come up with my own metric on how to understand these probabilities?
- Email Andrew progress

Berkeley Emergent Space Tensegrities Laboratory

# TENSEGRITY HRI MEETING

---

6.5.20 / 1:00-2:00PM / ZOOM <https://berkeley.zoom.us/j/2417093709>

## My Update

- Brainstorm of ways to conduct experiments while being careful about COVID-19:
  - 3 people:
    - We are already capable of remaining 6 ft apart while conducting the tests since one person is operating the camera, one is manipulating the robot, and one is collecting data on the computer
    - Everyone must sanitize hands and be clean when they come into the lab, remain at their respective stations (robot, camera, computer)
    - If the wires get disconnected from the board or the robot needs to be reset, the person operating the robot must fix/reset it
    - Everyone wears masks
    - If someone's help is really needed at a different station, maintain distance of 6 ft apart and use sanitary wipe to clean
    - Or everyone washes hands, puts on gloves, and avoids touching their face the whole time in lab. Then we wouldn't need sanitary wipes as often
  - 2 people:
    - Same as above, but one person manipulates the robot and writes on white board, and other person operates both camera and computer
- Read article comparing ROC (receiver operating curves) and precision recall curves  
<https://machinelearningmastery.com/roc-curves-and-precision-recall-curves-for-classification-in-python/>

- Learned that ROC curves should be used when there are roughly equal numbers of observations for each class -- would be OK to use this when we have a larger set of data in which we have purposefully run a set number of tests (although not sure exactly what they mean by “roughly equal”)
- Precision-recall curves should be used when there is a moderate to large class imbalance
- ROC curves “present an optimistic picture of the model on datasets with a class imbalance” and “might be deceptive and lead to incorrect interpretation of the model skill”
- PR plots can provide an accurate prediction of future classification performance because they evaluate fraction of true positives among positive predictions and doesn’t account for true negatives
- Either way, these curves are dealing with probabilities and thresholds to determine how sure the model is about making a classification or if it’s not sure, so we could start by converting our model to one that predicts the likelihood of a certain class and this would be more reliable data to publish than just the rate the model is correct

## Plan for this week

- Going to start trying to implement ROC methods between two classes of data and see if I can get okay results for the area under the curve metric
  - Sci kit learn links:
    - [https://scikit-learn.org/stable/auto\\_examples/model\\_selection/plot\\_roc.html](https://scikit-learn.org/stable/auto_examples/model_selection/plot_roc.html)
    - [https://scikit-learn.org/stable/modules/generated/sklearn.metrics.roc\\_curve.html](https://scikit-learn.org/stable/modules/generated/sklearn.metrics.roc_curve.html)
    - [https://scikit-learn.org/stable/auto\\_examples/model\\_selection/plot\\_roc\\_crossval.html](https://scikit-learn.org/stable/auto_examples/model_selection/plot_roc_crossval.html)
    - [https://scikit-learn.org/0.15/auto\\_examples/plot\\_roc.html](https://scikit-learn.org/0.15/auto_examples/plot_roc.html)
- Also going to start looking into models that return probabilities of something being in a certain class instead of the model just labelling it as the most likely one
- General sci kit learn links:

- <https://scikit-learn.org/stable/>
- [https://scikit-learn.org/stable/getting\\_started.html](https://scikit-learn.org/stable/getting_started.html)
- [https://scikit-learn.org/stable/auto\\_examples/classification/plot\\_digits\\_classification.html#sphx-glr-auto-examples-classification-pot-digits-classification-py](https://scikit-learn.org/stable/auto_examples/classification/plot_digits_classification.html#sphx-glr-auto-examples-classification-pot-digits-classification-py)
- Other machine learning-related stuff
  - <https://machinelearningmastery.com/how-to-develop-and-evaluate-naive-classifier-strategies-using-probability/>
  - <https://machinelearningmastery.com/what-is-bayesian-optimization/>
- Don't forget to keep working on human factors training:
  - <https://cphs.berkeley.edu/quickguideCITItraining.pdf>

## Meeting Notes

- One label vs. rest of labels, not one vs. one for all 24 (?) combinations
- More trees improve classifier performance

Berkeley Emergent Space Tensegrities

# TENSEGRITY HRI MEETING

---

5.29.20 / 1:00-2:00PM / ZOOM <https://berkeley.zoom.us/j/2417093709>

## Meeting Notes

- How to move forward with experimenting
  - Work remotely, use first prototype to get more data because we really need more data to write the paper
- Think about ways to conduct experiments effectively given the circumstances
- Look through Andrew's critiques
- Understanding how we're analyzing the results of our classification model
- So far we're just evaluating the ability of model to output correct label
- Can determine probability of model to label correctly
- Another metric: area under the receiving operating curve -- compares true positive rate vs. false positive rate. Curve based on number of samples (this is for binary classification)
- Packages in sci kit learn for AUC. see what the results are for our current model
- Biasing: the way we are training the model is such that it is predisposed to look for certain scenarios rather than looking or anything
- Look at different model types on sci kit learn
- In general learn sci kit learn
- Conditional probabilities to inform our understanding of how our model will perform based on different conditions
- CS 189 website and Data 100 textbook

# PAPER SECTION DRAFT

---

## 5.9.20

### V. EXPERIMENTS

Our research goal is to show that a tensegrity robotic system can be used to robustly differentiate between various interactions it can experience with humans. In order to prove that the classification model described above may be used reliably to distinguish between these interactions, it must be verified on an example dataset that describes a variety and multitude of human interaction data. We thus designed our experiments with our tensegrity robotic system around a defined set of human interactions to use with the system, modeled after the types of interactions we would expect to see in applications of this research. Our goal is to assemble a large set of data from these sample interactions to show that the classification modeling can be used to distinguish between them.

As described in detail in previous sections, we assembled a 6-bar tensegrity (compliant) robot prototype equipped with an array of 12 force sensing resistors placed on the ends of the compression rods, in addition to 3 accelerometers located in the central payload of the system. We hypothesized that distinct interactions would yield distinct datasets of force sensor and accelerometer data, and thus be distinguishable by a classification model. The tensegrity robot was tested on the following human interactions:

- Drop
  - A researcher holds the robot from a consistent height and releases it
  - Data of the release and impact, until the tensegrity comes back to rest, is recorded
- Squeeze
  - The robot is placed on a table. A researcher uses a large piece of material (e.g. plexiglass) to place on the top of the robot and compress the tensegrity structure three times
  - Data of the three successive squeezes is recorded together

Qualitatively, it is expected that the data between these two types of interactions, when plotted, will be visibly very distinct. In particular, when dropped, the tensegrity experiences very high forces within a very short interval due to its impact with the floor, as well as erratic vibrations post-impact. On the other hand, the tensegrity will experience lower, smoother forces over longer periods of time when the entire structure is intentionally compressed by a human. Based on these expectations, we expect to be able to both visually distinguish between the interactions as well as apply our classification model to distinguish between them based on distinct patterns in the data.

To make the experimentation robust, these interactions were tested from a variety of starting orientations. To expand upon the description of the tensegrity robotic system given in Section III.C Force-Sensor Array, we will describe the specific locations of the numbered force sensors for clarity in the description of the experimental setup. The tensegrity has 12 distinct nodes, each of which houses a single force sensing resistor, and each node and respective sensor is labeled with an integer from 1 to 12. We expect, depending on the orientation of the system

when the interaction takes place, and despite the fact that the forces propagate across the system to all compression elements, tension elements, and nodes, that different sensors will see different forces. For example, when squeezed, we expect that the force sensors in the nodes that are in contact with the table and the plexiglass will return higher force readings than the nodes on the sides.

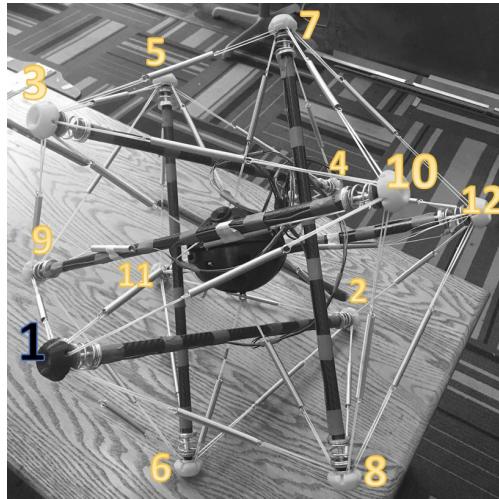


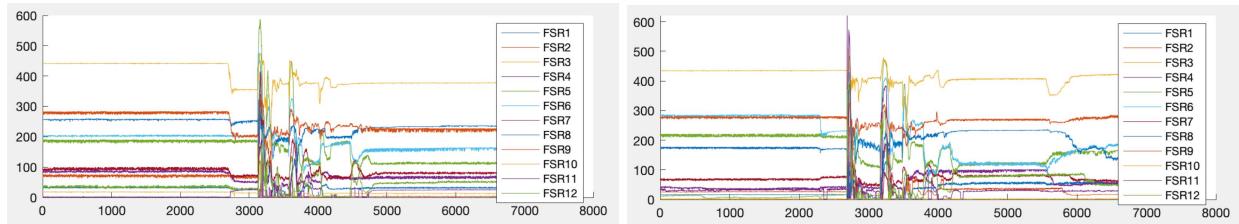
Fig. [] 6-bar tensegrity with labeled nodes

Hence, a dataset from e.g. two different drops, each with a different starting orientation, may look distinct. Therefore from a data analysis standpoint, in order to ensure that our classification model is as robust as possible, it is necessary to perform experiments with many various starting orientations.

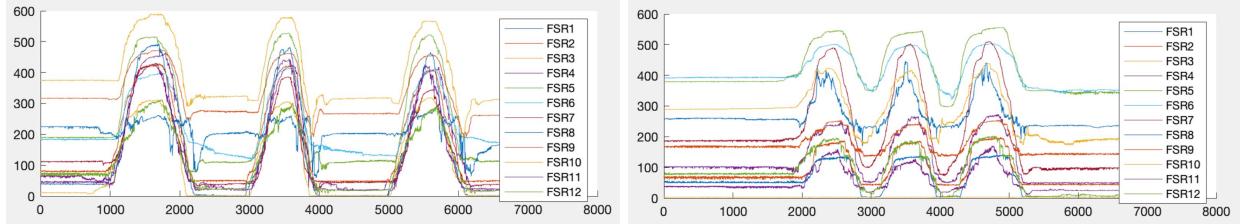
\*\* Describe experimental setup in more detail, including using wireless connection to microcontroller inside payload to collect FSR data \*\*

We performed drop tests from a variety of orientations, classifying each test as either “closed-face,” “open-face,” or “double rod,” depending on the node or string orientation we expected the tensegrity to land on. The squeeze tests were each classified as either “closed-face,” “open-face,” “double rod,” or “single rod.” Examples of two drop tests and two squeeze tests performed from different orientations are shown below:

Drop:



Squeeze:

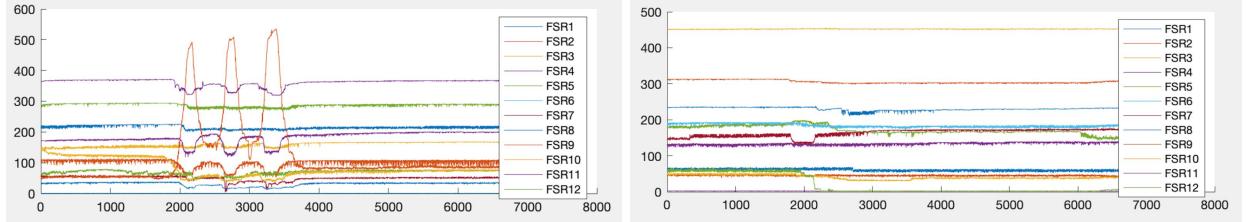


Accelerometer data is plotted separately and not shown here. Visually, the force plots of the drop and squeeze tests appear very distinct from one another. We are also able to infer that the two drop tests and two squeeze test were executed from different orientations based upon which FSR has the highest force readings. Note also that each FSR also has its own degree of noise, which may affect the accuracy of our classification model.

A classification model ideally has thousands of data points with which to train. Due to limited time we were only able to collect a few dozen squeeze and drop tests. Because we aim to prove that our classification model works as a proof of concept, and because our plans for the future of this research include addition more interaction classes to our data collection (including but not limited to roll tests and throw tests), we decided to include two more classes of data we passively collected to train our proof-of-concept classification model on:

- Node tests
  - A researcher squeezes a single node three times
  - Data was originally collected while doing this simply in order to make sure the wireless connection to the robot was working properly
- Null tests
  - Data is collected when no interaction is happening

Examples of a node test (left) and a null test (right) are shown below:



Discussions on the design of our classification model and results of running the model on our dataset follow in the next two sections.

## VI. CLASSIFICATION MODEL

Designing and implementing a working classification model on our dataset completes our proof-of-concept that physical interaction can serve as a reliable platform for human-robot interaction. In proper human-robot interaction, the ideal tensegrity robotic system responds in certain ways according to specific manipulations of the compliant system that the human performs intentionally. Therefore the robotic system is required to properly classify the interaction based on live force sensor data in order to respond accordingly. We have not yet developed the technology in order to implement real-time response, but our preliminary classification model and results will show that it is possible to leverage physical interaction in human-robot interaction.

We began the design of our classification model based on our experimental data and the classes of interaction we specifically chose to analyze. The purpose of this model is simple: to properly classify each test data point as a certain type of test. A more specific model might do this as well as identify the starting orientation of the robotic system based on its highest force readings. For now, we will proceed with just distinguishing between categories of tests.

- Comments from Andrew in original google doc 5.12.20

- <https://docs.google.com/document/d/1XcnhHoGMeiBvQVPj0DSxanErEMPMxFOP8WKK07F0Lyg/edit>

Berkeley Emergent Space Tensegrities

# TENSEGRITY HRI MEETING

---

5.6.20 / 1:00-2:00PM / ZOOM <https://berkeley.zoom.us/j/2417093709>

## Meeting Notes

- Discussion on datasets
- Normalize STD = 1
- Tuning sensitivity -- making sure random noise isn't classified as a test
- Want it to be as insensitive to outliers as possible
- Different size interval tests
- Discrete constant time windows over range of datatypes
- Featurization works well
- Use slopes of accelerometer data
- PCA: principle component analysis -- modes of a signal are its features
- Maximum jerk, total impulse
- Try minimizing features
- Numerical approach to see what features are the best
- Try slow, real-time classification with moving window?
- Spend more time on different classifiers, but can probably press forward with the best one -- random forest
- Random forest -- averages subset of decision trees for robustness
- Message: given FSR data and human manipulation of robot, can infer useful information for HRI
- How will we analyze the results of the classifier?
  - Confusion matrices -- true positive, false positive rates
  - Way to quantify ability of binary classifier to robustly classify

- Find out a way to do this with more tests
- Data 100 textbook
- Decide what matrices to use to determine accuracy of classifiers
- What quantitative results from classifiers to use with results

Berkeley Emergent Space Tensegrities

# TENSEGRITY HRI MEETING

---

5.1.20 / 1:00-2:00PM / ZOOM <https://berkeley.zoom.us/j/2417093709>

## My Update

- Found a mistake that was lowering my model's accuracy (kept in the time data)
- Played around a lot with different features and tried doing various models with all data vs. condensed data, featured data vs. no features, with accelerometer data vs. without accelerometer data -- it's all on my Jupyter notebook which is available on my github branch

```
In [5]: # This feature returns the percent of data points that are below 75% of the max

def three_quarters(column,percent):
    max = np.max(column)
    three_fourths = percent*max
    e = []
    j=0
    for i in range(len(column)):
        if column[i] <= .75*np.max(column):
            j+=1
    return j/len(column)
```

```
In [6]: # This feature takes the max of each column, takes the data point [change_factor] ms after that, and then normalizes
# the difference

def change_in_max(column, change_factor):
    max_index = np.argmax(column)
    minus_change_index = max_index - change_factor
    if minus_change_index > 0:
        minus_change_value = column[minus_change_index]
    else:
        minus_change_index = max_index + change_factor
        minus_change_value = column[minus_change_index]
    diff = np.max(column) - minus_change_value
    normalized = diff/np.std(column)
    return normalized
```

```
In [7]: # This function takes all 72 data points of size (7001,16) and puts them into a feature-engineered array.
# Desired shape: 72 data points of size (6,16) --> (72,6,16)

def features(array, three_quarters_percent, change_in_max_change_factor):
    len_dataset, _, number_of_columns = np.shape(array)
    number_of_features = 6
    condensed_data = np.zeros((len_dataset,number_of_features,number_of_columns))
    for j in range(72):
        data_point = array[j]
        number_of_columns = len(data_point[:, :])
        D = np.zeros((number_of_features,number_of_columns),dtype=np.float64)
        for i in range(number_of_columns):
            column = data_point[:, i]
            D[0,i] = np.mean(column)
            D[1,i] = np.var(column)
            D[2,i] = np.max(column)-np.min(column)
            D[3,i] = three_quarters(column,three_quarters_percent)
            D[4,i] = change_in_max(column,change_in_max_change_factor)
            D[5,i] = np.sum(column)/np.max(column) #or np.std(column). MAX WORKS THE BEST! went from 80% or so to 100%
        condensed_data[j]=D
        if np.mod(j,6) == 0:
            print(j, end=" ")
    return condensed_data
```

```
In [8]: # Condense data into averages over a specified number of data points

def condense(array, condense_factor):
    len_dataset, _, number_of_columns = np.shape(array)
    length = len(array[0])
    condensed_length = int((length/condense_factor))
    condensed_data = np.zeros((len_dataset,condensed_length,number_of_columns))
    for j in range(len_dataset):
        data_point = array[j]
        for k in range(number_of_columns):
            column = data_point[:, k]
            for i in range(condensed_length):
                condensed_data[j][i,k] = np.mean(column[i*condense_factor:(i*condense_factor+condense_factor)])
    return condensed_data
```

```
In [9]: # Takes the max of all features from the 16 datapoints. Output is 72 sets of 1 by 6 data, where each of the 6 points
# is the max of each feature from the set.

def everything_max(array):
    len_dataset, num_features, num_datapoints = np.shape(array)
    maxed_data = np.zeros((len_dataset, 1, num_features))
    for i in range(len_dataset):
        datapoint = array[i]
        D = np.zeros((1, num_features))
        for j in range(num_features):
            D[0,j] = max(datapoint[:,j])
        maxed_data[i] = D
    return maxed_data
```

```
In [10]: # Make model

def make_model(dataset,num_train):
    nsamples, nx, ny = np.shape(dataset)
    reformatted_data = np.reshape(dataset,(nsamples,nx*ny))

    train = reformatted_data[0:num_train]
    test = reformatted_data[num_train:nsamples]
    train_labels = labels[0:num_train]
    test_labels = labels[num_train:nsamples]

    # Return percent of correct labels
    def percent_correct(test_dataset, test_labels):
        total = 0
        a = model.predict(test_dataset)
        for i in range(len(a)):
            if a[i] == test_labels[i]:
                total += 1
        correct = total/len(a)*100
        return correct

    model = GaussianNB().fit(train, train_labels)
    print("Predicted labels = " + str(model.predict(test)))
    print("Correct labels = " + str(np.asarray(test_labels)))
    return percent_correct(test, test_labels)
```

- I got 100! By playing around with one of my features -- I think it was Yash's idea to take the sum of all data, but I was testing out different ways to normalize (max was the best)
  - `np.sum(column)/np.std(column)`
  - `np.sum(column)/np.max(column)`
- Trying different numbers of training vs. testing data points, different features, using maximization or not, using normalization or not:

```
In [12]: # Normalize and feature-ify the data
normalized_data = data/np.max(data)
normalized_featured_data = features(normalized_data,.75,50)
np.shape(normalized_featured_data)

# Make non-normalized, featured dataset (dataset size (72,6,15))
featured_data = features(data,.75,50)

# Make dataset that only has max of each feature
maxed_data = everything_max(featured_data)

0 6 12 18 24 30 36 42 48 54 60 66 0 6 12 18 24 30 36 42 48 54 60 66

In [13]: # Condense and featurify the normalized data
condensed_data = condense(normalized_data,10)
condensed_data = features(condensed_data,.75,5)

0 6 12 18 24 30 36 42 48 54 60 66

In [19]: np.shape(maxed_data)

Out[19]: (72, 1, 6)

In [40]: # Train model on normalized and feature-ified data
make_model(normalized_featured_data,60)

Predicted labels = [3 2 2 2 0 0 1 1 2 2 0 3]
Correct labels = [3 2 2 2 0 0 1 1 2 2 3 3]
Out[40]: 91.66666666666666

In [41]: # Train model on non-normalized, featured data (dataset size (72,6,15))
make_model(featured_data,60)

Predicted labels = [3 2 2 2 1 0 1 1 2 2 1 3]
Correct labels = [3 2 2 2 0 0 1 1 2 2 3 3]
Out[41]: 83.3333333333334

In [43]: # Train model on max-feature-engineered data
make_model(maxed_data,60)

Predicted labels = [0 2 2 2 0 0 1 1 2 2 1 3]
Correct labels = [3 2 2 2 0 0 1 1 2 2 3 3]
```

## Meeting Notes

- Work on with paper: force sensor array, my work relevant to classification modeling, have a good understanding of experiments
- Andrew writes most, we contribute some and comment
- Novel HRI approach

- Importance: 1st attempt of a platform for real mobile robotic platform. In future, propose that robotics will be largely used by humans. How leverage contact to make interactions better. Robot that is easy to manipulate, leverages contact to improve communication. Here are some preliminary results on how to do that
  - Intellectual merit
- Pose classification modeling on context of force sensing, argue why it works (not practically), thinking about how we will do classification modeling
- Details: what classification model, what featurization
- How design to best control for sources of error -- how error can make our classification models more robust?
- Analyzing results of how well classification model performs. Exclusivity or reliability, not just how right it is with given data
- Tensegrity system can use force sensing data to distinguish between interactions which has implications for the future of human-robot interaction. Info from sensors is useful, at the very least for proof of concept/demonstration of classification abilities from data
- If we had more reactions (everything you could think of) → much better argument

### **Plan for next week**

- Pick one specific area of the paper we want to take a stab at and try to flesh out bullet points more. Have done by Wednesday
- How best to express results in terms of classifier
- Comb through results -- figure out how to express results based on different metrics for interpreting results
- Meeting for Tuesday -- consolidate and merge results that everyone has put together

Berkeley Emergent Space Tensegrities

# TENSEGRITY HRI MEETING

---

4.24.20 / 1:00-2:00PM / ZOOM <https://berkeley.zoom.us/j/2417093709>

## Meeting Notes

- Paper: everything up to and including data processing (not 2nd iteration prototype -- maybe in future work). 1st paper ever published on tensegrity HRI
- Physical HRI with mobile robots, compliant robots
- Compliant mobile robotic system capable of doing physical HRI
- Akhil's software update: created function that creates dataset of different sizes, splits all data up into chunks of the same size
- Discussions of how to best combine approaches -- results for paper
- Ensemble learning method

## Plan for next week

- Moving forward, can put max or min of each feature into smaller dataset, try normalizing data against max of everything
- Paper contribution -- experimental design and results
- Idea of what we'll be working on for paper

Berkeley Emergent Space Tensegrities

# TENSEGRITY HRI MEETING

---

4.17.20 / 1:00-2:00PM / ZOOM <https://berkeley.zoom.us/j/2417093709>

## My Update

- Put a bunch of features into my script including some basic ones like mean, max, and variance and a couple other ones that I thought of, been interesting to run the script taking out certain features and seeing which ones affect the results
- My results have improved slightly since last time but what's interesting is that my script refuses to guess that anything is a drop which is weird because I was hoping that my features would target the differences between drops and the other tests, so maybe I should send Andrew what I have and he can give me feedback
- Also tried compressing my data into averages over ten or so data points but thus far it hasn't made a difference but I'll keep working on it

## Plan for next week

- Total force -- sum every data point, maybe normalize with max or standard deviation
- Max change in data across a few data points
- Take max of each feature → only ID dataset for training

Berkeley Emergent Space Tensegrities

# TENSEGRITY HRI MEETING

---

4.10.20 / 1:00-2:00PM / ZOOM <https://berkeley.zoom.us/j/2417093709>

## My Update

- Decided to start over my ML process because the one I started wasn't working
- Spent some time re-truncating and labelling the data using Akhil's algorithm, captured entire tests in a consistent time interval of 7 seconds so instead of having snippets of data I was wanting to see how well it can differentiate entire tests, because I think Akhil and Albert are only doing short time intervals
- Setting up the correct format of data that I wanted took a while so I didn't get around to feature engineering, but I have a script using sci kit learn that functions, it isn't very good it's about 50% accurate but it's using only the raw data so all things considered with 0 feature engineering it's 25% more accurate than random guessing (have 4 categories of tests)
- This is a learning process for me so I'm glad that I finally have a model that actually gives results even though they're not great so far

## Plan for next week

- Have ideas for feature engineering that I don't think have been done yet so I'm going to work on that next week and I'm still interested in trying to use Tensorflow which is more neural networks
- By next week my goal is to have a script that incorporates full feature engineering on the entire test
- Once I have a viable script with my 7 second tests I'm going to try to split up the data into much smaller time intervals

## Meeting Notes

- Bayesian map between the features -- correlations between features → if certain features have the same information
- Data compression -- averaging microseconds over 10 data points or something

- Another idea: figure out max of all data (all force sensors), figure out how many force sensors get above 75% of that value
- PCA – helpful tool
- TSNE – dimensional reduction tool
- SVM
- random forest
- cross-validation
- bootstrapping

Berkeley Emergent Space Tensegrities

# TENSEGRITY HRI MEETING

---

4.3.20 / 1:00-2:00PM / ZOOM <https://berkeley.zoom.us/j/2417093709>

## My Update

- Learned Anaconda, Jupyter, went over some python basics, a bunch of numpy tutorials
- Brainstormed features, thinking about trying a conventional script that analyzes features like max force, oscillation frequency, maybe variance and stuff like that and have about 60 lines of code for that
- Also had the idea to do image processing on the .png files themselves because if we should be able to tell that a drop test is different from a squeeze test then a computer should be able to also
  - Andrew: don't do that, literally just use the raw data :(
- Going really slowly because each .png has 2.9 million pixels that each have 3 RGB values (I could also do black and white but that's still a lot)
- I'm interesting in trying to follow this through but if it seems like it's not worth it then I'll try something else
- I'm using Tensorflow so far which seems ok but I'm also probably going to try to learn sci kit learn

## Plan for next week

- Same thing, just progress on stuff again
- Different data processing, featurization, model development directions

Berkeley Emergent Space Tensegrities

# DOCUMENTATION OF PROGRESS

---

## 4.1.20

### Progress

- So much coding! Mostly data organization and feature engineering for two different ML scripts
- Data processing is taking a while

### Next

- Things from previous days, finish putting data into appropriate format for ML model, hopefully complete ML models

Berkeley Emergent Space Tensegrities

# DOCUMENTATION OF PROGRESS

---

**3.29.20**

## Progress

- A lot of Tensorflow Udacity course, learning the framework for writing a machine learning script
- Idea for our ML: image processing with the image of our data?
- Did some of the human factors training

## Next

- Same as from 3.28

Berkeley Emergent Space Tensegrities

# DOCUMENTATION OF PROGRESS

---

**3.28.20**

## Progress

- Watched Python Anaconda tutorial, went over how to use Anaconda, Jupyter notebooks, python basics, a little bit of data visualization
  - Jupyter Notebook Tutorial:  
<https://www.youtube.com/watch?v=fiQTb7-rCPo>
- Watched Python for Data Analysis -- Python Pandas Tutorial
  - <https://www.youtube.com/watch?v=B42n3Pc-N2A>
  - Learned basic commands in pandas for data manipulation
  - Learned how to plot .csv files in python

## Next

- Saved videos (more Edureka python basics stuff)
  - Full course for Python beginners:  
<https://www.youtube.com/watch?v=rfscVS0vtbw>
  - Python OOP tutorial: <https://www.youtube.com/watch?v=ZDa-Z5JzLYM>
  - Intro to Python: <https://www.youtube.com/watch?v=uYjRzbP5aZs>
  - Python full course (12 hours):  
<https://www.youtube.com/watch?v=WGJJIrtnfpk>
  - Python numpy tutorial: [https://www.youtube.com/watch?v=8JfDAm9y\\_7s](https://www.youtube.com/watch?v=8JfDAm9y_7s)
  - Another python numpy tutorial:  
<https://www.youtube.com/watch?v=GB9ByFAIAH4>
- Convert intro to ML files to Jupyter, delete Colab notebooks
- Continue Tensorflow course
- Start looking at sci kit learn

- Feature engineering tutorials -- could this include basic stuff like mean, max, min, data of each force sensor
- Data 100 textbook
- Features: max, mean, oscillation frequency, number of occurrences of higher data points, e.g. like take  $\frac{3}{4}$  max, count what percentage of data is greater than that, maybe number of data points, etc. (more!!)  
variance/standard deviation

Berkeley Emergent Space Tensegrities

# TENSEGRITY HRI MEETING

---

3.27.20 / 1:00-2:00PM / ZOOM <https://berkeley.zoom.us/j/2417093709>

## My Update

- Interested in continuing to work on node CADs and sending them to whoever has 3D printing capabilities
- Also want to start thinking about redesigning the payload
- Learn Jupyter notebooks
- Data processing -- sci kit learn, pandas
- paper!

## Meeting Notes

- Resources from Andrew to expand skill set in data processing
- IEEE conference paper. Which conference?
- Data science resources:
  - Andrew's document for data science and machine learning resources: <https://docs.google.com/document/d/19A4s0uFYc0r0Egi6gq1y-QX2AnPVL2OgtmjeMiP9Bjo/edit?ts=5e7e5a3e>
  - <https://www.datacamp.com/>
  - Data 100 textbook: <https://www.textbook.ds100.org/intro>
- Anyone can work on data processing side of project, try out different models and evaluate accuracy
- Learn python, pandas (python package to manipulate tables easily)
- Python script on github
- Data 100 textbook
- Sci kit learn
- Logistic regression -- probabilities from 0 to 1

- Get scripts Andrew gives us and try to understand

### **Plan for this week**

- Spend time on task we're prioritizing most -- report on progress. e.g. train first model, make first pipeline (?)
- Explore resources given

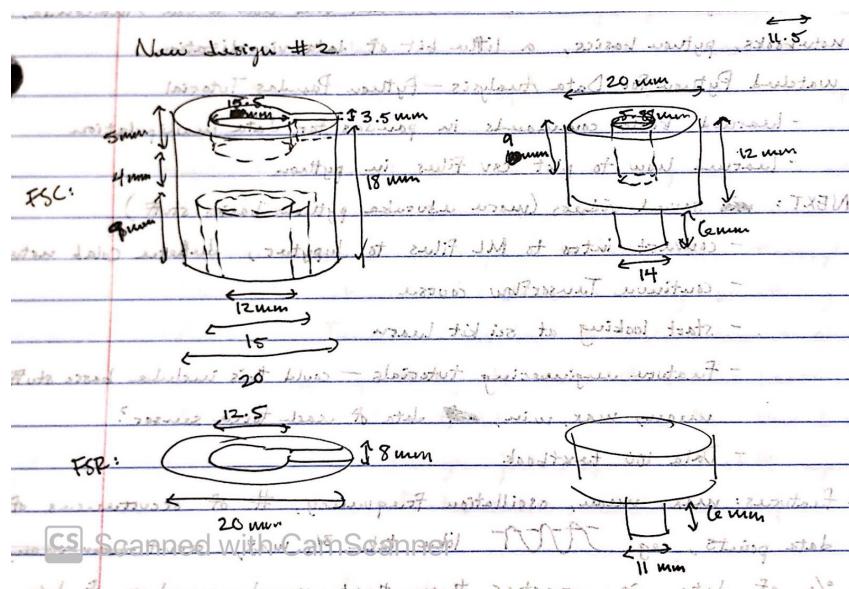
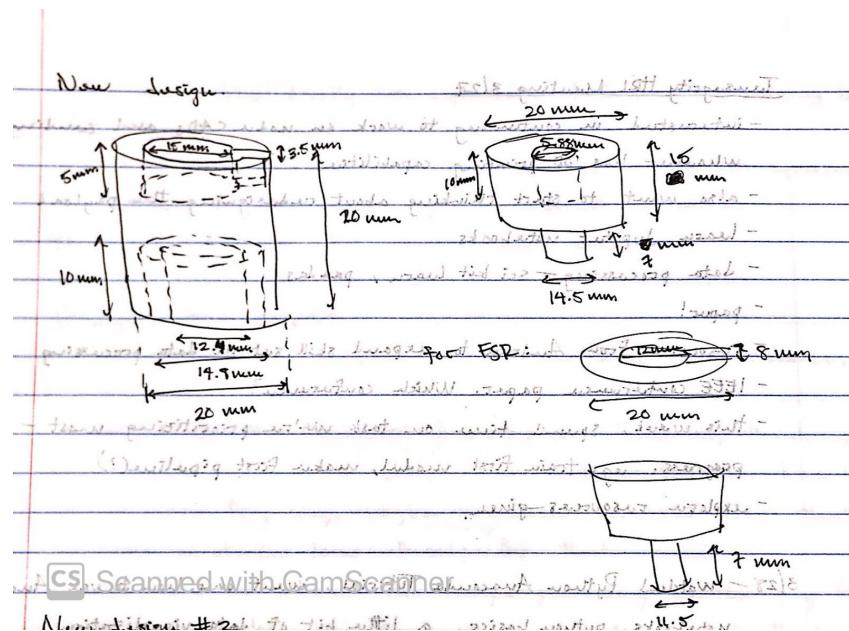
# Berkeley Emergent Space Tensegrities

# NODE DESIGN SKETCHES & CADS

3.27.20

## Designs

- File path for STL files: /Users/salatiemann/Documents/UC Berkeley/Year 3 Sem 2 (Spring 2020)/node redesign2



Berkeley Emergent Space Tensegrities

# TENSEGRITY HRI MEETING

---

3.13.20 / 1:00-2:00PM / 230 Hesse

## Meeting Notes (Transcript from Bi-Weekly Tensegrity HRI Meeting Notes)

- Andrew's deliverables:
  - Final sensor selection (it seems like FSR is the way to go)
  - Testing of FSR circuit (look into FSR integration doc that Andrew sent)
    - <https://www.sparkfun.com/datasheets/Sensors/Pressure/fsrguide.pdf>
  - Testing of several (2 or 3) different hardware configurations for sensor integration

Berkeley Emergent Space Tensegrities

# DESIGN DISCUSSION

---

3.9.20 / 1:00-2:00PM / 230 Hesse

**Meeting Notes (Transcript from Bi-Weekly Tensegrity HRI Meeting Notes)**

- Pick sensor and node design we want and actually build a whole robot
- Friday -- update Andrew about what we found/how we want to proceed
- Payload redesign? Get CAD designs from original
  - Bare minimum for testing

Berkeley Emergent Space Tensegrities

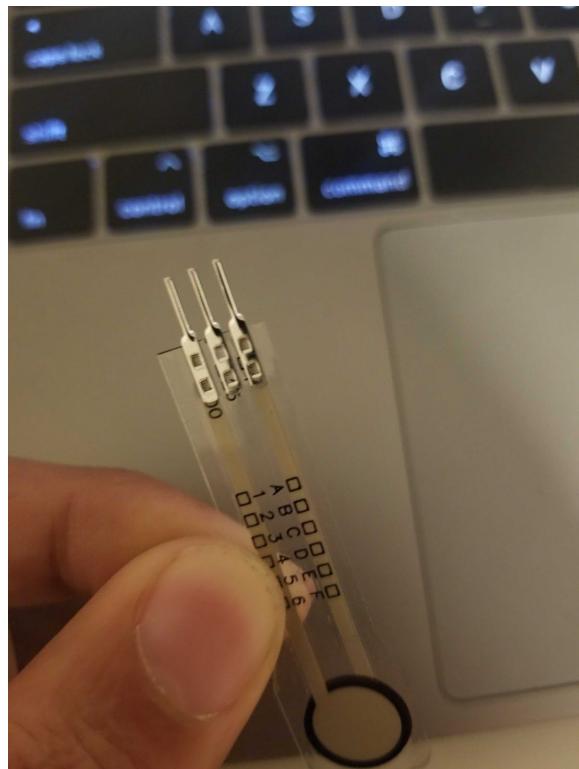
# TESTING MEETING

---

3.8.20 / 1:00-2:00PM / 230 Hesse

### Meeting Notes (Transcript from Bi-Weekly Tensegrity HRI Meeting Notes)

- Force sensing resistor testing meeting
- Tried testing the FSR with the op amp circuit but it didn't work as we didn't have the right op amp
- Got values from the FSR which made sense using a voltage divider
- Need to verify the op amp circuit and calibrate it ourselves



**Node Redesign Mockup**

New design #26 To - 14 mil x 22mm width 14.5

longitudinal ventral - cilioreticulae about 20 mm long; buccal area

A hand-drawn technical diagram of a cylindrical component. The cylinder has a top flange with a diameter of 18.5 mm and a thickness of 3.5 mm. The main body of the cylinder has a diameter of 12 mm and a height of 14 mm. A central vertical slot with a width of 12 mm is present. The bottom of the cylinder features a stepped base with a central circular cutout of diameter 12 mm and a side wall thickness of 4 mm. The overall height of the cylinder is 18 mm. The drawing includes several handwritten annotations: 'Sides' with a double-headed arrow between the top and bottom flanges; '4 mm' indicating the thickness of the base wall; '18 mm' for the total height; '12 mm' for the bottom base diameter; '14' for the bottom height; '12 mm' for the central slot width; '3.5 mm' for the top flange thickness; and '18.5 mm' for the top flange diameter.

~~Auto wird ab sofort 20 km/h - absolut verboten~~

~~For more details, refer to step 12-5.~~

FSR: Diagram of a transverse section of a canine tooth root showing a large apical lesion. The lesion is labeled as 8 mm in diameter.

$20 \text{ mV}$

Scanned with CamScanner

and the following additional sections of the Act:

Berkeley Emergent Space Tensegrities

# TESTING MEETING

---

3.4.20 / 1:00-2:00PM / 230 Hesse

**Meeting Notes (Transcript from Bi-Weekly Tensegrity HRI Meeting Notes)**

- Capacitive force testing meeting
- Need plot for 0-100 pounds from SingleTact document to fit uncalibrated data
- When placing weights, first kilo had a larger effect on the readings than subsequent kilos
- Prove relationship exists if we can get this 0-100 lb plot
- Issues that we didn't foresee: sensor neck is too short before it needs to connect to the board so we would need to either extend the ribbon cable or make our own boards
- Is there enough space to fit the electronics from all 12 sensors in the payload?
- Plan for Friday at 1: run validation tests on FSRs because they should be delivered soon
- From this point, we think the FSRs will ultimately be simpler to implement

# Berkeley Emergent Space Tensegrities

# DESIGN MEETING

---

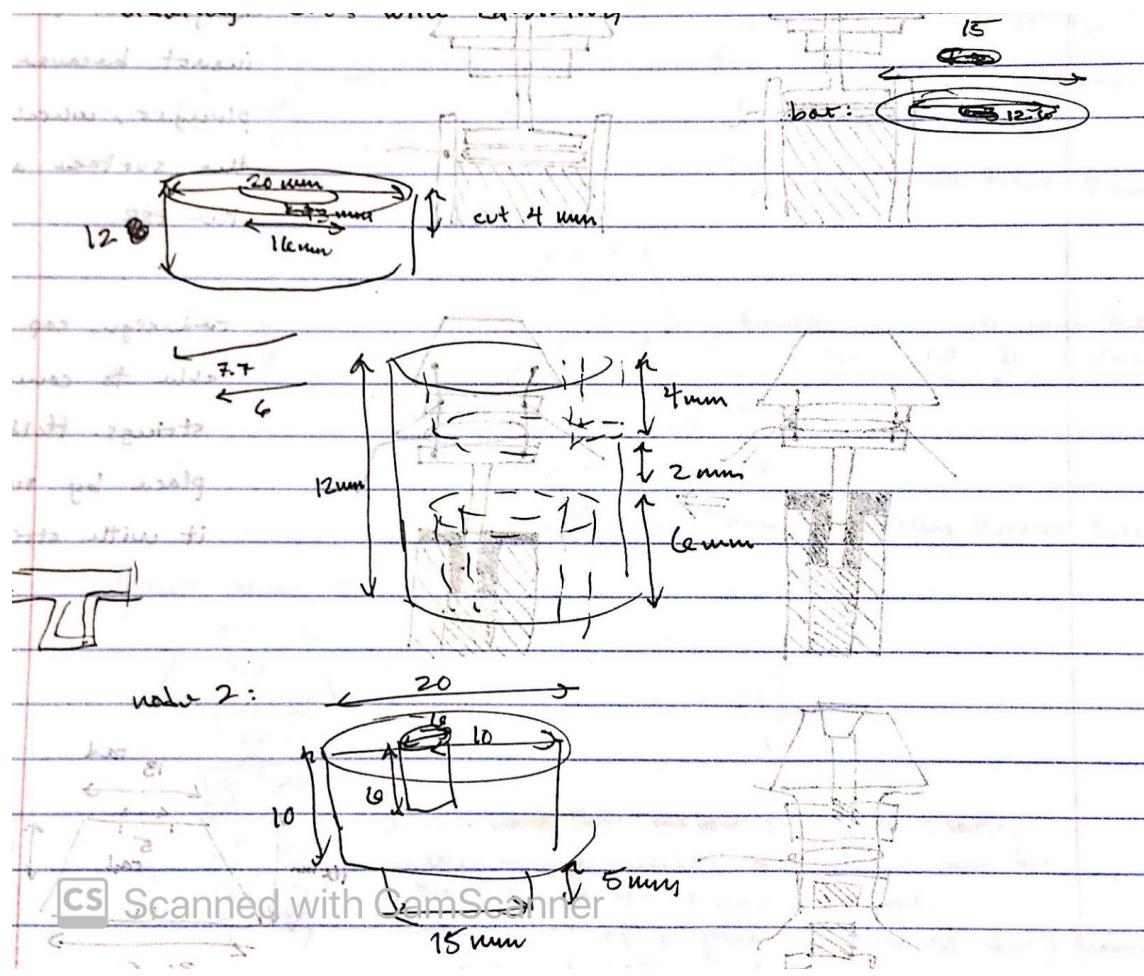
2.20.20 / 1:00-2:00PM / 230 Hesse

## Meeting Notes (Transcript from Bi-Weekly Tensegrity HRI Meeting Notes)

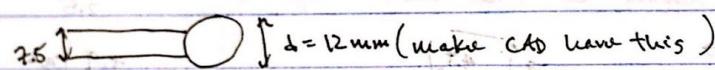
- Discussed calibrated capacitive sensor vs. non calibrated
- We want to buy one of each to see if we can calibrate the non calibrated one ourselves
  - Ordering CFSs with calibration
- Tasks:
  - test/interface with calibrated capacitive force sensor: Matthew/Yash
  - write/test out code to calibrate non calibrated capacitive force sensor: Matthew/Yash
  - write/test code to calibrate new FSR: Matthew/Yash
  - Some CAD designs
    - CAD with common parts: Akhil
    - “Implementation 1.5”: hybrid washer sandwich and plunger: Albert
    - Screw in end cap going into plunger/washer sandwich: Sala
    - Varun’s boltless design with washer sandwich: Varun
    - Anything else we think of
  - 3D print CAD designs for next Friday

## Node Redesign Mockup

- Filepath for CAD files: /Users/salatiemann/Documents/UC Berkeley/Year 3 Sem 2 (Spring 2020)/node redesign

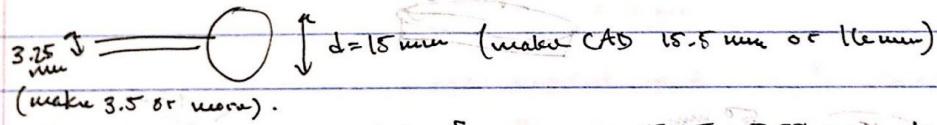


NEW FSR

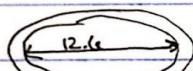


If want 3 mm hole, dimension hole to 3 mm; tap by hand.

FSC



hole for screw: 5.85-5.88 mm diameter



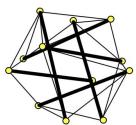
Scanned with CamScanner

# Berkeley Emergent Space Tensegrities

# FSR ARRAY DESIGN REVIEW MEETING

2.14.20 / 1:00-2:00PM / 230 Hesse

## Presentation



### Tensegrity Prototype V2 Force Sensors & Implementation

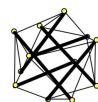
Matthew Kanter  
Sala Tiemann  
Akhil Padmanabha  
Albert Lee  
Varun Save

February <3 14<sup>th</sup>, 2020



## Presentation Outline

- **Ruled Out Sensors**  
Can't be Implemented
- **Recommended Sensors**  
Can be Implemented
- **Implementation Methodology 1**  
Washer Sandwich
- **Implementation Methodology 2**  
Bar and Plunger
- **Implementation Methodology 3**  
Boltless



2

## Ruled Out

- **Load Cells**  
Too Costly (~\$500/sensor)  
Too Heavy (~0.2 lbs/sensor)  
Large Accompanying Electronics
- **Optical Sensors**  
Typically used for Robotic Manipulation Tasks  
Very Costly (Upwards of \$3K)  
Large Accompanying Electronics

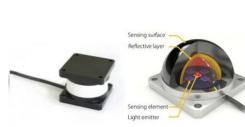


Image Retrieved From:  
<https://www.micromechanicalloadcells.com/>  
<https://www.transducertechniques.com/the-load-cell.aspx>  
<https://trends.directindustry.com/project-16720.html>  
<http://www.quadrature-ltd.co.uk/captoforce-force-sensing-membrane/>

3

## Capacitive Force Sensors

- Pros
- less error than current FSR's
  - higher force range (no springs needed)

Cons

- more complicated electronics needed
- \$70 for calibrated sensor with electronics



- Specifications
- Force Range: 0 to 450 N
  - Size: 15 mm diameter
  - Repeatability Error: <1%

Link: <https://www.singleact.com/micro-force-sensor/standard-sensors/15mm-450newton/>



4

## Better FSRs

### Pros:

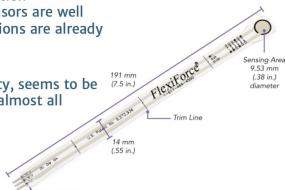
- Small size allows for easy implementation
- Current model uses FSRs, so these sensors are well understood and possible implementations are already in place

### Cons:

- In terms of sensing power and accuracy, seems to be worse than capacitive force sensor in almost all regards

### Specs for FlexiForce A201 Sensor:

- Force Range: 0–111N
- Repeatability: <±2.5% error
- Dimensions: 14 mm width, 9.53 mm diameter, trimmable length (maximum length: 191 mm)
- Cost: \$73.30 for 4 pack, 131.80 for 8 pack



Berkeley  
UNIVERSITY OF CALIFORNIA

5

## Sensor Implementation Methodology 1 : Washer Sandwich

### Advantages

- Eliminate need for force divider
- FSR can be larger than diameter of bar (no size constraint)
- Hold FSR in place with surrounding strings
- Can sense all forces, i.e. those from node compression and spring tension
- No movement, no friction

### Disadvantages

- Completely new design



Berkeley  
UNIVERSITY OF CALIFORNIA

6

## Sensor Implementation Methodology 2 : Bar & Plunger

### Advantages

- Eliminate need for force divider
- Plunger covers top surface area of FSR
- No major reconfigurations from original design – only redesign screw element
- No movement, no friction

### Disadvantages

- Need to fit FSR inside bar (size constraint)
- Modify bar to allow FSR wires through



Berkeley  
UNIVERSITY OF CALIFORNIA

7

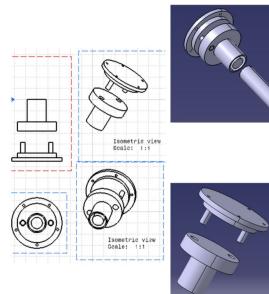
## Implementation Methodology 3 : Boltless

### Advantages

- Eliminates need for a force divider
- Size of force sensor not constraint by rod diameter
- No Bolt needed
- No movement, no friction

### Disadvantages

- Fundamental design changes required

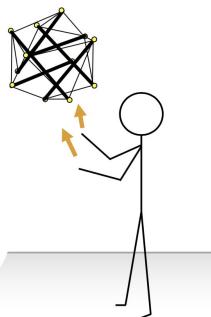


Berkeley  
UNIVERSITY OF CALIFORNIA

8

## Conclusion

- **Sensor**
  - Buy samples of the better FSRs and the Capacitive Force Sensors
  - We recommend the Capacitive Force Sensors for 2nd iteration prototype
- **Implementation**
  - No recommendation yet
  - Depends on the sensor



Berkeley  
UNIVERSITY OF CALIFORNIA

8

# Berkeley Emergent Space Tensegrities

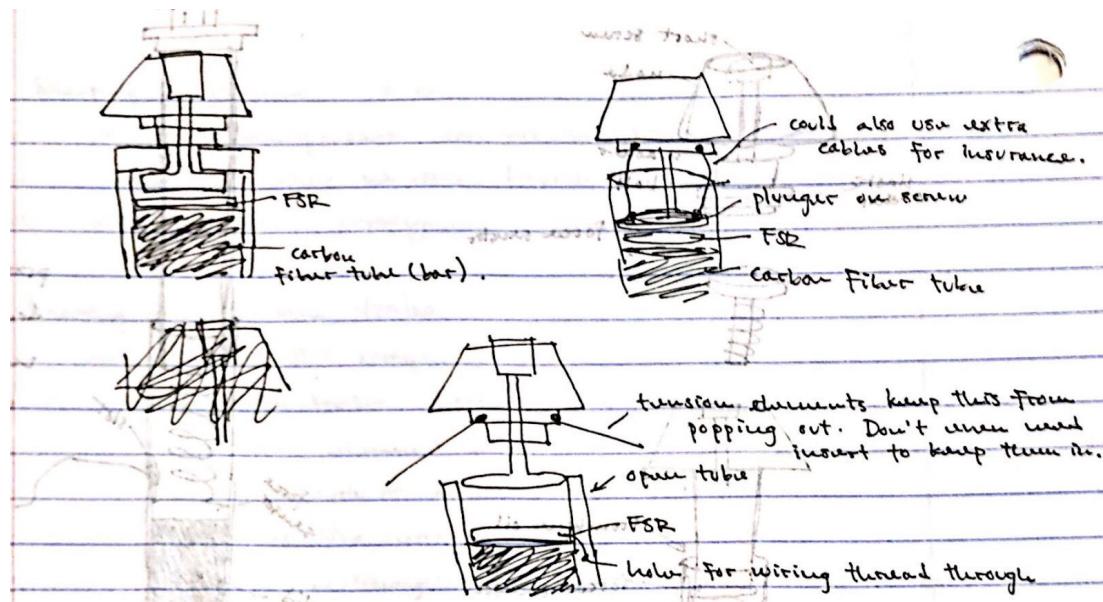
# DESIGN MEETING

2.7.20 / 1:00-2:00PM / 230 Hesse

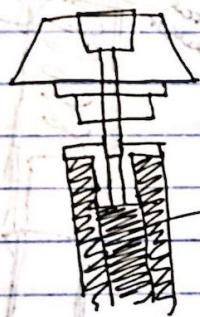
## Meeting Notes (Transcript from Bi-Weekly Tensegrity HRI Meeting Notes)

- Load cells are too expensive
- Akhil's force sensor comparison
- Redesign ideas for implementation:
  - Redesign end cap out of ninjaflex
  - Reimagine plunger system
  - Bigger washer "sandwich" by threading rope through both washers
  - Redesign end cap to house sensor and run ropes through end cap to ensure we still maintain rope tension
- Slides for Friday presentation: bad ideas, FSRs, capacitive force sensors, implementation ideas -- conclusion on best idea

## Node Redesign Mockup

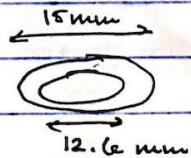
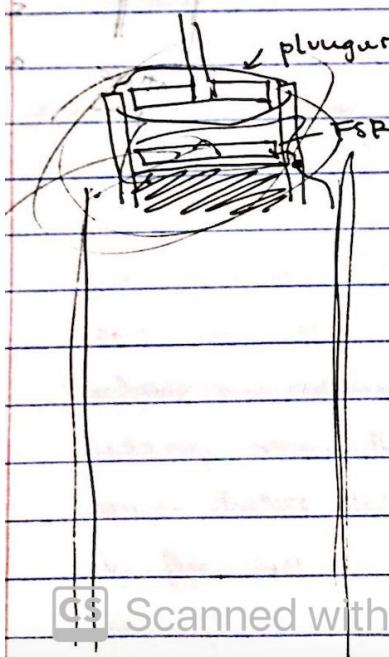


original valve design:

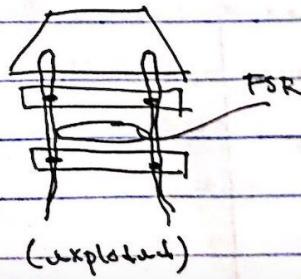
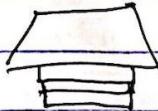


if we have this carbon fiber tube (bar),  
we can eliminate the insert and fit  
the force sensor inside the bar.  
Same logic as original design  $\Rightarrow$  don't need  
anything holding plunger in except  
tension elements.

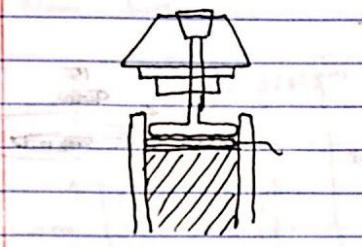
plunger. Only downside: might have more friction.



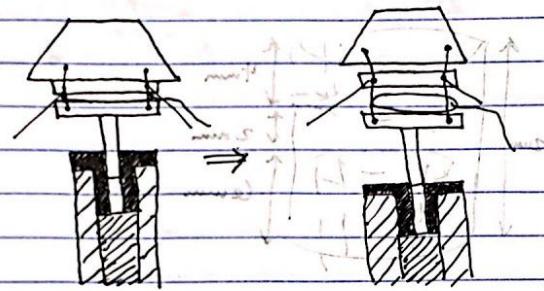
ideas: repaint end cap to eliminate need ~~area~~ for screw.



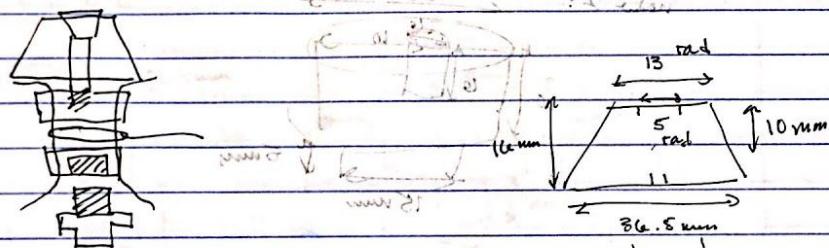
Scanned with CamScanner



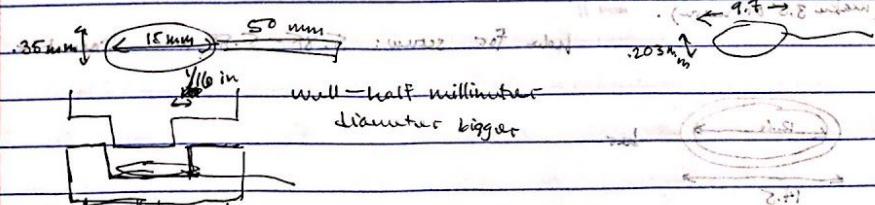
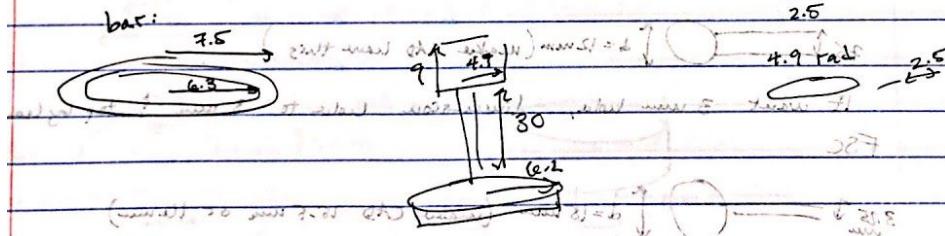
same concept as original  
design, but no need for  
insert because we have  
plunger, which covers  
the surface area of  
the FSR.



re-design cap to be  
able to connect to  
strings. Hold FSR in  
place by surrounding  
it with strings



washer: 22 mm diameter



Scanned with CamScanner

- Filepath for CAD files: /Users/salatiemann/Documents/UC Berkeley/Year 3 Sem 2 (Spring 2020)/end cap design



# Berkeley Emergent Space Tensegrities

# DESIGN MEETING

---

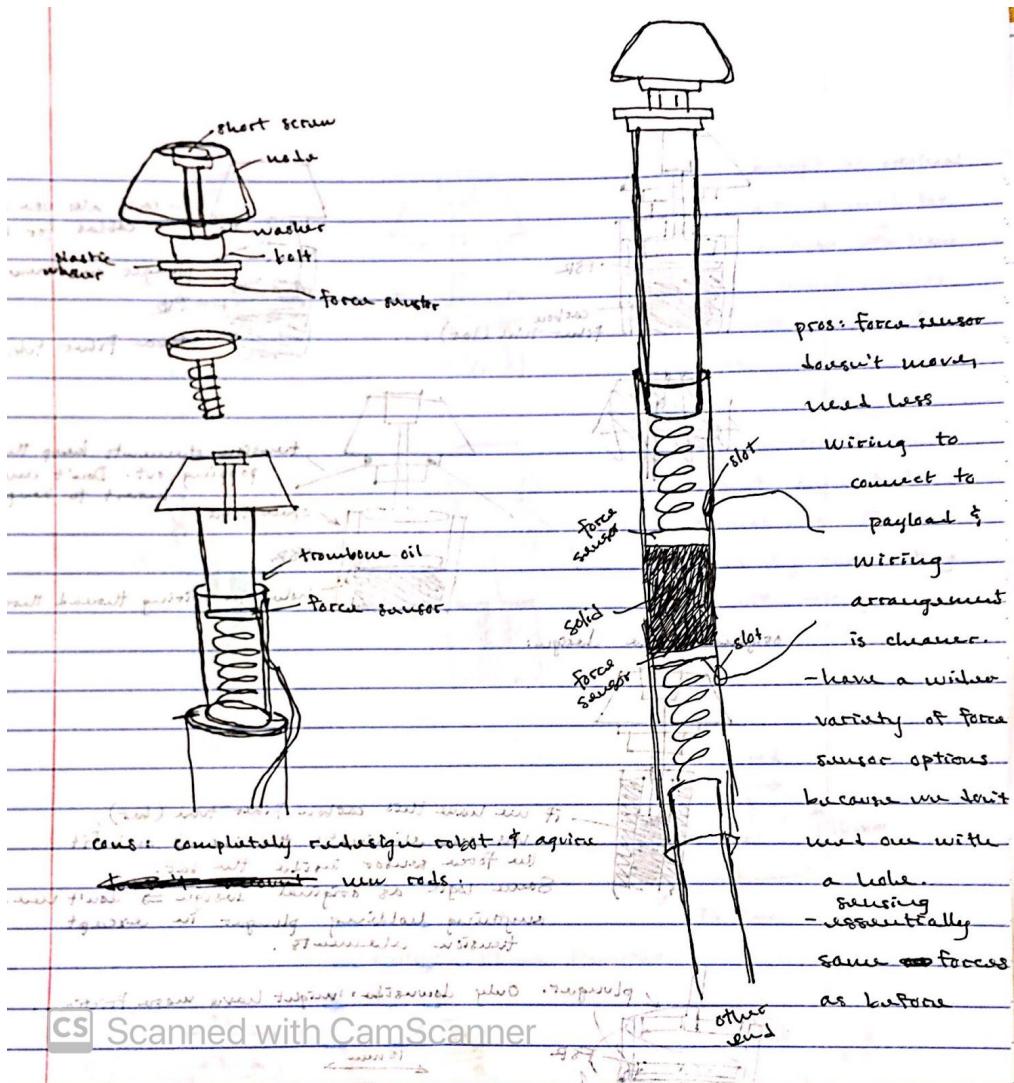
2.3.20 / 1:00-2:00PM / 230 Hesse

## Meeting Notes (Transcript from Bi-Weekly Tensegrity HRI Meeting Notes)

- OptoForce sensors won't work because of large accompanying electronics
- Looking into better FSRs and capacitive force sensors
- Capacitive give absolute values but require small packaging
- Concrete ideas:
  - Capacitive force sensing
    - <https://www.robotshop.com/en/capacitive-force-sensor-15-mm-45-n-10-lbs.html>
  - Better FSRs (Albert)
    - <https://www.tekscan.com/store/category/force-sensors-flexiforce>
  - Load cells (Matt and Varun)
    - <https://www.transducertechniques.com/tha-load-cell.aspx>
    - <https://www.flintec.com/media/downloads/mba-tw-miniature-datasheet-en.pdf>
    - <https://www.futek.com/store/Load%20Cells/LTH/LTH300>
  - Current FSR
    - <https://www.digikey.com/product-detail/en/interlink-electronics/34-00065/1027-1034-ND/7672219>
  - Pressure-sensitive conductive sheet
    - [https://www.adafruit.com/product/1361?gclid=Cj0KCQiAvc\\_xBRCYARIAC50T9mqIlQL90nIdTFyc0Hf-zoJ6D0gCwAobY2htpIvyZw\\_SFjBmSRPpJYaAsVCEALw\\_wcB](https://www.adafruit.com/product/1361?gclid=Cj0KCQiAvc_xBRCYARIAC50T9mqIlQL90nIdTFyc0Hf-zoJ6D0gCwAobY2htpIvyZw_SFjBmSRPpJYaAsVCEALw_wcB)
  - Combinations of sensors, e.g. FSRs + vibration sensors
  - Akhil's force sensor comparison

- <https://docs.google.com/spreadsheets/d/17eA4m2vv65IkrAYLVaAEp8R34-JoTdPCml7lj6MaIwQ/edit#gid=0>

## Node Redesign Mockup



Berkeley Emergent Space Tensegrities

# TENSEGRITY HRI MEETING

---

**1.28.20 / 12:00-1:00PM / 230 Hesse**

## **Meeting Notes**

- Deliverable for ideas for improvement on FSR array
- Andrew will be here Friday afternoons on biweekly basis
- 5 solutions each

Berkeley Emergent Space Tensegrities

# TENSEGRITY HRI MEETING

---

1.23.20 / 1:00-2:00PM / 230 Hesse

## Meeting Notes

- Next iteration FSR array:
  - Issues with friction (screws)
  - Make node elements more compact (less interference)
  - Critical analysis of different types of force sensing
    - Weight, size restrictions, budget
  - Options for switching/or why we should/not change it up
- Ideas for moving forward/changing force sensor array
- Ideas for how to use gyroscope data in conjunction with FSR data to get information that we couldn't get from gyroscopes alone
- Can classify which nodes it falls on with FSR data

Berkeley Emergent Space Tensegrities

# WEEKLY UPDATE

---

**1.13.20**

## My Update

- Have been looking at data on repository and using Matthew's truncating script to isolate the action events
- Didn't see anything uploaded on the drive, so started from the beginning and worked through all the data from the 10/18 and 10/21 folders
  - Drop test data seems to have only one action event per file but squeeze data has 12 -- was that on purpose?
  - Truncated data once to include all action events and then individually separated each squeeze set of data
- Uploaded the .csv files of the truncated data and the corresponding FSR plot
- Had the idea to work on not only classifying the type of interaction with the tensegrity but also the primary nodes involved in the interaction, given that the data given by each node is very distinct depending on the orientation
- Was able to connect to the internet using an ethernet connection on my Linux partition
- Followed setup instructions on the Spherical Tensegrity readme, but someone should probably check my work (not sure if I was successful in setting up Teensyduino)

# Berkeley Emergent Space Tensegrities

# TENSEGRITY HRI MEETING

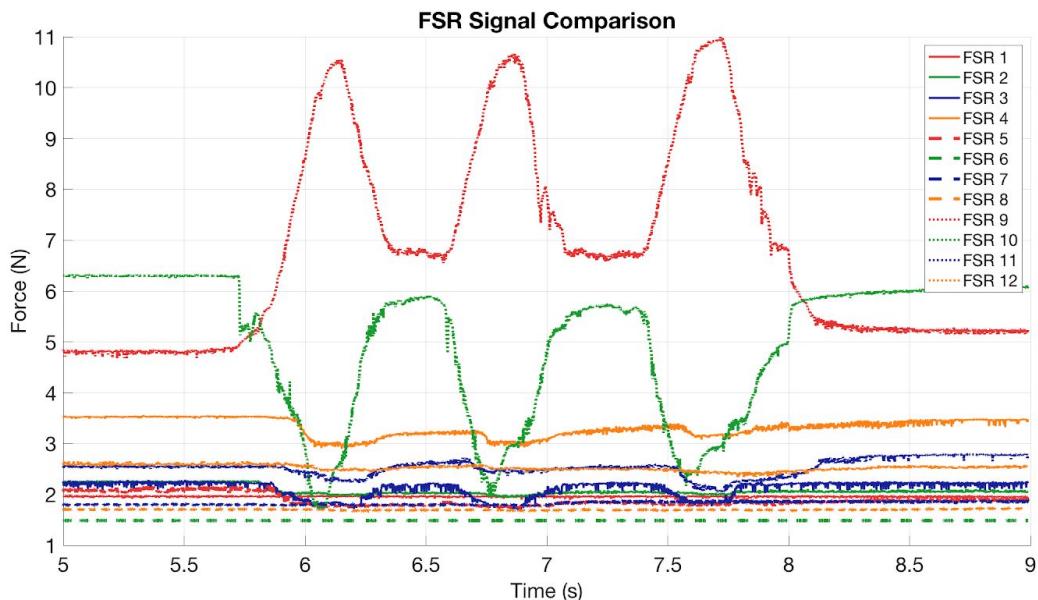
1.7.20

## Meeting Notes

- pandas python package (data processing)
- Conference paper submission March 1st
- 2nd iteration prototype

## Current Work

- MATLAB script to trim datasets
  - Filepath: /Users/salatiemann/Documents/UC Berkeley/Year 3 Sem 2 \ (Spring 2020) /BEST/fsr\_comparison\_visual\_with\_csvexport.m
  - 1 second of data before and after the action event begins and ends
  - Adjust name of file, adjust range, export data to .csv



Berkeley Emergent Space Tensegrities

# WEEKLY UPDATE

---

**1.3.20**

## My Update

- Looked at Spherical Tensegrity repo, but unable to set it up because Linux partition is having trouble with WiFi again
  - Hope that it's just because my home WiFi is flaky
- Have been looking at Linux tutorials to help familiarize myself
- Reading up more on machine learning looking at scikit learn tutorials
  - <https://scikit-learn.org/stable/tutorial/index.html>
  - Working less in data analysis because it seems like Akhil and Matthew are amending the data so far, but will get started with that soon!

Berkeley Emergent Space Tensegrities

# WEEKLY UPDATE

---

**12.24.19**

## My Update

- Started looking at Python sci kit learn documentation to get acquainted with machine learning
  - <https://scikit-learn.org/stable/>
- Watched a 3Blue1Brown series on neural networks
  - <https://www.youtube.com/watch?v=aircAruvnKk&t=3s>
- Planning on getting more familiar with Linux now that it's all set up and looking at the data on the repository

## Andrew's Assignments

- Look at setup instructions for the Spherical Tensegrity repo -- want to finish getting set up with software stack

Berkeley Emergent Space Tensegrities

# TENSEGRITY HRI MEETING

---

**12.5.19 / 3:00-4:00PM / 230 Hesse**

## **Meeting Notes**

- Get familiar with Linux, make sure everything is stable
- Basic commands
- Expect email by Monday every week regarding progress
- Operating exclusively in Jupyter notebooks
- Ubuntu 18.04
- Anaconda to install Python 2.7

Berkeley Emergent Space Tensegrities

# TENSEGRITY HRI MEETING

---

**11.18.19 / 3:00-4:00PM / 230 Hesse**

## Meeting Notes

- Lots of data from experiments! Large datasets
- Want to distinguish between interactions on trained data
- Next semester: heavily involved on current system and make 2nd prototype
- Idea! With data, can create simulation of what robot looked like during interaction
- Machine learning with Python/Jupyter -- sci py
- Come up with solutions to problems that are currently coming up with current prototype -- wires popping out
- Serious design project to iterate on prototype
- Idea that I had before -- push node 1 3 times to interact
- 6 axis IMU gyroscope -- translational and angular acceleration
- Noise combing
- Model icosahedron in MATLAB -- Simulink??
- learning/having Linux? By end of Thanksgiving break?
  - Version, packages
- Python for data processing. Sci-kit-learn (Python ML packages)
  - “Super easy it’s like... make model... dot... do”
- Train model on 200 demonstrations
- Debugging code with Arduino C++
- Bring Arduino home!
- Take data from force sensors and gyroscope, simulate entire motion of robot → real time

Berkeley Emergent Space Tensegrities

# ANDREW'S QUALIFYING EXAM PRACTICE

---

11.6.19 / 4:00-5:00PM / 230 Hesse

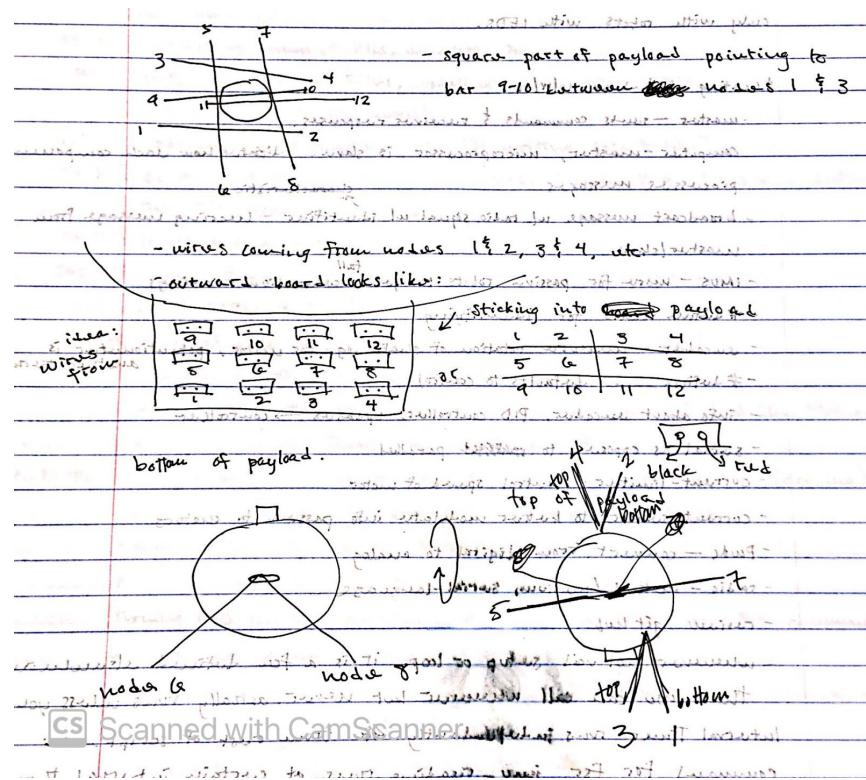
## Presentation Notes

- Challenges: promoting context and modeling soft robotics
- Compliant -- deformability, as opposed to soft components (soft robots)
- Tensegrity as a platform for HRI -- compliant, configurable, robust, mobile. Don't yet have good HRI
- Goal 1: design force-sensing tensegrity platform. Reliably detect variety of physical interactions
- Goal 2: interaction model in which we can distinguish among different interactions (types of physical interactions) using a force-sensing tensegrity
- Two parallel spring elements to extend range of force sensors
- Use demonstrations and classify them -- train classification model
- Use raw FSR readings to find tangible features
- Drop tests very distinguishable from squeeze tests, in terms of max force, oscillation frequency, and total impulse
- Gather large dataset spanning a variety of interactions for training
- Task-oriented HRI experiments with uninformed users to assess intuitiveness
- How much help do the robots offer in a disaster scenario, based on efficiency analysis?

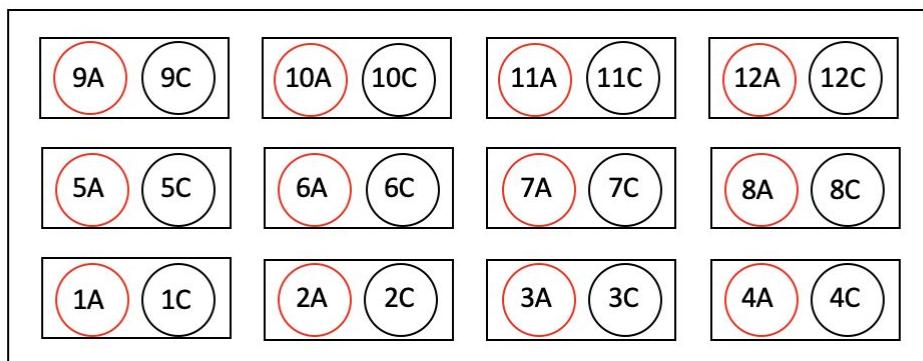
## Berkeley Emergent Space Tensegrities

# DEVELOPING CONVENTION FOR ROBOT

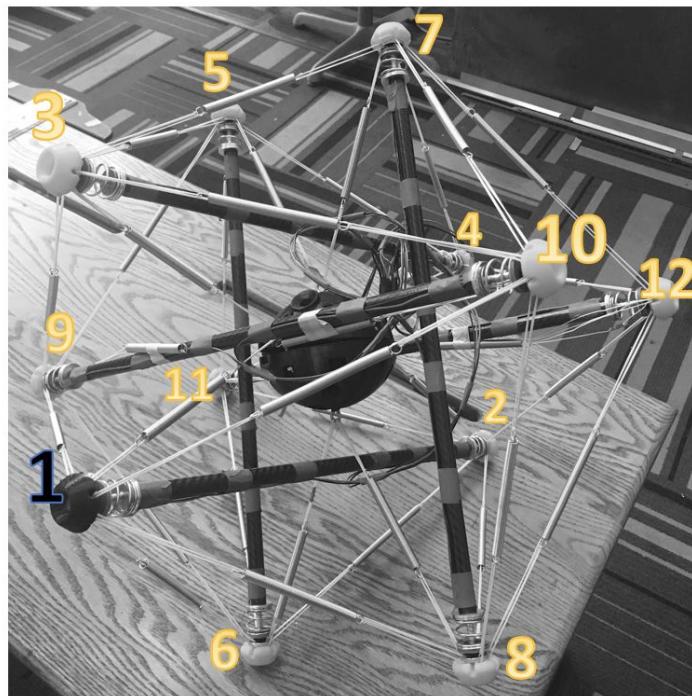
10.28.19



Labeling denotes anode and cathode coming from the indicated node. E.g. 1A indicates the anode (red wire) from node 1



Node numbering convention:



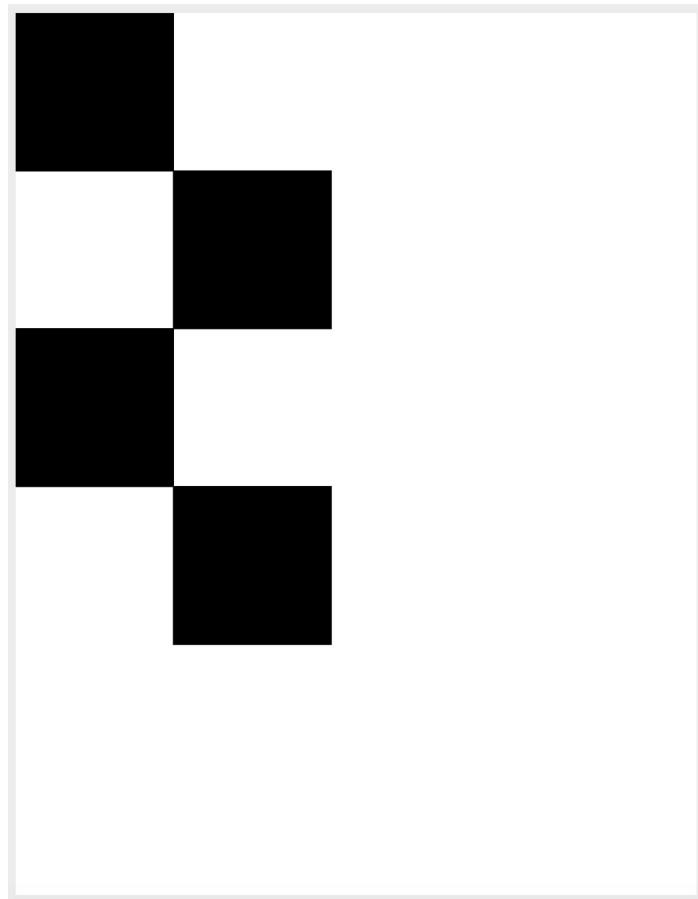
Berkeley Emergent Space Tensegrities

# EXPERIMENTAL

---

10.26.19 / 3:00-5:00PM / 230 Hesse

**Scale for Videos:**



**Test Node Configuration Document:**

- Closed-face
  - 1. 2-6-11
  - 2. 2-8-12
  - 3. 1-8-10
  - 4. 3-7-10
- Open-face
  - 5. 2-6-8
  - 6. 10-8-12
  - 7. 1-3-10
  - 8. 3-5-7
- Rod
  - 9. 2-4
  - 10. 6-8
  - 11. 10-12
- Squeeze
  - 12. 2-6-11
  - 13. 2-6-8
  - 14. 6-8
  - 15. 6

Berkeley Emergent Space Tensegrities

# ONE-ON-ONE WITH ANDREW

---

10.14.19 / 2:00-3:00PM / 230 Hesse

## Meeting Notes

- Master -- sends commands and receives responses
- Computer - master, microprocessor is slave. Dictates how each component processes messages
- Broadcast message with radio signal with identifier (characteristic) -- hearing message from master/slave
- IMUs -- more for passive robots to get fall acceleration readings
- #define macro for identifying
- Encoder -- measure rotation of shaft, e.g. for motor, potentiometer is another term
- # define \_\_ defaults to control
- Info about encoder, PID controller, updates → controller
- Serial as opposed to parallel
- Current-limiter -- control speed of motor
- Current sensors to better modulate info passed to motors
- PWM -- convert from digital to analog
- Radio -- method/medium, serial-language
- Review github
- Whenever not void setup or loop, it is a function defined elsewhere that you can call whenever but never actually runs unless you call it
- Interval timer runs independently from rest of script, command for fsr, imu\_reading runs at certain interval throughout running code

# Berkeley Emergent Space Tensegrities

# NOTES ON ARDUINO CODE

---

10.3.19

## Notes

- Filepath for folder: /Users/salatiemann/Documents/UC Berkeley/Year 3 Sem 1 (Fall 2019)/BEST/FSR\_Board
- CONSTANTS\_H:
  - #ifndef checks whether the given token has been #defined earlier in the file or in an included file; if not, it includes the code between it and the closing #else or the #endif
  - Defines CONSTANTS\_H as constants for Robot to use, default/initial values
  - Line 11 Are “slaves” other controllers that link to master controllers? Why are there so many slaves? Do they each do different specific things for the robot?
  - # define \_\_ \_\_ is search and replace
  - int \_\_ \_\_ is a variable in which the sketch can modify that variable under desired circumstances
  - const int is read-only → can't be changed in script
  - PID -- proportional-integral-derivative controller. Closed loop mechanism employing feedback widely used in industrial control systems and other applications
  - 17 What is IMU\_QUAT? Why microseconds? ENCODER?
  - 27-29 feedback constants?
  - 24 What is LED for?
- Board.h
  - 4 important constants/definitions from “Constants.h” file
  - PCB -- printed circuit board
  - Sometimes with #define, have identifier, replacement. Sometimes only have identifier, sometimes have both. What is the difference,

and how come you use #define without a replacement? Also e.g. wrote #define ROBOT FSR\_BOARD before #define FSR\_BOARD, 6. How does that work? What kind of object is the replacement?

- 14 is this calling the whole FSR\_BOARD.ino file? i.e. controller scheme
- 16-23 different robot options. Types of robots. Difference between TT5 and TT4 motors?
- 29 if robot type = TT4, import stuff from that file, define input/output pins for specific board
- 33 what do encoders do?
- 51-54 initializing arrays for PID control
  - Does “do use” imply that PID control is being defined in one of the imported files but it isn’t necessarily called until you do this?
- 56-58 initializing motor position limiting commands
- 62 DRV8833 variable defined in #include file, updating array of motor data
- 63 updating array of encoder data. What is this?
- 65-69 “&” is address, acts as pointer
  - This is the feedback. 65-68 updating controller
- 71 what does this line do (serial)
- If TT5 Robot, include these existing files
- 81-84 LEDs for TT5 robot \*\*\* is “LED” defined elsewhere? Why need to define again?
- 88 what is PHS? i.e. MOTOR1\_RHS
- 86-104 Motor encoders -- what do they do? What do LEDs blink for?
  - Defining which pins correspond to which motors
- 109-112 initializing PID control
- 114 what is PWM?
  - Pulse width modulation -- use electricity to control analog devices via digital signal

- 116-118 why do we need motor limits on this robot, as opposed to the other one? What physically does this do?
  - 121-128 feedback control
  - 131-136 updating controller with limiting values
  - 137-143 from <ACS712.h> file, write data from sensors
  - 147-151 or if robot using is TT5 with 6 motors, include these packages
  - 154-157 also uses LED array, defines which pins
  - IMU -- inertial measurement unit. Motion sensor basically. Reports a body's specific force, angular rate, sometimes orientation
  - 159-188 defining the pin locations of encoder, PWM, current motors for each motor (?)
  - 193-196 initializing arrays for PIDs, encoders, PWMs
  - 204 stuff from imported files, updating values written to PIDs
  - 207-213 not sure what REVERSE means?
  - 216-223 updating current limiters
  - 224-231 writing data to file?
  - 236-243 or if robot is the TT5 with 12 motors, include those files and define the LED array which is empty
  - 246-257 locations of motors
  - 265-278 array of motor information. Is this data or writing to motors?
  - 282 or if ROBOT is the FSR board, initialize LED array
  - 289-300 a lot of pins
  - 306-309 Wait... is this the data collection?
    - So the robots with motors are the mobile robots, correct?  
And the FSR one is new code for the new force sensor robot
- FSR\_BOARD

- Arduino code -- writing to board (.h file [C++) vs .ino file [Arduino]) including built-in packages, Encoder, other scripts and packages
- #defines ROBOT\_TYPE and FSR\_BOARD from other files
- No idea what Adafruit does
- Turning LED on, not sure what serial commands do
  - used for communication between the Arduino board and a computer or other devices
  - serial.printIn establishes connection -- prints confirmation that things are working
- No idea what Baud Rate/Xbee Channel are
- Error messages if connection to IMU (BN0055) detected
- Switching on and off LEDs but not sure why
- Reading messages from the master board
- Replying to Echo? Encoder reading?
  - If slave board receives new rates from master, sets update
- Broadcast from master?
- Repeats every millisecond
- Parallel loop -- a second void? Sends readings from IMU sensors
- Another parallel loop for fsr sensors. Sensors on both sides of each of 6 pins, and the readings are collected with simple analogRead(FSR1\_PIN) command. Sends readings from all 12 sensors
- led.ino
  - Include Board.h not sure what LED script tells LEDs to do but has to do only with robots with LEDs

Berkeley Emergent Space Tensegrities

# ONE-ON-ONE WITH ANDREW

---

**10.2.19 / 2:00-3:00PM / 230 Hesse**

## **Meeting Notes**

- Longer rod is cathode
- Form for Machine Shop and Hesse access
- Look at Arduino code on current board, practice parsing code
- Sets up board and calibrates, be able to make adjustments in code and run own experiments

# ARDUINO INFORMATION SEARCH

---

9.26.19

## Notes

- Microcontroller platform program to output certain things when you input certain things
- Connect sensors, code inside, trigger events. Connect to LEDs, motor, etc.
- Open source -- anyone can manufacture or modify Arduino hardware
- Arduino widely used for prototyping -- eventually design device with components they need for specific device
- BOARD is: write code in IDE and upload onto board, where main power for system goes into, where sensors/modules/shields connect to
- Shields connect directly to arduino board to add functionality
- Autonomous things -- sensors in locations farther away from arduino board get data from board → modules/sensors using wires
- Proximity sensor for autonomous vehicles
- Modules -- put arduino board in 1 location, control motors or servos from different location: connect motors/servos to motor/servo module, put arduino board to module. Modules add functionality, use sensors (can connect to many devices)
- Breadboard allows you to create projects without soldering
- Servomotor -- can tell specific position to go to
- TUTORIALS:
  - EEEEnthusiast
  - Eli the Computer Guy -- Arduino Introduction
  - Robert Paz: Introduction to Embedded Systems (EE260)
  - Jeremy Blum: Tutorial 01 for Arduino
  - Programming Electronics: Tutorial 01 Hardware Overview

Berkeley Emergent Space Tensegrities Laboratory

# ONE-ON-ONE WITH ANDREW

---

**9.24.19 / 1:00-2:00PM / 230 Hesse**

## **Meeting Notes**

- Get familiar with Arduinos
- Weekly meeting Wed @ 2PM
- Submit keycard access to 6th floor Etcheverry
- squishy/BEST meeting next week
- Machine shop training! And also experience machining
- Teensy 3.2

Berkeley Emergent Space Tensegrities Laboratory

# ONE-ON-ONE WITH ALAN

---

9.10.19 / 3:00-4:00PM / 230 Hesse

## Questions

- What does the team look like? Is it a lot of grads and undergrads working together or are you the main supervisor on the project?
- What are you specifically working on/most interested in?
- Do you interact/work directly with Dr. Agogino a lot? Do you have group meetings on a regular basis?
- If I am working on a specific aspect of the project such as manufacturing/controls, will I have the opportunity to learn about other aspects such as the software you are designing for the robots?
- You mentioned you have projects in mind that could fit with my skill set. What are those?
- What are long-term applications of these projects? What is the specific long-term application of the project/s I would be working on?
- What does mentorship look like in this lab?
- What is the balance between individual and collaborative work?

## Tasks

- Read the article on using Squishy robots to explore Saturn's moon Titan
  - Easier land on/explore moon, deploy scientific equipment
- Deep learning of tensegrity locomotion