

Fall 2015

CREATING A SMARTPHONE APPLICATION
FOR MEASURING RESPONSES OF AN
EXPERIMENTAL STRUCTURE AT
MULTIPLE LOCATIONS AND FOR K-12
STEM OUTREACH RELATED TO
STRUCTURAL ENGINEERING

Kyle David Wyatt

University of New Hampshire, Durham

Follow this and additional works at: <https://scholars.unh.edu/thesis>

Recommended Citation

Wyatt, Kyle David, "CREATING A SMARTPHONE APPLICATION FOR MEASURING RESPONSES OF AN EXPERIMENTAL STRUCTURE AT MULTIPLE LOCATIONS AND FOR K-12 STEM OUTREACH RELATED TO STRUCTURAL ENGINEERING" (2015). *Master's Theses and Capstones*. 1054.

<https://scholars.unh.edu/thesis/1054>

This Thesis is brought to you for free and open access by the Student Scholarship at University of New Hampshire Scholars' Repository. It has been accepted for inclusion in Master's Theses and Capstones by an authorized administrator of University of New Hampshire Scholars' Repository. For more information, please contact nicole.hentz@unh.edu.

CREATING A SMARTPHONE APPLICATION FOR MEASURING RESPONSES OF AN
EXPERIMENTAL STRUCTURE AT MULTIPLE LOCATIONS AND FOR K-12 STEM
OUTREACH RELATED TO STRUCTURAL ENGINEERING

BY

Kyle David Wyatt

Bachelor of Science, University of New Hampshire, 2012

THESIS

Submitted to the University of New Hampshire

in Partial Fulfillment of the Requirements for the Degree of

Master of Science

in

Civil Engineering

September 2015

This thesis has been examined and approved in partial fulfillment of requirements for the degree of Master of Science in Civil Engineering by:

Thesis Director, Dr. Tat Fu, Assistant Professor, Civil Engineering

Dr. Erin Bell, Department Chair & Associate Professor, Civil Engineering

Dr. Nicholas Kirsch, Associate Professor, Electrical and Computer Engineering

Dr. Ricardo Medina, Associate Professor, Structural Engineering

On May 20th, 2015

Table of Contents

LIST OF FIGURES.....	V
LIST OF TABLES.....	VII
ABSTRACT	VIII
CHAPTER 1 INTRODUCTION.....	1
1.1 COMPARING SENSORS HARDWARE	3
1.2 OUTREACH BACKGROUND.....	5
1.3 TIME SYNCHRONIZATION METHODS	6
1.4 PROJECT PURPOSE	8
CHAPTER 2 SMARTPHONE APPLICATION	11
2.1 RECORD TEST DATA	11
2.2 RECORD WITH TIME SYNCHRONIZATION	15
2.2.1 <i>Time Synchronization</i>	18
2.2.2 <i>Synchronization Method</i>	19
2.3 COMBINE CSV FILES	23
2.3.1 <i>Interpolating Data</i>	25
2.4 ANIMATE STRUCTURE BY TEST	29
2.5 CREATE A STRUCTURE	31
2.5.1 <i>Camera Angle</i>	33
2.5.2 <i>Displace the Structure</i>	33
CHAPTER 3 TESTING.....	36
3.1 MODEL STRUCTURAL INFORMATION	37
3.2 SMARTPHONE SETUP.....	39
3.3 TESTING CONFIGURATIONS	41
3.4 DAMAGED FLOORS	43
3.5 POST PROCESSING PROCEDURE.....	44
3.6 PRELIMINARY TIME SYNCHRONIZATION TESTING	44
3.6.1 <i>Initial Time Synchronization Testing Results</i>	45
3.7 AIRPLANE MODE TESTING	46
3.7.1 <i>Time Synchronization Testing with Airplane Mode</i>	50
3.8 STRUCTURAL ANALYSIS RESULTS	52
3.9 ANALYSIS OF RESULTS	54
3.10 SMARTPHONE SENSORS MOUNTING AND SYNCHRONIZATION COMPARISON	57
3.10.1 <i>Smartphone Synchronization Method Compared to Microstrain</i>	58
3.10.2 <i>Synchronization Error Affect on Smartphone Stiffness Values</i>	64
3.10.3 <i>Synchronization Error Affect on Microstrain Data</i>	66
3.10.4 <i>Smartphone Mounting Techniques Compared to Microstrain</i>	69
3.11 STIFFNESS RESULTS COMPARED TO A DEDICATED SENSING NETWORK	74
3.11.1 <i>Stiffness from Microstrain Sensors and Smartphone Sensors</i>	74
3.12 SUMMARY: STRUCTURAL ANALYSIS WITH SMARTPHONES	76
CHAPTER 4 OUTREACH PROGRAM.....	78
4.1 STUDENT BREAKDOWN.....	80
4.2 PRE-OUTREACH SURVEY.....	82
4.3 POST-OUTREACH SURVEY	83
4.4 THE PRESENTATION	83
4.5 ACTIVITY 1 – DETERMINING STIFFNESS AND DAMPING	85
4.6 ACTIVITY 2 – COUNTING OSCILLATIONS	90
4.7 ACTIVITY 3 - SPAGHETTI STRUCTURE TEST	92
4.8 RESULTS	96
4.8.1 <i>Fisher’s Exact Test</i>	97

4.9	CONCLUSION	98
CHAPTER 5	THESIS SUMMARY	99
5.1	ERRORS AND LIMITATION	101
5.2	FUTURE WORK	102
APPENDIX	104
	<u>APPENDIX A: EIGEN-SYSTEM REALIZATION ALGORITHM (ERA) WITH IMPULSE EXCITATIONS.....</u>	105
	<u>APPENDIX B: LEAST SQUARE ESTIMATE FOR STRUCTURAL STIFFNESS</u>	106
	<u>APPENDIX C: MULTIPLE REGRESSION ANALYSIS.....</u>	107
	<u>APPENDIX D: MATLAB CODE</u>	110
	<u>APPENDIX E: JAVA CODE – MAINACTIVITY CLASS.....</u>	115
REFERENCES	121

List of Figures

Figure 1-1 Thesis Contribution to the Field of Structural Health Monitoring.....	10
Figure 2-1 Record Data.....	12
Figure 2-2 CSV Header	13
Figure 2-3 Java Code “onSensorChanged” to Save Acceleration Measurement Into a File	13
Figure 2-4 Java Code “onSensorListenerUnRegistered” to stop the buffered writer	14
Figure 2-5 Java Code Set Time Delay	15
Figure 2-6 Record With Time Synchronization.....	16
Figure 2-7 Java Code to Set Phone to be Discoverable	17
Figure 2-8 Bluetooth Scanning	18
Figure 2-9 Time Protocol.....	20
Figure 2-10 Combine CSV Files.....	24
Figure 2-11 Time Difference	26
Figure 2-12 Line Up Data	26
Figure 2-13 Assemble Time and Accelerations.....	27
Figure 2-14 Overlap Section.....	27
Figure 2-15 Sample Data	28
Figure 2-16 Sample Data Graph	29
Figure 2-17 Animate Structure	30
Figure 2-18 Create a Structure	31
Figure 2-19 Drawing Structure	33
Figure 2-20 Displace the Structure	34
Figure 3-1 Experimental Structure.....	37
Figure 3-2 Spring and Turnbuckles (Cross Bracing).....	38
Figure 3-3 Lumped Shear Structure.....	39
Figure 3-4 Phone on Sticky Pad.....	40
Figure 3-5 Damage Configurations (Shaded Floors Are Damaged)	42
Figure 3-6 Damaged Floor.....	43
Figure 3-7 Time Synchronization Error.....	46
Figure 3-8 Synchronization Value Change with Airplane Mode On	47
Figure 3-9 Time Delay Test 5 Airplane Mode Off	48
Figure 3-10 Synchronization Values During Structure Testing	50
Figure 3-11 Raw Data without Time Synchronization	51
Figure 3-12 Synchronized Data	52
Figure 3-13 Graphical Structural Stiffness Changes 1	56
Figure 3-14 Graphical Structural Stiffness Changes 2	57
Figure 3-15 Microstrain to Smartphone Data Comparison - Raw Data	58
Figure 3-16 Microstrain to Smartphone Data Comparison - Raw Data Shifted by Eye.....	59
Figure 3-17 Script Variables – shift_range and RMS_diff.....	60
Figure 3-18 Calculate RMS Difference Values for Every Value of shift_range	61
Figure 3-19 Minimum RMS Difference Value and Location for Each Floor	62
Figure 3-20 Overall Setup - Sticky Pad Test	70
Figure 3-21 Sticky Pad Testing - Side View	70
Figure 3-22 Velcro Placement	71

Figure 3-23 Sensors on Left (Synchronized)	72
Figure 3-24 Sensors Right Side (Synchronized).....	72
Figure 4-1 Comparing Group Sizes (Small-Left, Large-Right).....	81
Figure 4-2 Outreach Quiz	82
Figure 4-3 Braced Structure.....	84
Figure 4-4 Damper Example.....	85
Figure 4-5 Blank Phone Activity	86
Figure 4-6 Activity 1 Phone Layout	87
Figure 4-7 Phone Activity Answers.....	88
Figure 4-8 Phones Structural Response (Fingers Represent an Applied Point Load)	89
Figure 4-9 Class Metal Structures.....	90
Figure 4-10 Displacing Class Structures.....	91
Figure 4-11 Structure Responses (as shown in class)	92
Figure 4-12 Spaghetti Structure "Uni Tatos"	93
Figure 4-13 Displacing the "Shake table"	94
Figure 4-14 Spaghetti Structure Results	95

List of Tables

Table 1-1 Sensor Comparison.....	4
Table 2-1 Example Synchronization Times.....	21
Table 3-1 Configuration 2: 6th Floor Damaged Results.....	53
Table 3-2 Smartphone Structural Stiffness Results	53
Table 3-3 Percent Change of Stiffness Relative to Healthy Structure (shaded boxes indicate damage locations)	54
Table 3-4 Location in shift_range to Align Smartphone and Microstrain Data	63
Table 3-5 Time Synchronization Error in Smartphones	63
Table 3-6 Stiffness Values with Adjusted Synchronization – Healthy Structure	64
Table 3-7 Microstrain to Adjusted Smartphone Stiffness Comparison	65
Table 3-8 Microstrain to non-adjusted Smartphone Stiffness Comparison.....	66
Table 3-9 Stiffness Values with Synchronization Error	67
Table 3-10 Top 3 Floors 1/256 Sync Error Vs Normal	67
Table 3-11 Top 3 Floors 3/256 Sync Error Vs Normal	68
Table 3-12 Every Other Floor Shifted 1/256 Sync Error Vs Normal	68
Table 3-13 Sticky Pad Testing Sensor Locations	71
Table 3-14 RMS Sensor Comparison	73
Table 3-15 Smartphones Stiffness Results	75
Table 3-16 Microstrain Stiffness Results.....	75
Table 3-17 Percent Stiffness Change - Smartphone Accelerometer.....	76
Table 3-18 Percent Stiffness Change - Microstrain Accelerometer	76
Table 4-1 Outreach Differences.....	78
Table 4-2 Student Breakdown by Class.....	80
Table 4-3 Activity 1 - Smartphone Breakdown per Class	81
Table 4-4 Enter and Exit Quiz Results	96
Table 4-5 Quiz Results.....	97
Table 4-6 Fisher Test – Amount Correct/Incorrect.....	97
Table 4-7 Fisher Test – Variable Assignment	98

Abstract**CREATING A SMARTPHONE APPLICATION FOR MEASURING RESPONSES OF
AN EXPERIMENTAL STRUCTURE AT MULTIPLE LOCATIONS AND FOR K-12 STEM
OUTREACH RELATED TO STRUCTURAL ENGINEERING****BY**

Kyle David Wyatt

University of New Hampshire, September, 2015

Structural health monitoring (SHM) systems are used to measure and analyze structure data (*e.g.*, floor accelerations and strains in structural members) to identify damage (or structural changes) to a structure. With aging infrastructures and collapses of recent structures such as the 2007 I-35W Mississippi River Bridge and the 2013 clothing factory in Bangladesh, SHM can help address an important societal issue in structural safety and reliability. In the current practice, SHM systems include dedicated sensors linked (via wires or wirelessly) to data acquisition systems. These sensing systems are typically costly and impractical for many educational curriculums. A lack of exposures to college students limits applications and understanding of SHM in the practicing engineering industry. By replacing these dedicated sensing systems with a common technology such as smartphones, this thesis project aims to make SHM experiments inexpensive and practical to college students. Additionally, the project can assist in exposing K-12 students to SHM and the general field of structural engineering at a young age and increasing their interest in becoming engineers.

This multidisciplinary research included developing a smartphone application using the Java™ programming language on the Android platform. The application utilizes the phone's user interface, internal accelerometer, internal storage, and Bluetooth to create a user friendly experience. One portion of the application is used for SHM purposes. It assists users in time-synchronizing multiple phones, recording acceleration data and detecting changes in structural properties. When compared to a dedicated sensing system used in a lab setting, data from the smartphones produced similar results.

Another portion of the application, incorporated into an educational outreach program at a local middle school, was designed to help students understand the basic concepts of structural dynamics — more specifically, how stiffness and damping affect a structure's motions. This interactive smartphone application, coupled with its ability to be a cost-effective system for measuring structural responses in classroom experiments, can get students excited about engineering.

Chapter 1 Introduction

Structural health monitoring (SHM) assesses a structure's integrity through analyzing structural response data such as accelerations and strains to detect changes in structural characteristics. Changes in structural characteristics can be caused by deterioration and damage in structures. Farrer *et al* (2007) describes damage as changes introduced into a system that adversely affects its current or future performance. Using SHM systems, a trained engineer can determine whether the structure is still capable of its function (healthy) or if it needs repair (damaged).

Traditional assessments of structural health were conducted by visually inspecting the members of a structure. As technology advances, engineers incorporated new sensing systems into structures to constantly monitor a buildings health to enhance longevity and safety. The technology takes away the interpretation of visual expectation and utilizes a dense array of sensors is deployed in specific locations throughout the structure to record responses such as accelerations and strains. Structural responses are then transmitted to a data acquisition system through either a wired or wireless connection. Once all responses are obtained by the system, a trained engineer can evaluate the responses and determine if the structure has any damage by comparing to previous data for the structure. Damage to a structure can be associated with the deterioration of a structural member. Identifying members in need of repair can assist in maintaining structures for a prolonged period of time. Although systems are highly effective, current SHM systems being implemented into structures have very high costs. In 2002, Celebi *et*

al mentioned some SHM systems can amount to a cost of \$4000- \$5000 per sensing channel. Although this is on the high end, it provides prospective on the overall costs for these systems.

Smartphones could eventually provide an alternative to expensive sensing systems because they provide an all-inclusive system capable of recording accelerations, synchronizing times, and saving data locally. Current methods of incorporating smartphones into SHM systems are strictly for experimental purposes. Recently, there have been a few studies investigating using smartphones to record structural responses. In 2013, Kotsakos *et al* used different smartphones to determine the natural frequencies of a structure by using the peak picking method. The authors were capable of detecting accelerations caused by the building's motions at natural frequencies using a sampling rate of 65 Hz and an Android Galaxy tab 2 7.0 accelerometer equipped device.

In 2015, Yu *et al* showed that iPhone devices could obtain accelerations for structural vibrations. A test attached four sensors to a structure floor to determine if all four would obtain the same accelerations. The four sensor devices were a wired, wireless, smartphone sensor, and an external sensor attached to a smartphone. It was found that all acceleration data compared well to one another.

Also in 2015, Feng *et al* used two iPhone devices (iPhone 3Gs and iPhone 5), an Android device (Samsung Galaxy S4) and a reference sensor (PCB Piezotronics NI SCXI-1531). All four devices were mounted to a shake table and accelerations were recorded. The author studied the recorded accelerations in a time history analysis which showed the issue of not having time synchronization for the devices. The study found the more recent devices were “significantly more accurate” than the older generations.

These recent findings showed that smartphones can record data that closely resembles data collected from the dedicated sensor systems. However, these previous efforts concentrated on collecting data using individual smartphones compared to a typical SHM system in which a network of sensors work together to record accelerations from multiple locations of a structure. Feng *et al* mentioned having a time synchronization issue when displaying data. Time synchronization is also one of the main issues when recording accelerations from a network of sensors or smartphones. Even when a network clock could be only a fraction of a second different from another device, it is not acceptable in structural dynamics.

1.1 Comparing Sensors Hardware

For acceleration sensors, there are commercially available dedicated sensors such as different models of Microstrain® accelerometers. Other universities such as the University of Southern California (USC) and University of Illinois Urbana-Champaign (UIUC), also developed SHM sensing systems. The following table shows a comparison of smartphones sensing system with sensing systems from the aforementioned company and universities.

Compared to the other dedicated accelerometers, the smartphone accelerometers are a far more cost effective option. Smartphone sensing systems are all inclusive; they have onboard memory, processors and wireless communication channels and the ability to work separately (such that they can be placed as far away from one another as needed) after they are time-synchronized (details will be discussed in Section 2.2.1 “Time Synchronization”). Meanwhile, the dedicated sensing systems are dependent on a data acquisition (DAQ) to obtain the data from the sensors. Microstrain sensors use a DAQ unit that costs an additional \$995 (“G-Link®”, 2015). Since the smartphones record data locally, data can be downloaded to a computer without

a DAQ. The Moto G smartphones also cost significantly less than the Microstrain accelerometers. Each Moto G smartphone costs approximately \$180, while each Microstrain accelerometer costs \$545 (“G-Link®”, 2015).






	Moto G - 2013 (Used) (LIS3DHMEM S, n.d)	Galaxy s4	Microstrain	USC (Chintalapudi, 2006)	UIUC (Rice, 2008)
Sensor Name	LIS3DH–3-axis accelerometer	K330 – 3-axis accelerometer	G-Link® - LXRS®	netSHM	LIS3L02AS4
Sensor Image					
Commercial Sensor Price:	\$180	\$300	\$545	n/a (not a commercial product)	n/a (not a commercial product)
Price: DAQ	None	None	\$995	n/a	n/a
Sensor Range	± 4g	± 2g	±2g or ±10g	±2.5g	± 2g
Sensor Resolution	12 bit	16 bits	12 bits	16 bits	16 bits
Sampling Rate	~100 Hz	~100 Hz	128 – 512 Hz	5 – 20000 Hz	280 – 4480 Hz
Battery	2070mAh	2600mAh	220mAh	n/a	196.2mW

Table 1-1 Sensor Comparison

However, there are a few drawbacks to smartphone accelerometers. Dedicated accelerometers designed to record accelerations for SHM purposes have a higher sampling rate, while also having a better resolution than some models of Smartphones. Also, smartphone accelerometers, due to the Android Operating System, do not have the capability of setting an

exact time interval between measurements. The code is just a suggestion to the processor to record data from the accelerometer at that certain rate. While dedicated sensors are capable of recording up to 1000 Hz (or data points per second), smartphones are currently limited to approximately 100 Hz. Considering that smartphone hardware is not designed for measuring structural vibrations, smartphones' specifications of sampling rate and resolution (bit rate) compare fairly well as a cost effective alternative to the sensors listed above for experimental structural analysis systems. As technology advances, it is expected that limitations of smartphones will decrease; and sampling rate should continue to get better.

1.2 Outreach Background

Given the smartphone application is also developed for outreach activities related to structural engineering, this section briefly discusses two separate outreach programs with an emphasis on structural engineering. To the best of the author's knowledge, there is not a published study using smartphones to help show structural engineering concepts to students. There are, however, applications available that allow students to design and build structures and apply loads.

The outreach conducted in 1997 by Carroll designed a bridge that would be assembled in an elementary school classroom. The bridge designed for the outreach was six feet long and two feet wide, with 29 total members. The students were given a presentation and asked to assist in constructing the structure. Once the bridge was constructed it was elevated onto two chairs per side and the students were permitted to crawl through. This project was also expanded to incorporate a ten foot long design capable of allowing teachers to easily cross. The purpose of

the outreach was to educate students on one of the many aspects of engineering while also introducing terminology and hands on experience constructing a bridge.

The outreach conducted in 2000 by Symans was designed to enhance K-12 student's interest in engineering, provide insight to structural engineer activities, and plant seeds to cause students to consider engineering as a future career. The day before, the students were shown a video to provide an introduction to engineering and showed how engineers are involved in a wide variety of projects. The day of the visit began with a presentation on careers in engineering and showed students that structural engineering was just one small part of the entire engineering community. After a brief questionnaire on the video presented to the students, they were then introduced to bridge design software. The software allowed the students to design a bridge, select member materials, run a load test, repair member failures, and optimize the bridge design by minimizing costs.

1.3 Time Synchronization Methods

Time synchronization was a large problem for analyzing a structure using smartphones. To remedy this situation, one technique was developed and a few others considered. The techniques were Epoch time, Network Time Protocol (NTB) or Simple Network Time Protocol (SNTP), Christian's algorithm, and Berkeley algorithm.

While Epoch time isn't a synchronization method, it is a common computer time that was initially assumed to match between devices. After conducting our own research on the topic, it was determined to lack sufficient accuracy for structural analysis due to the times varying

between devices. Other methods were combined with the Epoch time to synchronize multiple devices' epoch time.

Cristian's algorithm is the simplest method for setting a synchronization time. It simply sends a time to the node device and tracks the time it takes to get a reply. Typically, this method requires devices to ping packets of data off of a server. According to Krzyanowski in 2000, this method suffers from possible failure of the server or node, and also is subject to interference from delays inside the node device. This method, while the simplest was determined to have too much error for time synchronization.

According to Krzyanowski, the Berkley algorithm developed by Gusella and Zatti in 1989, takes the time from all devices, averages them together, and then synchronizes all devices to the averaged time. While this is a viable method, current coding experience made this method complicated to execute on the smartphones and was not used in the application.

The Network Time Protocol method utilizes multiple levels of stratum and multiple modes of synchronization to maintain several devices in synchronization. This method overall is far too complex to have been implemented in this project, however, a Simple Network Time Protocol method was used. A unicast mode (a client sends a request to a designated server) was used to calculate the round trip delay as well as the local offset. Using those times, the time difference between two selected devices could be calculated. To improve on the SNTP, multiple transmissions were conducted to help eliminate errors.

The method chosen of the proposed application resembles the Simple Network Time protocol for time synchronization and uses straight phone to phone communication, through Bluetooth with no additional loops or steps to synchronize. This method was chosen because it allowed for the synchronization process to occur once at the beginning, which freed the

processor to focus on data acquisition and also to be simple to implement in the smartphone application. One master device must be synchronized to each node phone, individually. For example, Phone 1(master device) would synchronize with Phone 2 (Node1) and subsequently with Phone 3(Node 2) and so forth. When synchronizing data in this method, Ping in 2003 cites synchronization needs to account for five main delay factors (each are described below in how they correspond to the developed application):

- (1) Sender Processing Delay – time it takes to buffer the packet for transmission.
- (2) Media Access Delay – time between buffering and sending between links.
- (3) Transmit Time – time to transmit the message over the Bluetooth link.
- (4) Radio Propagation Time – time to propagate over the air. Radio propagation is negligible in distances less than 100 meters.
- (5) Receiver Processing Time – Time for node to pass packet from master to cue the reply message.

While there are errors that could skew the time synchronization by a couple milliseconds the threshold determined for this application with this current method was .0117 seconds for an experimental or classroom setting. This method could be improved by communicating times down to the nano-second or a more advanced synchronization method.

1.4 Project Purpose

This project is different from prior studies on using smartphone in measuring structural accelerations with its capability of time-synchronizing between multiple devices and recording accelerations as a network. The developed application allows users to obtain accelerations from

multiple floors/locations of a structure and send data to a computer for post-processing and estimating any changes in structural characteristics between tests. With current limitations of the smartphone technology (sampling rate, sensor resolution), this application could be implemented strictly as an educational alternative in a lab setting. As technology and the application advance in sophistication other implementations and features could be added.

The availability of smartphones and the proposed smartphone application allows college educational programs to incorporate SHM concepts into their curricula as well as increase interest in Science, Technology, Engineering and Math (STEM) in K-12 schools. Several advantages of replacing dedicated sensors with smartphones include reduced cost, a programmable user interface, on board accelerometers, processors and storage, along with various wireless connectivity (e.g., WiFi, Bluetooth) to communicate with other devices. Along with the advantages above, abundance of smartphones among children aged 12 through 17 is increasing with 37% of all teens having smartphones in 2013, which is an increase from 23% in 2011 (Madden, Lenhart, Duggan, Cortesi, & Gasser, 2013). In 2014, Ransford found smartphone use since 2009 increased by 51% to 89%. Smartphone popularity among middle/high school and college students can provide educational programs with limited assets to utilize a resource already possessed by students.

As part of this thesis, the feasibility of using smartphones in a structural analysis situation will be tested. A portion of the application will be dedicated to determining the difference in epoch times between two sensors so accelerations can be recorded in a synchronized manner. A method for mounting smartphones to a structure will be created. The ground will be excited and accelerations from every floor will be recorded. The reason this is being created is to determine if smartphones could be included into a classroom setting in order to enhance learning related to

structural dynamics, and also to provide a less expensive, graphical and comparable alternative to current wireless accelerometer.

Figure 1-1 depicts the contribution of this thesis in regards to structural analysis. The created application will ideally replace the sensors and DAQ system in a structural analysis setup.

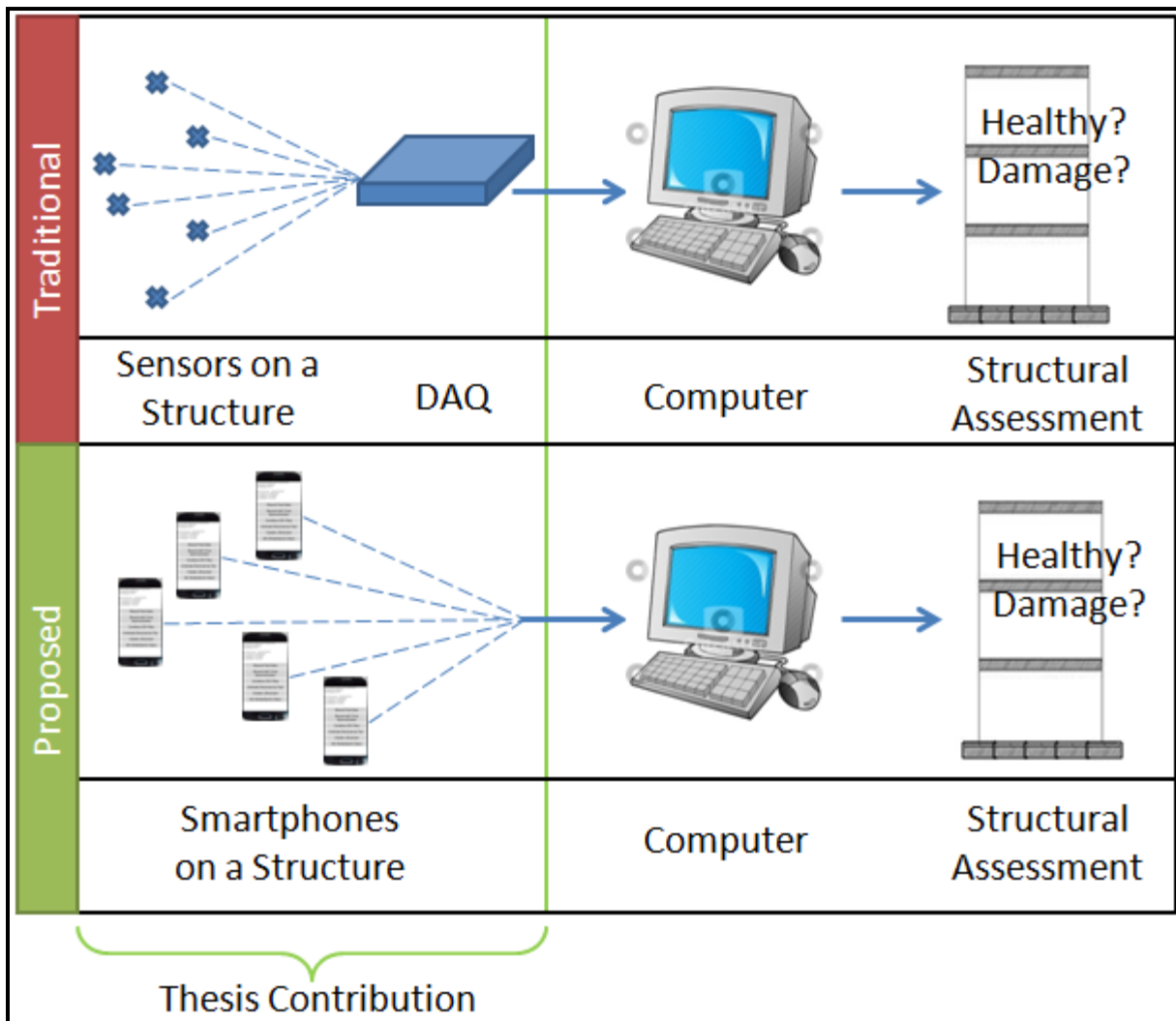


Figure 1-1 Thesis Contribution to the Field of Structural Health Monitoring

Chapter 2 Smartphone Application

The application was developed to accomplish two main objectives; (1) time synchronize with a network of smartphones to record data for structural analysis in a lab/educational setting (2) enhance students understanding of structural engineering concepts through visual learning and a hands on experience. The application is developed using the Java programming language, on Eclipse Juno—a programming computer application—for the Android platform. The application utilizes the phone’s touch screen interface, internal accelerometer, internal storage, processor and Bluetooth to create a user friendly experience. The activities used in the application are: (i) recording acceleration data, (ii) connecting devices via Bluetooth to synchronize time and send files, (iii) combining data files recorded by multiple devices, and (iv) providing an interactive animation module for displaying structure responses.

2.1 Record Test Data

The “Record Test Data” section of the application was developed to use a smartphone’s internal accelerometer to record acceleration test data and be post processed externally. In addition to experimental testing, this application could be used on individual devices by middle or high school students in a physics or science class to conduct hands-on experiments requiring accelerations.



Figure 2-1 Record Data

The “Record Test Data” portion of the application is laid out in Figure 2-1. The portion has to be accessed on the individual device to record data from the accelerometer. The top line of the screen is used for the user to enter a file name, and defaults to a file name associated with the current date and time of when the record data screen is loaded. There is a file list located at the bottom of the screen that lists every filename from the application folder for users to verify a filename has not been used. Between the file name line and the array of files on the secure digital card (SD card), are two buttons: “Start Recording Data” and “Stop Recording Data”.

When the “Start Recording Data” button is clicked, it becomes disabled to insure that multiple files cannot be created at once; then, the “Stop Recording Data” button is enabled and the file list turns red to signal the phone is recording see Figure 2-1 (Step 2). Internally, the phone registers a built-in Java function, “SensorManager,” to listen to accelerometer and obtain acceleration changes as fast as possible. Also, a CSV file (*i.e.*, a spread sheet formatted file) is created with metadata which contains the filename, date and time when the record button is clicked, the phone time synchronization information (discussed in Section 2.2.1 “Time

Synchronization”). In the metadata section of the data file, there are unused file/data properties that are reserved for future application expansion. For example, there is sampling rate property that will be used if the application allows users to choose a specific sampling rate. An example of the header is demonstrated in Figure 2-2.

X---Date/Time:	11_21_2014_13:19:18									
X---Number of Phones:										
X---Phone Epoch Information:	moto2	-702	moto3	-1331	moto4	-904	moto5	-416	moto6	-929
X---Sampling rate:										
X---Interested directions:	X	Y	Z							
X---Phone 1:	Xcoord	Ycoord	Zcoord							
X---Timestamp(ms)	X-Accel (nY-Accel (nZ-Accel (m/s^2)									

Figure 2-2 CSV Header

The phone writes the current epoch time, acceleration in the x-, y- and z-direction to the CSV file created by the user every time the phone registers a new acceleration. Epoch time is the amount of time in milliseconds since January 1st 1970. Code used to write time and accelerations to a csv file can be seen in Figure 2-3:

```
public void onSensorChanged (SensorEvent event)
{
    if(event.sensor.getType()==Sensor.TYPE_ACCELEROMETER){

        long epoch2 = System.currentTimeMillis();
        float x=event.values[0];
        float y=event.values[1];
        float z=event.values[2];

        try {
            bw.write(epoch2 + "," + x + "," + y + "," + z + "_ls");
            bw.flush();
        } catch (IOException e){}

    }
}
} //end OnSensorChanged
```

Figure 2-3 Java Code “onSensorChanged” to Save Acceleration Measurement Into a File

When the “Stop Recording Data” button is clicked, this button is disabled, “Start Recording Data” button becomes enabled again, the file list returns to a light gray color, and the user can record another set of data see Figure 2-1 (Step 3). Internally, the phone terminates the `SensorManager` to reduce battery usage. It also closes the file writer and saves the file to the SD card to be opened or transferred later which can be seen in Figure 2-4.

```
public void onSensorListenerUnregistered()
{
    try{
        bw.close();
    } catch(IOException e){}
} //onSLUR
```

Figure 2-4 Java Code “onSensorListenerUnRegistered” to stop the buffered writer

One issue with using the Smartphones to record data is the inconsistency in time stamps between data points. The average difference between time stamps or “ Δt ” is approximately .01 seconds, however this varies depending on when the phone receives acceleration from the sensor. Figure 2-5 is a suggestion to the processor to read the sensor value at a certain time. Currently the code directs the phone to read the acceleration values as fast as possible. Ideally, the code could be manipulated to record data on a set time stamp, however, Android does not currently allow for this kind of coding. In older APIs it was possible to set a recommended Δt , however, the recommendation was simply a suggestion to the processor and was depreciated in later releases.


```
public void createCSVFile(String filename) throws IOException
{
    //Set up the sensor
    sensorManager = (SensorManager) getSystemService(SENSOR_SERVICE);
    sensorManager.registerListener(this, sensorManager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER),
        SensorManager.SENSOR_DELAY_FASTEST);
}
```

Figure 2-5 Java Code Set Time Delay

2.2 Record with Time Synchronization

One major goal to be achieved during this thesis was to create an application that allows for a network of Smartphones to collect usable acceleration data without external assistance. For this goal to be possible, a method is needed for the devices to synchronize their clocks. Without time synchronization, the acceleration data would be out of phase and useless in accessing a structure's health.

In the "Record with Time Synchronization" portion of the application, one "Master" device is set to pair to the node devices, request their information, connect and synchronize time stamps, then record acceleration data.

The first step in this portion of the application is to connect the devices using Bluetooth. Moto G smartphones being used have Bluetooth 4.0 equipped on board. This version of Bluetooth requires the phone to use less battery than older versions.

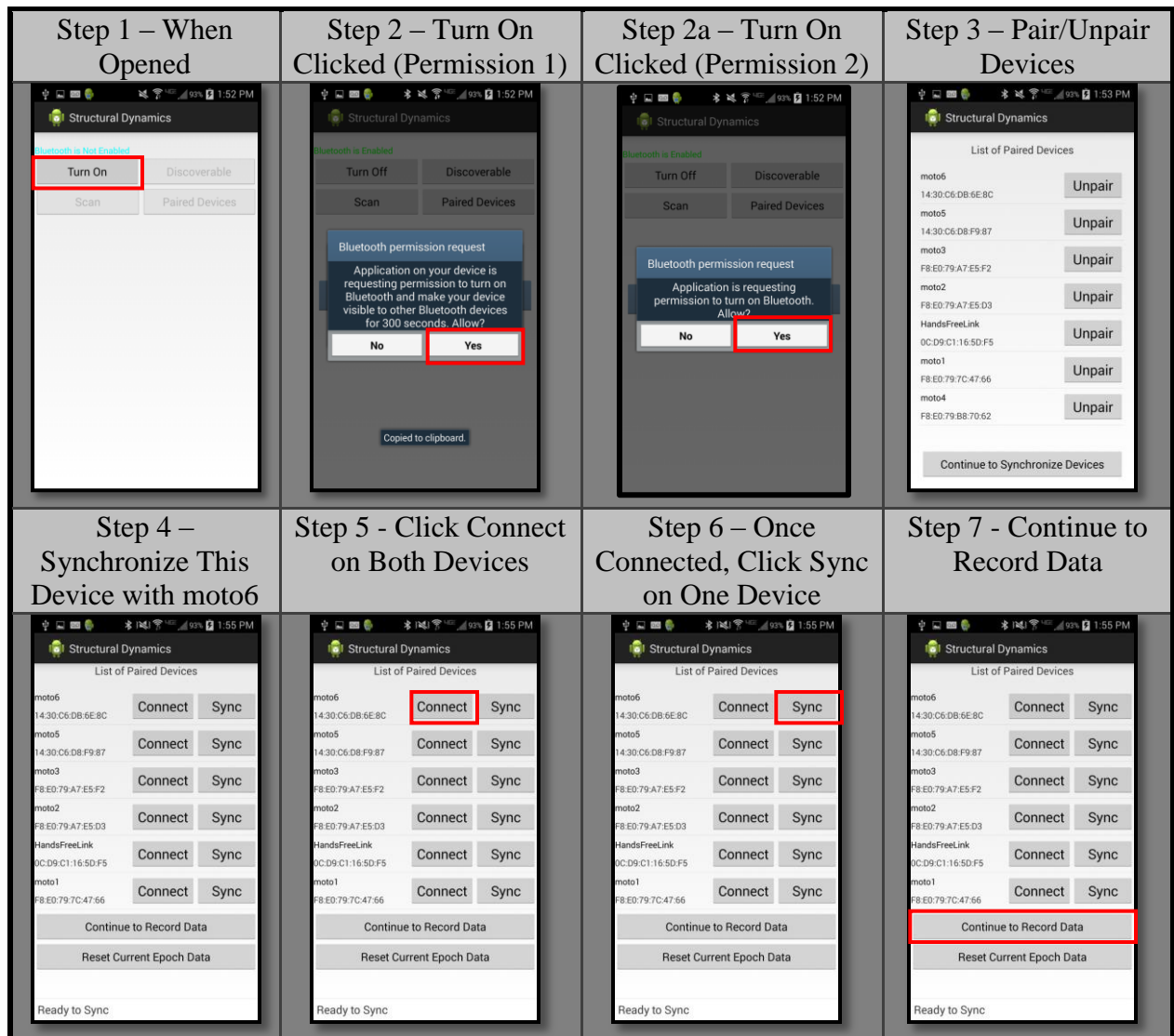


Figure 2-6 Record With Time Synchronization

The layout of the phone shown in Figure 2-6 (Step 1), is comprised of are four buttons located at the top of the screen. The top left button toggles Bluetooth on and off. When turning the Bluetooth on the application also asks the user’s permission for the phone to be discoverable by other devices via Bluetooth in Figure 2-6 (Step 2). By putting phones in the discoverable mode, it allows phones to be able to be paired if they are not already paired. In Bluetooth communication, two devices must be paired (similar to a handshake) before they can exchange data. Each device has to be paired to another individually; multiple devices cannot be

synchronized at one single time. For example, the user has to synchronize with the second device, then the third... etc.

If the Bluetooth on the device is already enabled, the “Discoverable” button at the top right of the screen is used to set the phone as discoverable. In the programming of the application, discoverable mode was set to last for five minutes, the longest time that Android allows. This gives the user the largest amount of time to scan for devices. The Java code can be seen in Figure 2-7.

```
public void visible(View view){
    bluetoothAdapter.enable();
    stateBluetooth.setText("Bluetooth is Enabled");
    Intent getVisible = new Intent(BluetoothAdapter.ACTION_REQUEST_DISCOVERABLE);
    getVisible.putExtra(BluetoothAdapter.EXTRA_DISCOVERABLE_DURATION, DISCOVER_DURATION);
    startActivityForResult(getVisible, REQUEST_BLU);
}
```

Figure 2-7 Java Code to Set Phone to be Discoverable

On the bottom left is the “Scan” button which enables the current phone to scan for other devices that are discoverable. As the phone finds new devices, it alerts the user by displaying a pop-up box with “Scanning...” with the option to cancel the scan Figure 2-8. Also, after a device is found, the phone notifies the user by displaying “Found: Device name”. This can be seen at the bottom of Figure 2-8. After the phone has completed its scan for new devices, the phone displays the same content that appears when the “Paired Devices” button is clicked (described below). In addition to paired devices, there is also a list of new devices that were found during the initial scan.

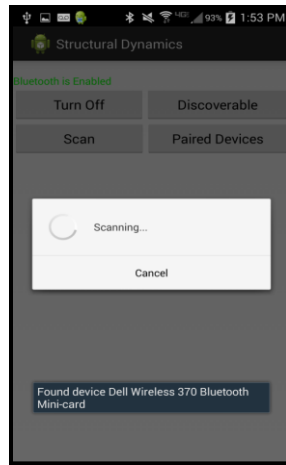


Figure 2-8 Bluetooth Scanning

The “Paired Devices” screen, displayed in Figure 2-6 (Step 3), lists all of the devices that have been previously paired along with their media access control (MAC) address, a unique 12 character identifier associated with each individual device, and a button for each device to the right. The button to the right will be labeled “Pair” if the devices are not paired or “Unpair” if the devices are already paired. When the “Pair” Button is clicked the devices will begin the process of pairing where one phone sends a request to the other, as long as both phones click yes, the phones will be paired and the “Pair” Button will change to “Unpair”. Once all of the devices the user would like to have paired are paired, the button “Continue to Synchronize Devices” will allow the user to sync the master phone’s epoch time to each paired device.

2.2.1 Time Synchronization

The “Time Synchronization” screen of the application, Figure 2-6 (Step 4), looks similar to the List of paired devices screen; however, the time synchronization screen has a “Connect” and “Sync” button for each device. This portion also contains a button “Continue to Record

Data” and a button to “Reset Current Epoch Data”. Each “Connect” button only works if it is clicked on both devices the user wishes to connect. Once the two devices are connected, the “Sync” button should be clicked on the master phone so the master phones can calibrate the time difference between itself and the connected node device and save to its own CSV file. The method used to calculate the time difference to send a message is described in the next section.

2.2.2 Synchronization Method

In an ideal scenario the epoch times for phones A and B would be identical at any point in time; however, in practice this is false. Another ideal assumption would be that it takes a negligible amount of time to send a message between two devices; however, in practice this is also false. To synchronize epoch times on two devices a process was developed to send time messages between phones, calculate the average time it takes to send a message and then calculate the difference in epoch times between the devices. Once the synchronization value is found, it is saved to the metadata of the file and must be included in the post processing procedure. Data collected using the synchronization method is not synchronized until their time stamps are changed during the post processing procedure.

To synchronize time between two phones, a series of messages will be sent between a master phone (Phone A in Figure 2-9) and a node phone (Phone B in Figure 2-9). Each message sent between devices contains the timestamps needed to communicate the average time to send a message, T_{sendmsg} . Figure 2-9 details the messaging process and the timestamps being recorded in the messages to determine the time it takes to send a message between devices.

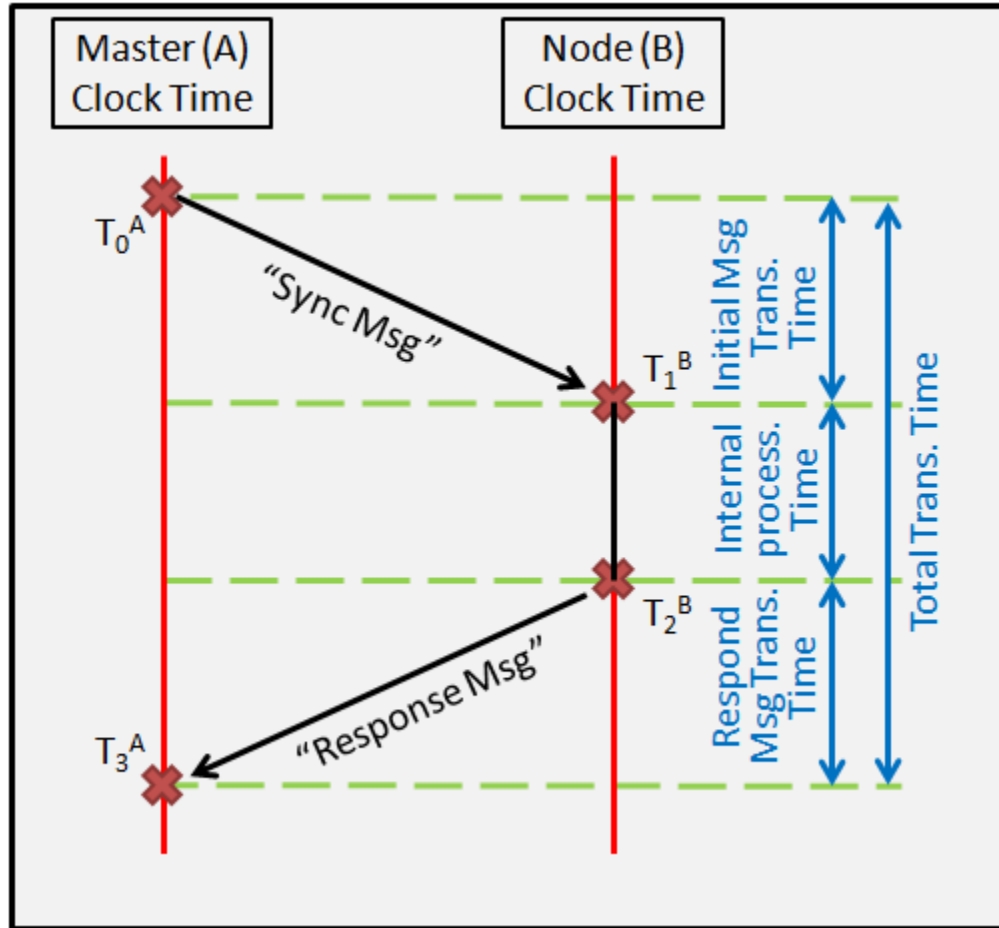


Figure 2-9 Time Protocol

In Figure 2-9, the timestamps shown are denoted in the following way:

- T_0^A = Time when the Master Device sends its first message (Msg1)
- T_1^B = Time when the Node Device receives the Masters first message (Msg2)
- T_2^B = Time when the Node sends the message (Msg2) back to the Master
- T_3^A = Time when the Master receives message (Msg3) from Node
- T_4^A = Time when the master sends its message (Msg3)

One complete transmission cycle contains four epoch times and is used to return an average time of transmission for the application. The first four times, one cycle, will be defined as the following four variables (T_0^A , T_1^B , T_2^B , T_3^A). T_0^A is the initial epoch time on Phone A. T_1^B is the epoch time on Phone B with it receives the initial time from Phone A. T_2^B is the epoch time

when Phone B send a message with T_1^B and T_2^B back to Phone A. T_3^A is the epoch time when phone a receives the message from Phone B. All four times are then added to a string on the master device (Figure 2-9) and used to calculate the transmission time between Phones A and B.

The process of sending and receiving messages between Phone A and Phone B is repeated 150 times. The 150 transmissions were chosen to minimize error in calculating time differences while also taking a reasonable amount of time to complete the calculation. In practice, sending 150 transmissions takes about five seconds to complete. After 150 transmissions, the first four timestamp values from the string ($T_0^A, T_1^B, T_2^B, T_3^A$) are plugged into Equation (1) to compute T_{sendmsg} on the master device.

Example Synchronization Times			
T_0^A	T_1^B	T_2^B	T_3^A
1416594200629	1416594231246	1416594231247	1416594200632

Table 2-1 Example Synchronization Times

$$T_{\text{sendmsg}} = \frac{(T_3^A - T_0^A) - (T_2^B - T_1^B)}{2} \quad (1)$$

Equation (1) returns the average time it took to send two messages, T_{sendmsg} ; one from Phone A to Phone B and the second, Phone B to Phone A. Subtracting the two epoch times associated with Phone B from each other, or $(T_2^B - T_1^B)$, removes the internal time it takes to process the code. Subtracting the two times associated with Phone A, or $(T_3^A - T_0^A)$, yields the total time round trip time for a single transmission. Dividing the total gives the average time it takes to send a message one way.

After calculating T_{sendmsg} between Phones A and B for the first transmission, T_{sendmsg} is calculated for the rest of the transmissions. Each T_{sendmsg} is averaged together to get the best representation of the time it takes to send a message from Phone A to Phone B.

After obtaining the average T_{sendmsg} , a second set of 150 transmission messages are then sent between the master and the node phones and another average T_{sendmsg} is computed based on these new 150 transmissions. This is to verify T_{sendmsg} by checking if the second set of transmissions will yield the same (within 1%) T_{sendmsg} . If the first set of transmissions is verified to have the same delay as the second set, the average T_{sendmsg} from the second set of transmissions is used to calculate the difference in epoch times between the master and the node phones.

$$T_{\text{difference}} = T_0^A - (T_1^B - T_{\text{SendMsg}}) \quad (2)$$

The code in Equation (2) shows how the application finds the time difference associated with set of transmissions. The equation displays the code to find the time on device B when T_0^A is sent. The first four times of the transmissions are separated and assigned to the variables “ T_0^A ”, “ T_1^B ”, “ T_2^B ”, “ T_3^A ”. The first two times are plugged into equation to find the true difference between the epoch times.

$$T_{\text{difference}} = (T_3^A - T_{\text{SendMsg}}) - T_2^B \quad (3)$$

The times T_2^B and T_3^A are both plugged into Equation (3) and determine the time difference on the return trip. Both values are then saved; the process is repeated for all data, the average of all the $T_{\text{difference}}$ values is taken, and that value is used to as the difference between the epoch times.

Using this method of synchronization eliminates the internal time it takes to exchange the packet from receiving the sending. Neglecting the internal processing time positively determines that other applications running and will not interfere with the synchronization processes.

After confirming both sets of transmissions have the same $T_{\text{difference}}$ between the master and node device, the $T_{\text{difference}}$ is saved to the master device. The time difference variable is saved to the metadata of the CSV file created in the “Record Data” portion of the application.

2.3 Combine CSV Files

Combine CSV Files portion of the application was developed to combine multiple data sets into one with one timestamp, and then to animate a multi degree of freedom system on the phone. Including this in the application make visually representing collected data possible and also makes post processing easier by combining data into one file.

Upon clicking the “Combine CSV Files” button, the user is prompted by choosing the amount of files they would like to combine, Figure 2-10 (Step 2). Once the user selects the amount of files they would like to combine, they are brought to a screen with a line for each file the user would like to combine and a line labeled “Save As:” which allows the user to name the new file. The bottom of the Combine Files screen has a list of files available to be combined, Figure 2-10 (Step 3). If one of the files does not exist in the folder, the phone will alert to the user with a notification.

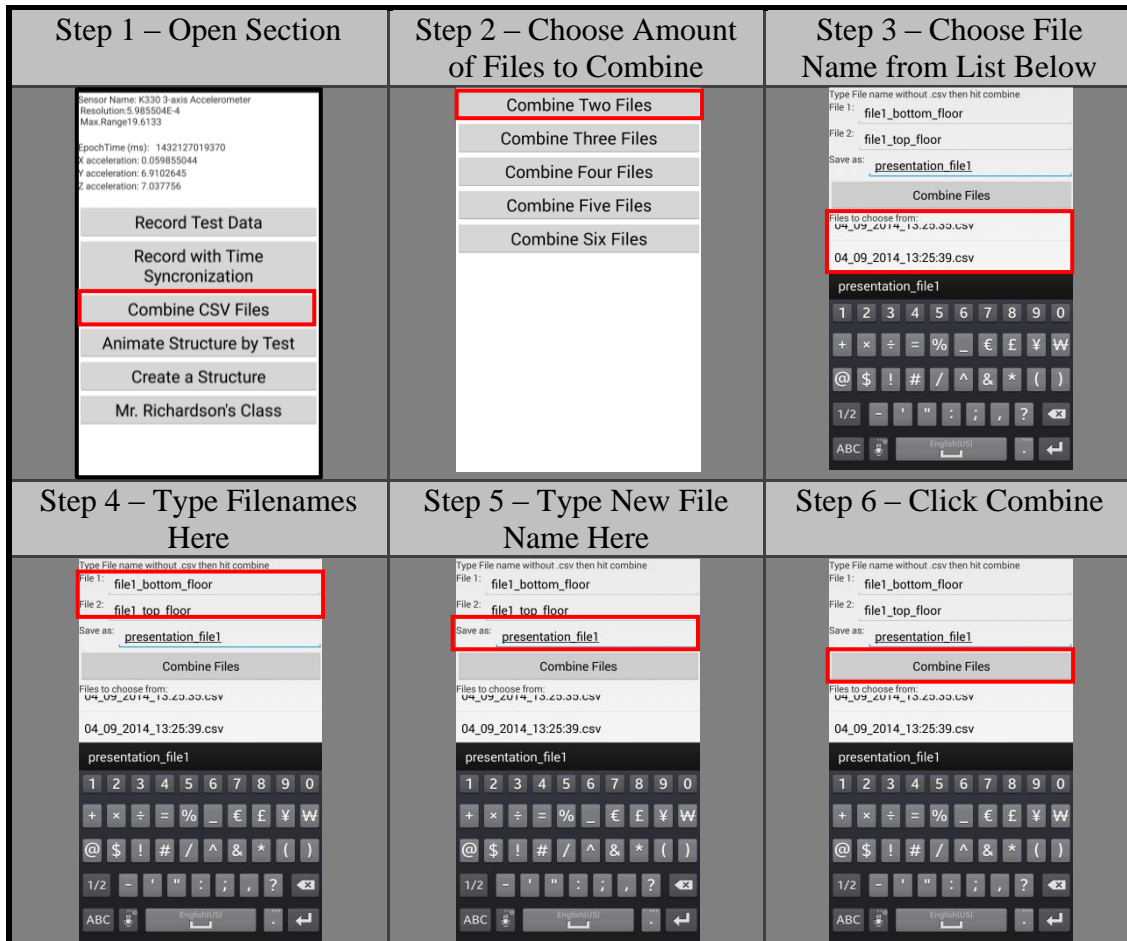


Figure 2-10 Combine CSV Files

One shortcoming of the application is its inability to send files from one phone to another. In the case where multiple devices are used to record data from different stories of a structure, the native Bluetooth file transfer capabilities in the Smartphone need to be used. Another method that can be used to get all the files onto one device is by connecting the devices to a computer and transferring the files from phone to computer to master phone. Regardless of transfer method, the files need to be transferred to the “Accelerometer App” folder in the Smartphone for the application to find the file.

2.3.1 Interpolating Data

The method for interpolating data from the epoch timestamp to a set time stamp was developed for MATLAB, and transposed to Java. The script allows for an n degree-of-freedom (DOF) or any n-story structure for post processing. It is understood there are other methods of interpolation out there, however, to make this application and method independent, a linear interpolation code was written. To describe how the interpolation process occurs, the MATLAB code will be discussed because it is much more condensed than the Java code.

One major issue with using this interpolation method is changing the data from the raw data to a new set of data. Since android does not allow for the user to record data on a set time interval, linear interpolation is used and can introduce error into the data. Even with the error added from interpolating, stiffness values recorded from the experimental structural testing are comparable to a dedicated sensing system (discussed later).

The user first enters all the files they would like to combine, the script looks up each file and takes all the data contained in the CSV file and sets it to a table with the columns time, x -, y -, z -accelerations. To make the process easier to describe, two files will be used this example. File A's data (Master file) will be called "m1", and File B's data will be called "m2". The script then looks into "m1", finds its metadata (Figure 2-2) to obtain the vector for time synchronizing. The "Time Synchronizing Vector" contains all the time synchronization values for the collected data. As discussed in the Synchronization Method section of the thesis, the value is used to transpose the data to be synchronized.

With the acceleration data and time synchronization values loaded into the application, the script begins to synchronize the data. For "m1", the time array (first column in "m1") is

separated from the accelerations and the first time is subtracted from the vector. The purpose of this is to bring the vector back to zero. The vector is then divided by 1000 to convert from milliseconds into seconds (Figure 2-11). The new time vector is labeled “t1”:

```
t1 = (m1(:,1)-m1(1,1))/1000; % time difference in seconds
t2 = (m2(:,1)-m2(1,1))/1000;
```

Figure 2-11 Time Difference

In this figure,

- m1 = data contained in the CSV file specified as FileA
- m2 = data contained in the CSV file specified as FileB
- (:,1) = every value in the first column.
- (1,1) = the first row and first column of the data set
- t1= new time stamp created from m1’s times
- t2= new time stamp created from m2’s times

The script then brings in “t2” as shown in Figure 2-12, adds the time synchronization value corresponding to “m2”, and then adds the difference between the first two times in “m1” and “m2”. The first time in “m1” and first time in “m2” yield the difference between the epoch times and the “timeSync” value synchronizes the time difference:

```
% lines up data
t2 = t2 + timeSync(1,2) + (m2(1,1)-m1(1,1))/1000;
```

Figure 2-12 Line Up Data

where:

- timeSync = vector of time differences from the master device to node device
- (1,2) = the first row and second column in the array

The new time stamp is then brought back to the acceleration data, and set to a new variable, data1:

```
% [time(seconds), x(m/s2), y(m/s2), z(m/s2)]  
data1 = [t1, m1(:,2), m1(:,3), m1(:,4)];  
data2 = [t2, m2(:,2), m2(:,3), m2(:,4)];
```

Figure 2-13 Assemble Time and Accelerations

Next, the data sets are cut down to find the overlapping area. The minimum of t1 and t2 is taken, and the maximum of those values is used as the start time for the time interval. Also, the maximum of each data set is taken and the minimum value is used as the end time of each data set. The start time and end time are used to create the “Overlap Zone” depicted in Figure 2-14. This figure has 6 different phones data to depict how the overlap zone would be found with more than 2 devices.

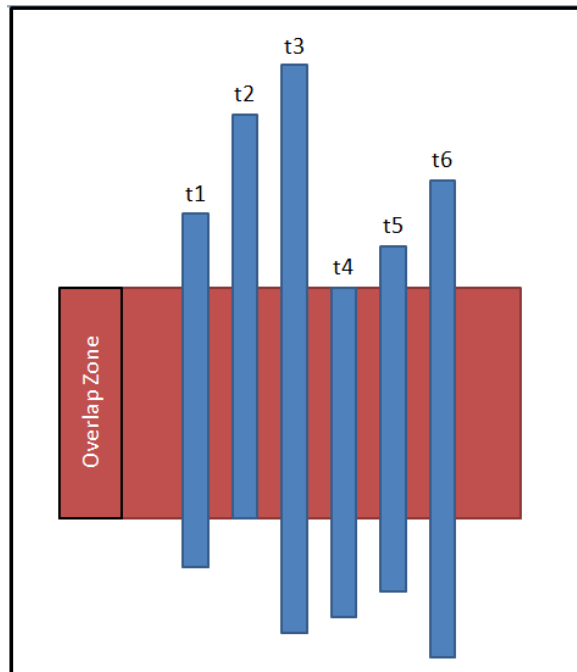


Figure 2-14 Overlap Section

Once the application has created the data sets to be the same length as each other, it then interpolates each data set to the new, monotonically increasing timestamp with a delta t specified by the user, SetT. In the following example, SetT is at a sampling rate of 100 Hz.

SetT	t1	Y-Acceleration
0 T_{interp}	0 T_{before}	0.565 y_{before}
0.1	0.07	0.777
0.2	0.13	0.917
0.3	0.19 T_{after}	1.178 y_{after}
	0.24	1.105
	0.3	0.878

Figure 2-15 Sample Data

Time T_{interp} is taken from the “SetT” vector, and the application searches through t1’s vector (time vector from Phone 1) for T_{before} that is the timestamp before T_{interp} and for T_{after} that is the timestamp after T_{interp} . The acceleration values, y_{before} and y_{after} associated with T_{before} and T_{after} , respectively, with are used to estimate y_{interp} using this following equation:

$$y_{interp} = \frac{y_{after} - y_{before}}{T_{after} - T_{before}} (T_{interp} - T_{before}) + y_{before} \quad (4)$$

Figure 2-16 shows a portion of an example set of data that could have been recorded from the accelerometer. The numbers were simplified for the ease of demonstration. The vertical green line demonstrates the time being interpolated.

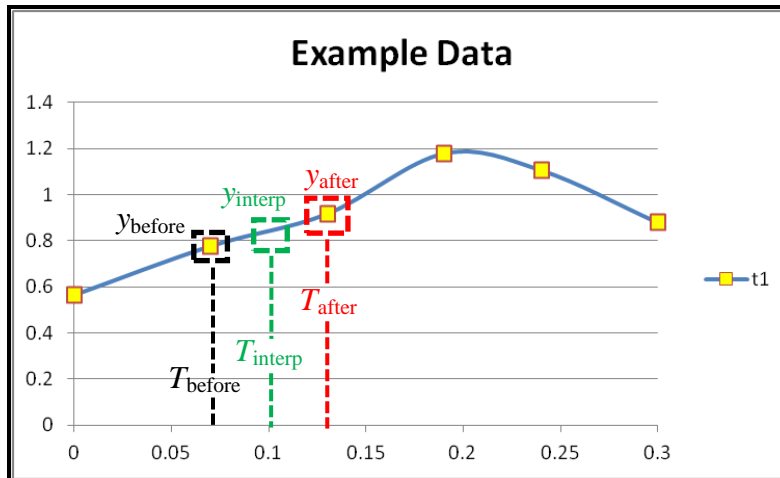


Figure 2-16 Sample Data Graph

The interpolated acceleration that would be associated with time T_{interp} (0.1) from SetT is 0.847 m/s^2 . After the new acceleration is calculated at its new timestamp, every other time in the SetT is used until SetT has a new Y-Acceleration for each time in its vector. Once file1 is done, it completes the same interpolating data function for every other file the user specified. At this point the data can be analyzed.

2.4 Animate Structure By Test

During an educational outreach conducted at the Hillside Middle School in Manchester, New Hampshire, an experiment on a student-built structure was conducted to illustrate a simple test to the middle school students. The “Animate Structure By Test” portion of the application was able to take the acceleration data recorded during the test and animate the structure’s motions on screen for the students to observe. For the purpose of the outreach, the code was written for only a two story structure with the first floor representing the ground floor and the second floor representing the structure, Figure 2-17.

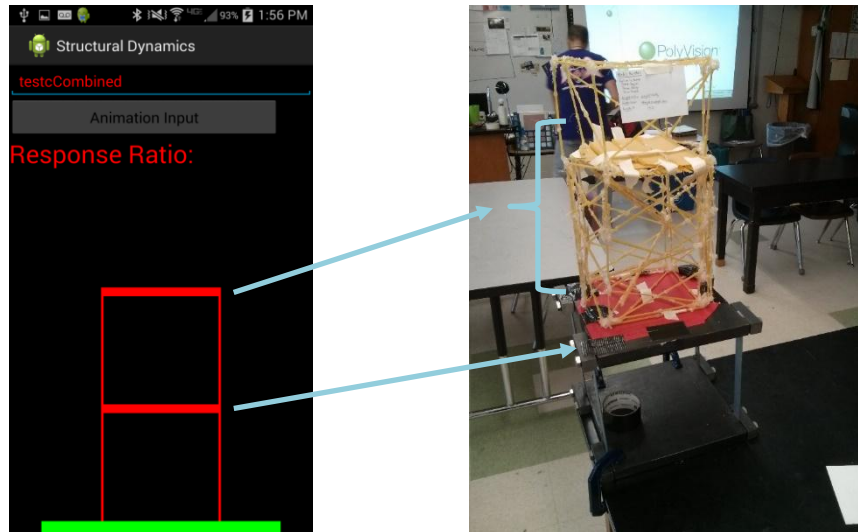


Figure 2-17 Animate Structure

The screen that the students observed is illustrated in Figure 2-17 (Left) and contains a line to enter the file name that was put together in the “Combine CSV Files” portion of the application. Below the file line is a button to animate the structure below by the file entered on the line above. After the structure is animated by the phone, the phone calculates the response ratio of the structure. The response ratio is given as:

$$\frac{\sqrt{\frac{1}{n} \sum_{i=1}^n y_{i,\text{structure}}^2}}{\sqrt{\frac{1}{m} \sum_{j=1}^m y_{j,\text{ground}}^2}} \quad (5)$$

where:

- n = number of time steps in $y_{i,\text{structure}}$
- m = number of time steps in $y_{j,\text{ground}}$
- $y_{i,\text{structure}}$ = the y-axis acceleration at the i -th time step of the structure
- $y_{j,\text{ground}}$ = the y-axis acceleration at the j -th time step of the ground

This value would allow students to compare individual tests that did not have the same magnitude of excitation input in outreach settings (e.g., hand shaking a “shake table” in a classroom at Hillside Middle School).

To animate the structure, the “Combine CSV Files” portion of the application was used. Once the acceleration data of the test structure and shake table were all in the same file, the smartphone used the following method to animate the structure. The smartphone read the CSV file entered by the user, line by line. For every line of the CSV file that was read, the phone read the y-acceleration (the direction of the shaking motion) of each story, and created a new frame with the structure’s stories displaced by the acceleration amount in the CSV file.

2.5 Create A Structure

When the user enters the “Create A Structure” portion of the application, it presents them with the option of choosing a structure that has 1 to 6 stories (Figure 2-18).

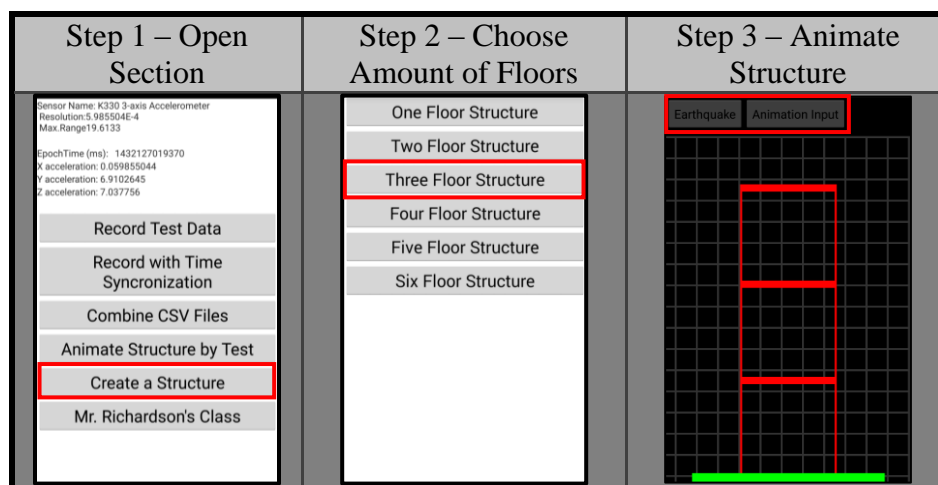


Figure 2-18 Create a Structure

The “Create A Structure” portion of the application allows for the user to (1) create a structure by choosing the number of floors in the structure, and (2) interact with the structure by two excitation input methods: impulses and earthquake. This portion of the application was built in eclipse using OpenGL. OpenGL is a library in java used for 2D or 3D visual programming. In OpenGL, each shape starts out being coded as a matrix that is then translated to a triangle. Triangles are then combined to create new shapes. For example, to draw the face of the column in a structure, the rectangle needs to be split into two triangles, visually represented by the blue and red triangles in Figure 2-19. Each triangle is drawn by giving the code the vertices in a counterclockwise manner.

The structure is formed floor by floor, separated into 4 columns and 1 floor. Each column or floor of the structure is created from 12 triangles put together to form a single cube. The columns and floors are created at a certain height, width, and depth that are set in to fit the structure to the screen. Coding the application in this manner allows it to be used on any Android device regardless of screen size or pixel count.

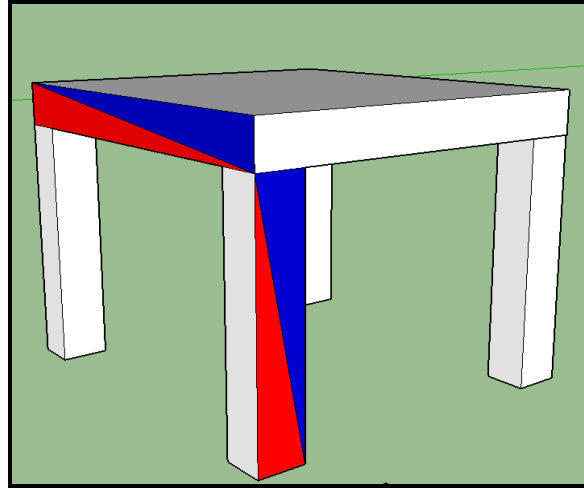


Figure 2-19 Drawing Structure

2.5.1 Camera Angle

Currently the camera angle is set so the entire structure will be visible on any sized screen (with font size on phone set to normal). The phone takes in the width and height of the screen, and creates the display relative to the size of the screen. In the future, the camera angle, and position could be changed by the user, however the current code does not allow for the user to change these preferences. If the code were to be altered, the structure can appear in a 3-dimensional orthogonal view, which allows all lines to be parallel to one another.

2.5.2 Displace the Structure

When the user displaces a story of the structure to the right (or left), it is equivalent to the structure being impacted by an impulse force from the left (or right) on that story. The reference point starts at the edge of the screen and uses a ratio from the center of the screen to the structure. For example, if the user chooses to displace the structure story to the far right edge of

the screen, it would be associated with a displacement factor of one. If the user chooses to displace the structure story exactly between the center and the edge of the screen, the displacement factor would be 0.5. If the user chooses not to displace the story (i.e., the story remains in the center of the screen), the displacement factor would be equal to zero, which would cause no displacement. Figure 2-20 shows how the user would displace multiple stories of the structure. In this case, the displacement factors of Stories one and two would be equal to one since they are displaced to the far right edge of the screen. The displacements would act as two point loads being applied to the right at both the first and second stories.

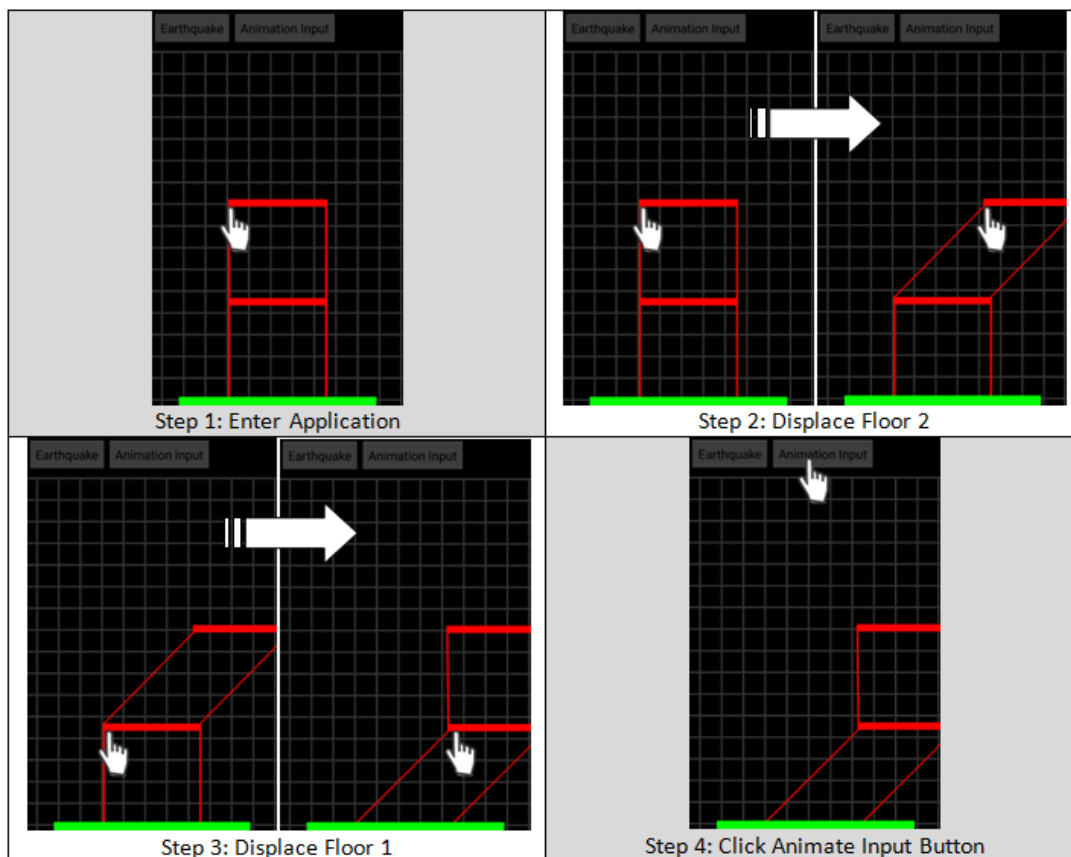


Figure 2-20 Displace the Structure

Displacing the structure and then clicking earthquake will not apply an earthquake with this displacement. Instead, the structure will get reset to zero and then the earthquake motion will be applied.

Chapter 3 Testing

Some of the current experimental methods of structural analysis at the University of New Hampshire (UNH) include strain gauges, Microstrain wireless accelerometers, and wired accelerometers. The smartphone application was developed for schools similar to UNH but also schools without the necessary funds who might be interested in structural analysis in an experimental setup.

Tests were conducted on the smartphone application to determine if it would be possible to obtain comparable data to current methods of structural analysis. Testing conducted includes; time synchronization testing to determine if the method works correctly, airplane mode testing to insure the phones do not update their clock with the network, and structural analysis to compare the stiffness values obtained from the smartphones and compare to a current accepted method.

3.1 Model Structural Information

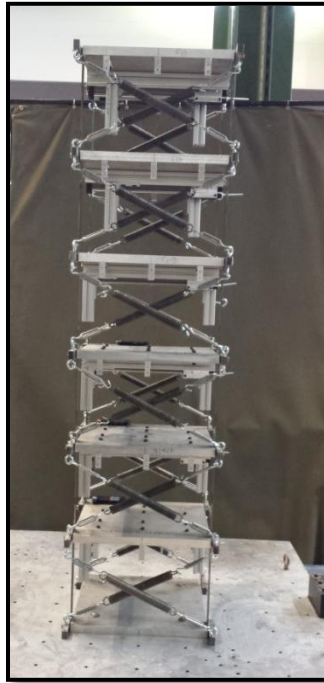


Figure 3-1 Experimental Structure

The experimental structure, as shown in Figure 3-1, is a six-story shear structure mounted on the shake table at the University of New Hampshire. The structure is a metal frame structure that consists four steel columns with rectangle cross sections and each floor is formed mainly by an aluminum plate. Floor plates are held by friction by four extra steel stocks at four corners to keep the plate perpendicular to the column. Each story has identical height of 12 inches and identical weight of 44.7 lb.

A single cross bracing to increase inter-story stiffness consists of one spring, and two turn buckles (Figure 3-2). The stiffness of each spring is 13.41 lb/inch and the mass of all three components, two turn buckles and one spring is 1.07 lb. When a story is fully braced, the four cross bracings add a total of 4.28 lb.



Figure 3-2 Spring and Turnbuckles (Cross Bracing)

The six-story shear structure can be simplified as a lumped mass shear structure (Chopra, 2012) as shown in Figure 3-3, the fundamental period is designed to be 0.6 seconds to represent a six-story steel frame structure.

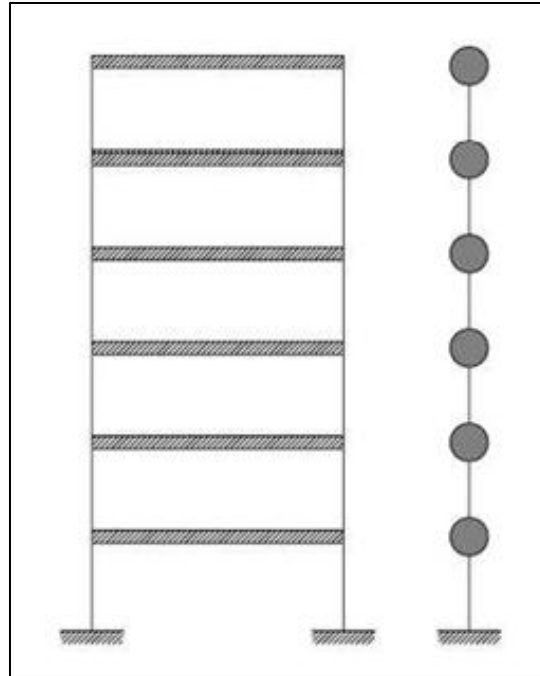


Figure 3-3 Lumped Shear Structure

3.2 Smartphone Setup

An important aspect of the testing procedure involved securely attaching the smartphones to the structure in a simple yet effective method that can be used in middle/high school settings. The adhesive option chosen and used throughout the testing was a sticky pad typically used in cars. Through testing, Sticky pads provided the necessary friction to insure the smartphone does not move and also allowed for a cheap, quick and simple setup. The method was chosen to be easily attainable for an experimental setup in a lab setting for the middle school to college curriculum. As seen in Figure 3-4 the sticky pad is applied to the structures surface, and the phone is then placed on top of the sticky pad. A key for this method to work effectively is to correctly align the smartphones to the structure (such that the y-direction of the phone is parallel to the direction of the structure's vibrations). The smartphone was guided onto the table so it would be aligned with the edge as parallel as possible to the edge of the structure. A flat object,

square to the structure, was used to guide the smartphone into place. Two examples of an object that were used were a piece of wood, and the edge of a hardcover book (both materials commonly available in a school environment).

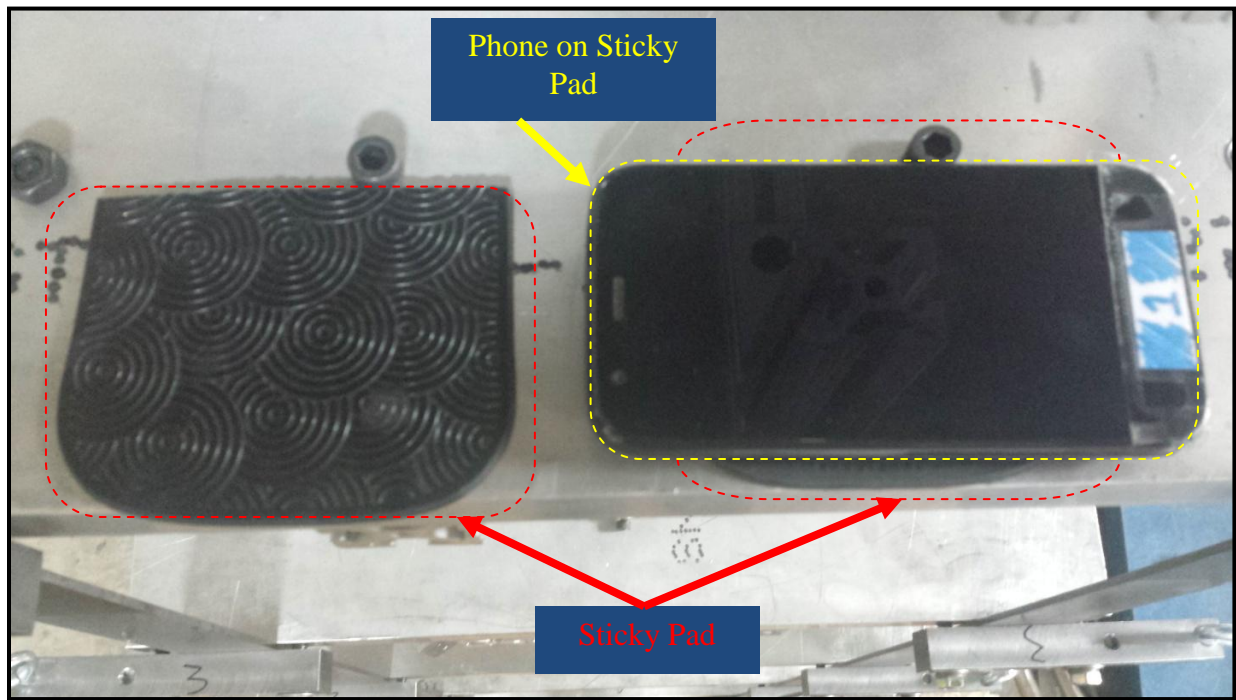


Figure 3-4 Phone on Sticky Pad

To maintain adhesiveness of the sticky pads, they would need to be washed and air dried prior to testing. Dust is easily attached to the sticky pad and, thus, decreases the sticky pads' adhesiveness. Also, it is advised to wipe the surface of the structure floors down to minimize the quantity of dust on the surface. It is recommended this method only be used in situations of single axis vibration. Testing was only conducted in one axis, cannot definitively confirm it would hold up in the x, or z axis of vibration, however, it is probable.

3.3 Testing configurations

Testing was conducted on the experimental structure to assess the smartphone sensing network's ability to detect damage on a structure. The six testing configurations, as shown in Figure 3-5, were (1) healthy, (2) 6th floor damaged, (3) 4th floor damaged, (4) 1st floor damaged, (5) 1st and 4th floors damaged, and (6) 4th and 6th floors damaged. Configurations were chosen to have data for both a damaged floor and when the floor is considered healthy.







Configuration 1: Healthy Structure	Configuration 2: 6th Floor Damaged	Configuration 3: 4th Floor Damaged																		
 <table border="1" data-bbox="540 359 599 959"> <tr><td>6</td></tr> <tr><td>5</td></tr> <tr><td>4</td></tr> <tr><td>3</td></tr> <tr><td>2</td></tr> <tr><td>1</td></tr> </table>	6	5	4	3	2	1	 <table border="1" data-bbox="932 359 990 959"> <tr><td>6</td></tr> <tr><td>5</td></tr> <tr><td>4</td></tr> <tr><td>3</td></tr> <tr><td>2</td></tr> <tr><td>1</td></tr> </table>	6	5	4	3	2	1	 <table border="1" data-bbox="1300 359 1359 959"> <tr><td>6</td></tr> <tr><td>5</td></tr> <tr><td>4</td></tr> <tr><td>3</td></tr> <tr><td>2</td></tr> <tr><td>1</td></tr> </table>	6	5	4	3	2	1
6																				
5																				
4																				
3																				
2																				
1																				
6																				
5																				
4																				
3																				
2																				
1																				
6																				
5																				
4																				
3																				
2																				
1																				
Configuration 4: 1st Floor Damaged	Configuration 5: 1st & 4th Floors Damaged	Configuration 6: 6th & 4th Floors Damaged																		
 <table border="1" data-bbox="540 1127 599 1728"> <tr><td>6</td></tr> <tr><td>5</td></tr> <tr><td>4</td></tr> <tr><td>3</td></tr> <tr><td>2</td></tr> <tr><td>1</td></tr> </table>	6	5	4	3	2	1	 <table border="1" data-bbox="932 1127 990 1728"> <tr><td>6</td></tr> <tr><td>5</td></tr> <tr><td>4</td></tr> <tr><td>3</td></tr> <tr><td>2</td></tr> <tr><td>1</td></tr> </table>	6	5	4	3	2	1	 <table border="1" data-bbox="1300 1127 1359 1728"> <tr><td>6</td></tr> <tr><td>5</td></tr> <tr><td>4</td></tr> <tr><td>3</td></tr> <tr><td>2</td></tr> <tr><td>1</td></tr> </table>	6	5	4	3	2	1
6																				
5																				
4																				
3																				
2																				
1																				
6																				
5																				
4																				
3																				
2																				
1																				
6																				
5																				
4																				
3																				
2																				
1																				

Figure 3-5 Damage Configurations (Shaded Floors Are Damaged)

3.4 Damaged Floors

When the structure was created, there were hooks added so springs and turnbuckles could be attached to the structure to increase the inter-story stiffness. This method was developed to temporarily damage the structure, and could be replicated in other experimental structural models.

When the springs were removed to change the inter-story stiffness, they were taped to the floor from which they were removed (Figure 3-6) to maintain a constant mass for that specific floor. If the springs mass was not added back onto the structures floor, it would have altered the stiffness estimation. Damage was applied to the structure by removing one floors cross bracing, therefore decreasing the stiffness of the floor.

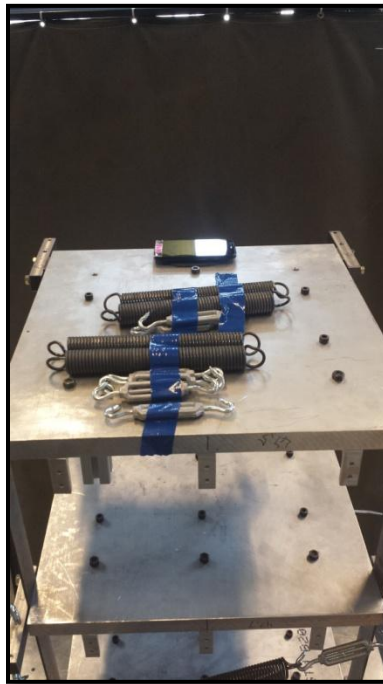


Figure 3-6 Damaged Floor

3.5 Post Processing Procedure

After the devices collect the accelerations, the files are loaded to a computer, and manipulated by the process described in the Section 2.3.1 Interpolating Data. To reiterate, the MATLAB script would take in the data from each device, time synchronize using the synchronization value, find overlapping area of data, interpolate to one time stamp, and sets up the data to be plugged into the Eigen-system Realization Algorithm (ERA) (see Appendix A). The ERA code is used to estimate the mode shape and frequency of each mode of the structure. The estimated modal parameters are then used (via a least square method as detailed in Appendix B) to determine the stiffness values of the structure.

3.6 Preliminary Time Synchronization Testing

The time-synchronization method discussed in Section 2.2.1 “Time Synchronization” was used so the phones could successfully communicate their differences in epoch times. Time synchronization on the phones used a method of sending each phone’s epoch times back and forth and compiling those times to calculate the difference in epoch time between the phones.

The method was created to remove any error due to differences in the devices’ internal clocks. At any given moment, the devices’ clocks can vary based on when they last communicated with the server. Since this method was created just for this purpose, testing was done to verify that the method was accurately determining the difference in epoch times.

A simple test was completed on the test structure, by placing all six phones on the same floor; displacing the structure and allowing the structure to vibrate. For the test, the devices were placed on the 4th floor of the structure which allowed the easiest access during testing. The

magnitude of displacement and the floor level of the original displacement did not need to be recorded since the ability to line up data was the only variable being tested and the values would not be compared between tests.

3.6.1 Initial Time Synchronization Testing Results

While looking at the initial data received, some of the phones were not syncing correctly. In Figure 3-7, the phones labeled “Moto 6” and “Moto3” are not synchronized with the other phones. The source of error for these tests was determined to be associated with the network clock updating periodically and network clocks not being accurate down to the millisecond. This means the phones would update the local epoch time on the device to be as close to the network clock. Although this would seem advantageous to maintaining the same time between devices, the network does not synchronize every device to the exact time with enough precision needed for the data to work. The six phones can be as far as 10 seconds apart as seen in some testing results.

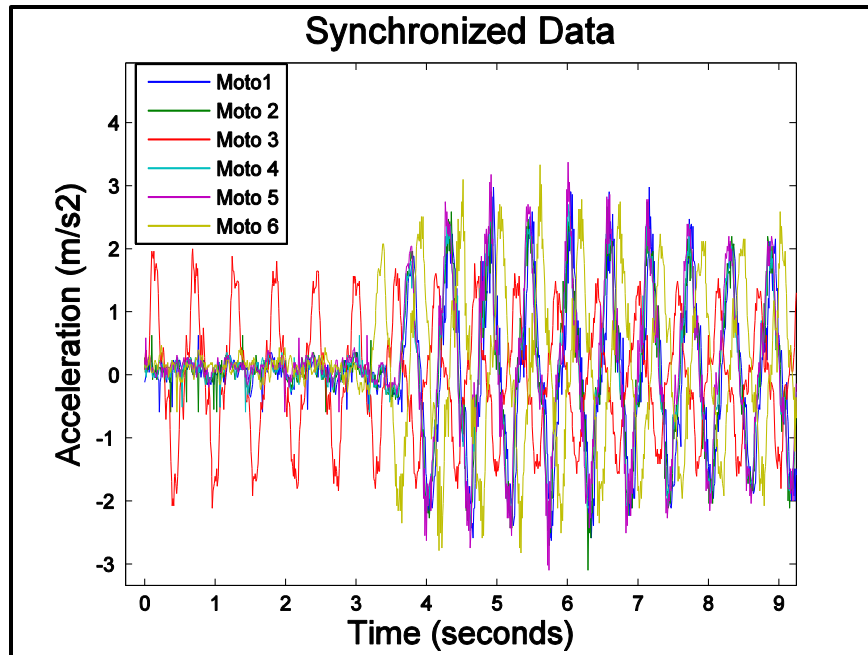


Figure 3-7 Time Synchronization Error

3.7 Airplane Mode Testing

To solve the issue of phones updating to the network time while testing, the phones were put into airplane mode. Airplane mode is used on the device to disconnect from wireless signals which in this case is the network. Taking away the phones ability to sync with the network would allow for the phones to have more reliable time synchronizations. Theoretically, if the devices are not capable of synchronizing with the network time they will maintain a more accurate difference between devices. To correct for the difference in time between devices, the phones calculate a synchronization value before testing. The process of synchronizing the devices epoch time was discussed in the section “Record with Time Synchronization”. When airplane mode was not enabled on the device, the synchronization method becomes less accurate because the phones will update with the network time. Updating with the network changes the local time significantly, rendering the time synchronization method useless.

To confirm airplane mode was working, and the devices are maintaining similar synchronization values, multiple synchronizations were run in succession and the master file saved to the device. With the master files on the phone, the time synchronization values were recorded and graphed. The purpose of this testing was to see how long the obtained synchronization values would remain the same. The difference should be similar (not exceeding .1 seconds) for multiple tests in succession to prove the devices are not varying off of the initial value and that the time synchronization method used in the application is reliable.

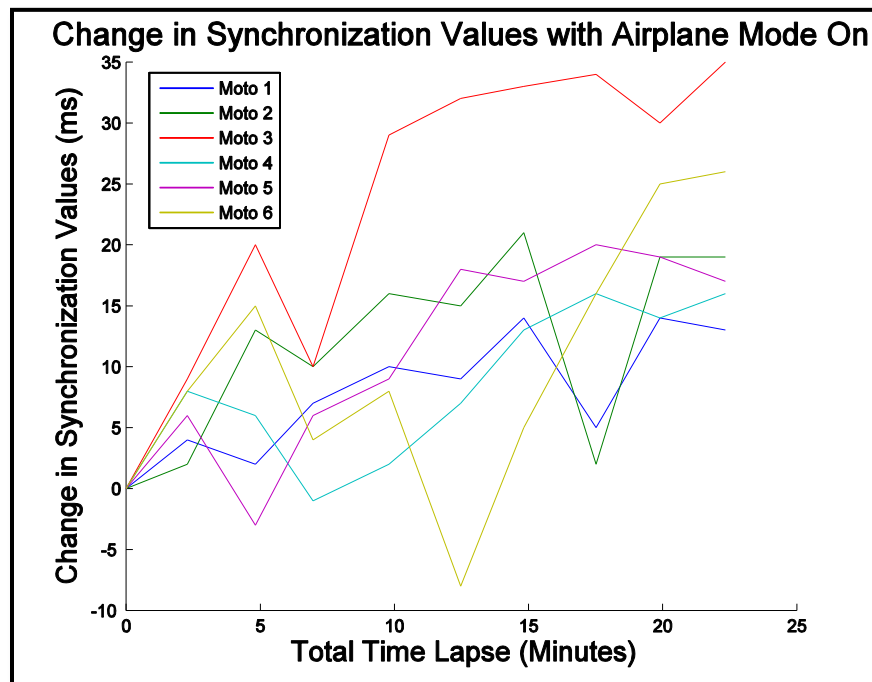


Figure 3-8 Synchronization Value Change with Airplane Mode On

In Figure 3-8, time synchronization was conducted with airplane mode enabled. Each line shows the difference between the initial time synchronization value and value for the present test over time. Obtaining a synchronization value was discussed in Section 2.2.1 “Time

Synchronization” of the application and is the difference in epoch time (ms) between the node device and master device.

Further testing was conducted to help determine how frequently the devices reconnected to the network time and to solidify the importance of airplane mode. Figure 3-9 was a test conducted in the same manner as Figure 3-8, however, Figure 3-9 had airplane mode off. It showed two inconsistencies in “Moto 4” and “Moto 5” that prove why the phones need to be in airplane mode during testing. With airplane mode disabled, only about 7 minutes passed before one of the phones updated with the network and significantly changed (More than 0.1 seconds) its epoch time. While airplane mode was enabled the devices epoch times in Figure 3-8 did not vary by more than .04 seconds.

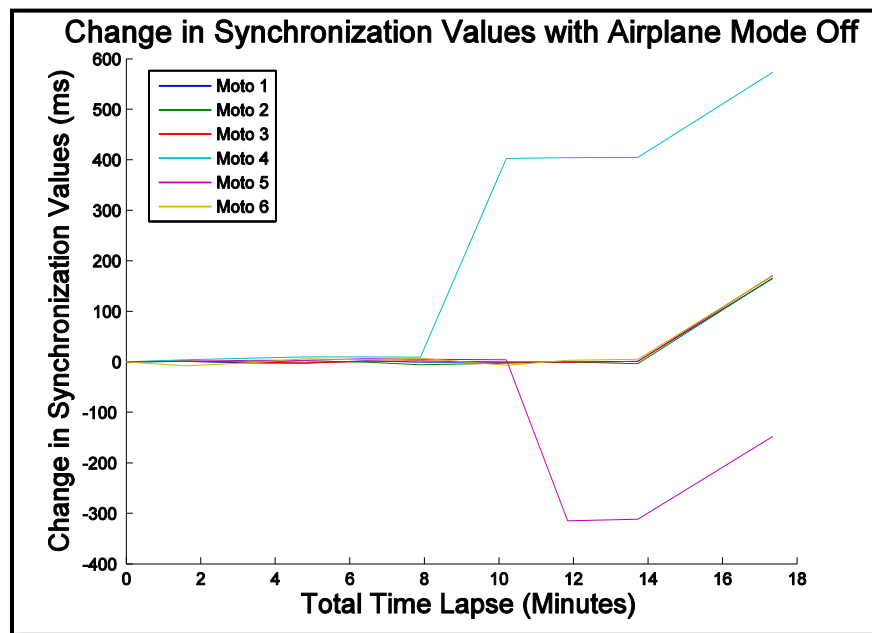


Figure 3-9 Time Delay Test 5 Airplane Mode Off

The purpose of presenting the data in this way was to visually see how the time synchronization value would diverge from the original synchronization value. Figure 3-8 and Figure 3-9 confirmed activating airplane mode would keep the time synchronization value more

constant than without airplane mode over the course of testing and provided more consistent values. Consistent values with airplane mode enabled proved the devices would not reconnect to the network while collecting data which had caused errors in the past. Seeing a minimal change in time values between the two devices provided confidence the data recorded from the devices, combined with the synchronization values would result in quality data when used in structural testing.

It is recommended that users should synchronize the phones as often as possible to insure accurate time synchronizations as phones' internal clocks diverge more over time. Ideally, the user will synchronize between every test to obtain the most accurate synchronization difference. However, Figure 3-10 shows the phones did not begin to significantly diverge from the first synchronization value until 60 minutes after the initial synchronization. Values from Figure 3-10 were taken from structural analysis testing discussed in the next section.

In the SHM experiments, devices were synchronized about every ten minutes from the initial synchronization to minimize the chance of the devices' times diverging. Airplane mode was activated on all the phones while tests were conducted.

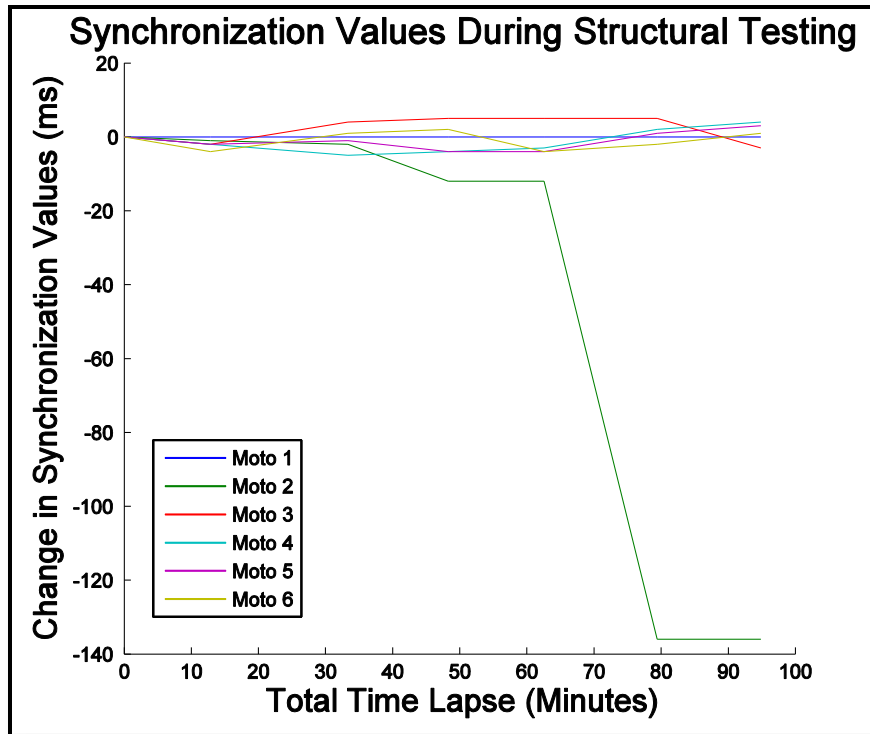


Figure 3-10 Synchronization Values During Structure Testing

3.7.1 Time Synchronization Testing with Airplane Mode

After the airplane mode was proved to work and the guidelines for testing were put into place, the next step was to prove the synchronization method correctly aligned the accelerations of six different sensors.

For the tests below, all six phones were placed on the same floor of a 6-story structure to ensure the same vibrations would be seen by the sensors. Between each test, the phones would synchronize with the master phone and the test was repeated. Devices were synchronized between tests to analyze how the synchronization feature was working on the application. The structure was excited and the phones recorded the accelerations. Seven tests were conducted with the phones with the following results persisting throughout the testing.

In Figure 3-11 and Figure 3-12, the improvement by the developed method of time synchronization was evident. Without the time synchronization (Figure 3-11), acceleration data from multiple devices do not line up properly when only using the network clock because internal smartphone clocks do not have a precise time down to the millisecond that is required in structural testing.

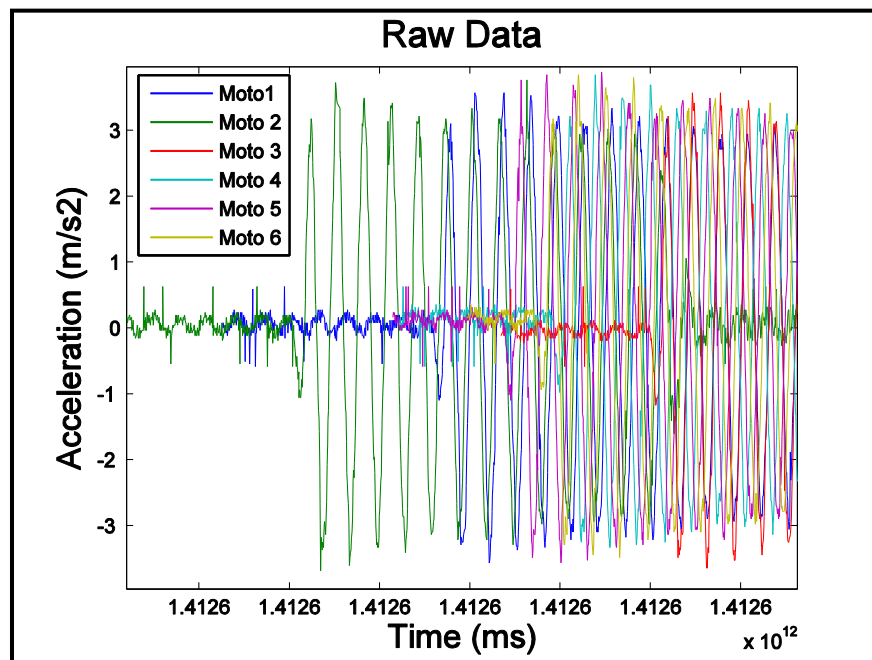


Figure 3-11 Raw Data without Time Synchronization

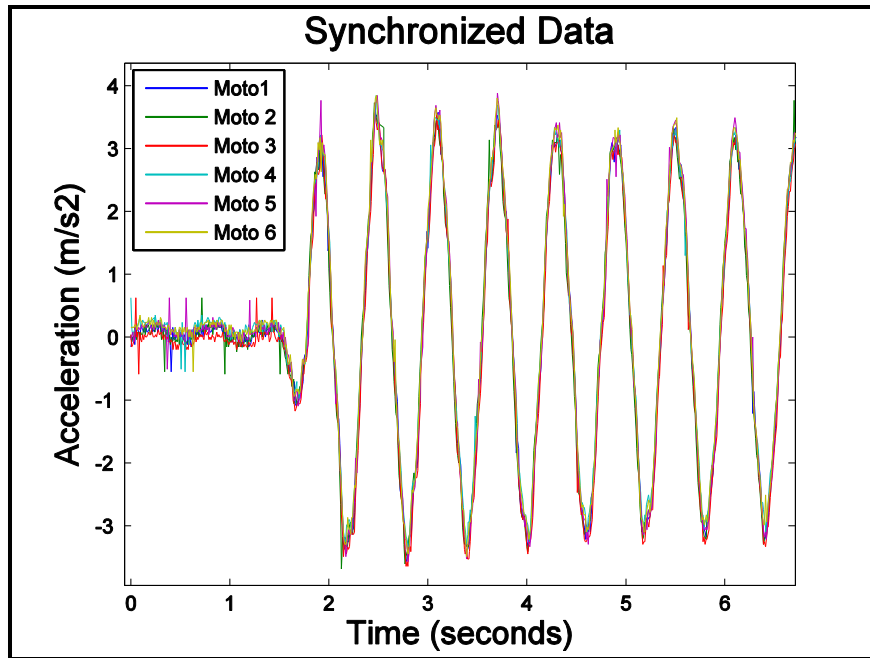


Figure 3-12 Synchronized Data

Figure 3-12 shows all six phones' data aligned directly over each other due to the synchronization method available on the device. Data is taken from the phones, and using the synchronization value, are aligned over each other. With these results, and the other test results that are not shown, it was determined that the time synchronization feature of the application works properly and could be used in a lab experiment SHM situation.

3.8 Structural Analysis Results

To test how the smartphones could handle a SHM situation, data was collected, and post processed using methods described above. From the post processing methods, stiffness values were obtained for each individual floor. To account for experiment variations, multiple tests

were run for each configuration. An example of the four tests for one configuration can be seen in Table 3-1.

Configuration 2: 6 th Floor Damaged (lb/inch)					
	Test 1	Test 2	Test 3	Test 4	Average
Floor 1	246	228	233	240	237
Floor 2	263	270	267	265	266
Floor 3	263	262	263	268	264
Floor 4	246	258	249	253	252
Floor 5	285	275	280	280	280
Floor 6	252	254	253	252	253

Table 3-1 Configuration 2: 6th Floor Damaged Results

Table 3-1 shows four data sets recorded from the configuration with damage located at the 6th floor. Floor 1's estimated stiffness values (246, 228, 233, 240) were averaged together (column on the right) to get the best representation for the inter-story stiffness of Floor 1.

Averages were taken for every floor of every test and the averaged test results for the structure are shown in Table 3-2.

Stiffness Estimation (lb/inch)						
Configuration	1	2	3	4	5	6
Damaged floors	None (Healthy)	6th	4th	1st	4 th & 6th	1 st & 4 th
Floor 1	249	237	253	195	267	209
Floor 2	274	266	269	278	257	262
Floor 3	251	264	266	248	272	266
Floor 4	268	252	228	253	216	213
Floor 5	269	280	279	260	285	283
Floor 6	288	253	287	288	250	285

Table 3-2 Smartphone Structural Stiffness Results

3.9 Analysis of Results

Analysis of the results in the previous section was dependant on the change in stiffness for the floor. By removing the inter story cross bracing the stiffness change was expected to reduce by approximately 12-18 percent.

% Change from Healthy Structure – Smartphone Accelerometer					
Configuration	2	3	4	5	6
Damaged floors	6th	4th	1st	4th & 6th	1st & 4th
Floor 1	-4.81	1.82	-21.84	7.36	-16.02
Floor 2	-2.87	-1.77	1.50	-6.28	-4.48
Floor 3	5.00	5.70	-1.39	8.27	5.98
Floor 4	-5.95	-14.82	-5.63	-19.28	-20.28
Floor 5	4.06	3.96	-3.36	6.06	5.33
Floor 6	-12.04	-0.15	0.00	-12.90	-1.05

Table 3-3 Percent Change of Stiffness Relative to Healthy Structure (shaded boxes indicate damage locations)

In Table 3-3, the percent change of stiffness values were found using:

$$\Delta \hat{k}_i (\%) = \left(\frac{\hat{k}_{i,\text{damaged}}}{\hat{k}_{i,\text{healthy}}} - 1 \right) * 100 \quad (6)$$

where $\hat{k}_{i,\text{damaged}}$ and $\hat{k}_{i,\text{healthy}}$ are the estimated stiffness values of the i -th floor on the damaged and healthy structure, respectively. Values in the table that have damage are highlighted in green. A negative number denoted a decrease in stiffness (*i.e.*, damage) for that floor. The process was run with every floor of every structure comparing exclusively to the healthy structure for the purpose of determining if the smartphones could detect damage in the structure. A damage that occurred between the 5th floor plate and 6th floor plate was referred to as “damage to the 6th floor”.

For a change in stiffness to be significant enough for the smartphones to identify damage the stiffness decrease has to be greater than 10 percent. Through previous experiments, a change of only 5% stiffness was too small for the sensors to accurately identify a stiffness change in the structure as it fell within the amount of anticipated error. As seen in Figure 3-14 6th and 4th Floor damaged, analysis can show a change of anywhere from 0 to 7% that can be associated to error.

This method was used to compare each floor from a damaged structure to the corresponding floor on the healthy structure; the results for each configuration are graphically represented in Figure 3-13 and Figure 3-14. In Figure 3-13, 6th Floor Damaged, damage is indicated by the largest negative value (over 10 percent) and only occurs on the 6th floor.

As seen in Figure 3-13, the smartphones were capable of determining stiffness losses on the floors; all damaged locations were successfully detected. The damage (stiffness losses) varied from 12.04% in Configuration 2 (6th floor damaged) to 21.84 % in Configuration 4 (1st floor damaged). However, there were some notable errors in the stiffness estimations. A few estimates are approaching the 10% thresholds to become false positives (*e.g.*, 8.27% stiffness loss on the 3rd floor in Configuration 5: 4th and 6th floors damaged). There were also negative stiffness loss estimates (positive % differences); since increases in stiffness were not expected, these estimations were ignored.

Figure 3-13 and Figure 3-14 graphically represent the change in stiffness next to the image of the structural configuration.

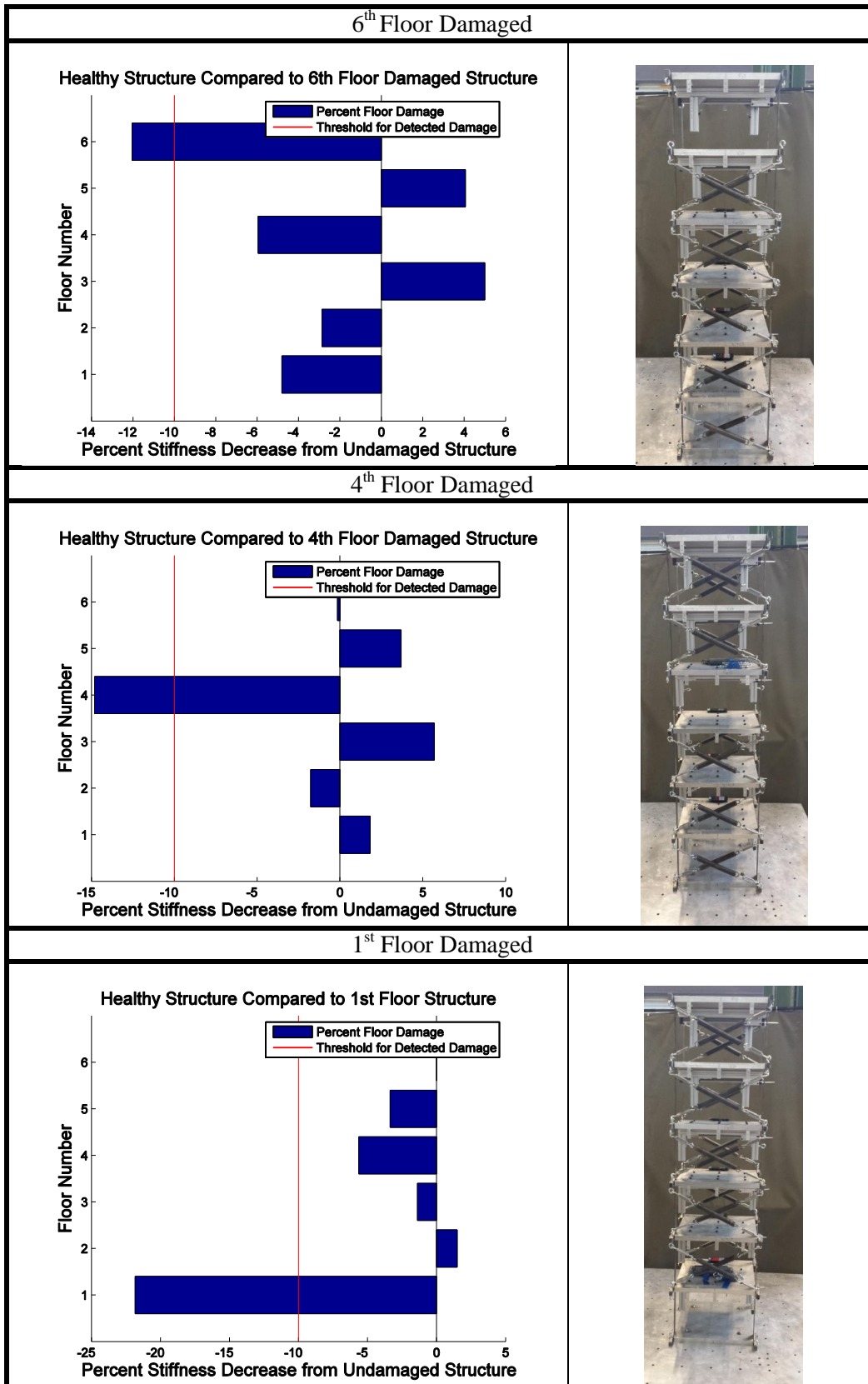


Figure 3-13 Graphical Structural Stiffness Changes 1

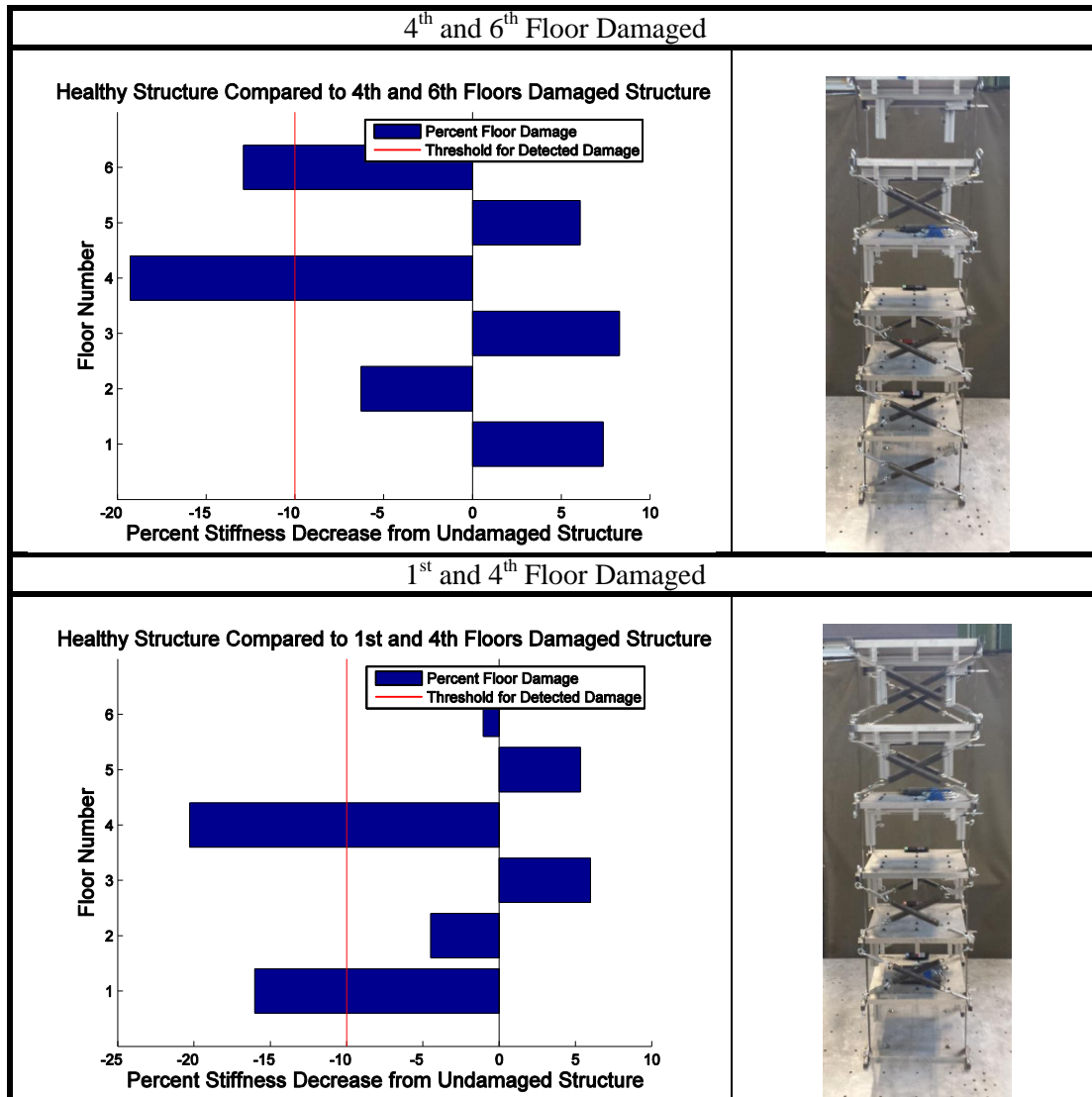


Figure 3-14 Graphical Structural Stiffness Changes 2

3.10 Smartphone Sensors Mounting and Synchronization Comparison

Data obtained from testing was post processed to confirm; (1) the time synchronization method on the smartphones was accurate when compared to Microstrain accelerometers, (2) using sticky pads was a viable method of attaching smartphones to an aluminum floor structure during a single axis test by recording the same accelerations as the Microstrain sensors.

3.10.1 Smartphone Synchronization Method Compared to Microstrain

A MATLAB script was written to confirm the time synchronization method was working properly to align the Microstrain and smartphone structural health monitoring data from the same floor. One assumption made is the Microstrain accelerometers are synchronized to each other exactly. Through research at the Microstrain website, it was determined the Microstrain accelerometers can be off by a maximum of 32 nano-seconds, which is smaller than the time scale the smartphone application is recorded.

The script would first take the smartphone data, synchronize and adjust the data similar to the procedures outlined in the Interpolation Data section of the thesis. The Microstrain accelerometer data was also brought in and saved to its own time vector, starting at zero. For the purpose of this example, the first floor of data was first examined; this data set contained the master phone for the smartphone data. The master phone was the device all other smartphones are synchronized while tested. First floor raw accelerations were graphed from the Microstrain sensors and smartphone, and are represented in Figure 3-15.

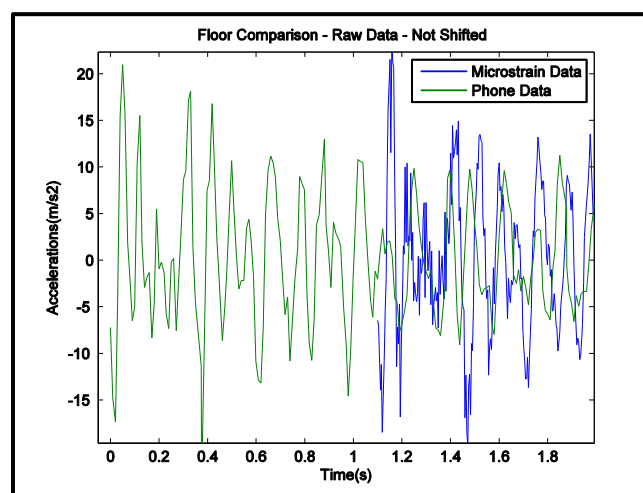


Figure 3-15 Microstrain to Smartphone Data Comparison - Raw Data

The figure above shows the first floor acceleration data for both Microstrain and smartphone sensors, however, they are not properly aligned. There is not a common time stamp between both the Microstrain and smartphone accelerometers so manual shifting of the data is necessary to line up the data for comparison. Figure 3-16 shows the data aligned as best as possible by eye. The data does appear to be aligned; however, as shown below further calculations are required to correct a few hundredths of a second difference.

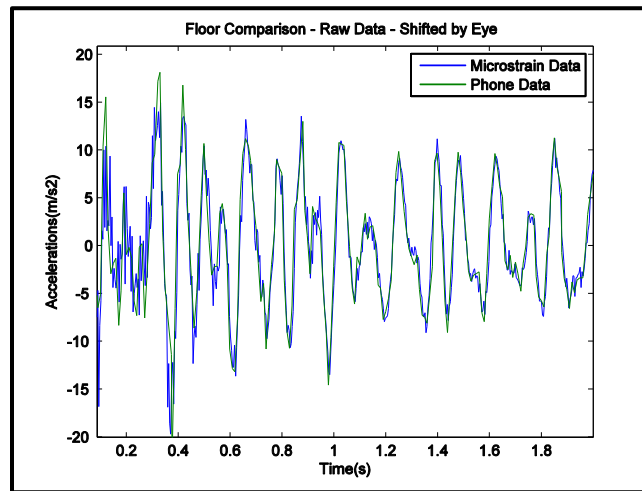


Figure 3-16 Microstrain to Smartphone Data Comparison - Raw Data Shifted by Eye

The major complication with aligning smartphone to Microstrain signals is aligning on the time axis. Although the accelerations are close right now, a method was developed to determine how much the smartphone data would be shifted to best line up with Microstrain data. To do this, the Microstrain data was shifted by the variable “shift_range”, interpolated onto the smartphones timestamp, the RMS (root mean square) of the difference of the smartphone and interpolated Microstrain accelerations or RMS_{diff} were taken (see Eq. 7), and the process is repeated to find the lowest difference in RMS_{diff} . The process is described further below.

Figure 3-17 shows the two variables used to determine the time shift necessary to line up the smartphone data to the Microstrain data.

```
shift_range = [-100:100]/HzRui; % range of times
% width = 6, length = shift_range
RMS_diff = zeros(length(shift_range),size(dataPhoneTimeALL,2));
```

Figure 3-17 Script Variables – shift_range and RMS_diff

where:

- shift_range = Range of times to shift Microstrain sensors signals, seconds
- HzRui = Sampling rate of Microstrain sensors, 256 hz
- dataPhoneTimeALL = Time matrix for all six devices
- RMS_diff = saves every RMS difference at the time it occurred in shift_range

The Figure 3-18 is the code used to determine how close the signals are with different time shifts applied to the Microstrain signal. The point when “RMS_diff” is the minimum will correspond to the time that best aligns the Microstrain with the smartphone data. To better describe Figure 3-18, the first loop will be discussed.

```

for kk = 1:length(shift_range), % use every time_shift value
    tRuiLoop = tRui + shift_range(kk); % shifts ruis constant time vector by kk
    % interp Ruis data onto phone times

for jj=1:size(dataRui,2), % Interpolates all of Ruis data to Phone timestamp
    dataRuiInterp(:,jj) = interp1(tRuiLoop,dataRui(:,jj),dataPhoneTimeALL(:,jj),'spline');
end

% For loop to calculate the RMS between two signals
for ii=1:size(dataRuiInterp,2) % calculates RMS for every floor
    xSumSquares = sum((dataPhoneALL(:,ii)-dataRuiInterp(:,ii)).^2);
    RMS_diff(kk,ii) = sqrt((1/length(dataPhoneALL))*xSumSquares); %rms equation
    % finds RMS for each time, and each floor
end
end

```

Figure 3-18 Calculate RMS Difference Values for Every Value of shift_range

where:

- dataRui = Microstrain accelerometer data
- tRui = time vector of the Microstrain signal
- tRuiLoop = time vector of the Microstrain signal with shift

The code starts by taking the first value from the “shift_range” array, which is -100/256. It then applies that shift to the variable “tRui” and changes it to “tRuiLoop”. The second for loop uses the new time stamp to interpolate every floor of the Microstrain data (“dataRui”) onto the smartphones timestamp (“dataPhoneTimeALL”). The last for loop uses the difference between each point of smartphone data subtracted by each data point of the Microstrain acceleration, and then uses the RMS_{diff} equation below. The equation works floor by floor. So “ii” starts at 1, or floor 1, and then moves to floor 2, etc.

$$RMS_{diff} = \sqrt{\frac{\sum_{i=1}^n (x_{i,smartphone} - \bar{x}_{i,Microstrain})^2}{n}} \quad (7)$$

The equation above takes the acceleration of smartphone and Microstrain data ($x_{i,smartphone}$ and $\bar{x}_{i,Microstrain}$, respectively) at the first data point, calculates the difference and find the RMS of the differences for all n sampling points in the acceleration data. The closer the number is to zero

means the better the signals match up. The “RMS_diff” the first time would be saved at the first row in the first column which corresponds to the first time shift of the Microstrain data and the first floor. This process is then repeated for the first floor by moving the time shift over, interpolating Microstrain data, and then finding the “RMS_diff” until all “time_shift” values have been used, and then moves to the second floor.

Once the code is completed, the Figure 3-19 finds the minimum RMS value for each floor and its location in the vector. The RMS value and location are saved into the “RMS_Val_Loc” variable with the first row corresponding to the first floor data, second row corresponding to the second floor data and so forth.

```

% loop used to find the minimum RMS value of each floor, puts Info in:
% Floor1(value, location)...
% Floor6(value, location)
for floor = 1:size(RMS_diff,2),
    [val,loc] = min(RMS_diff(:,floor));
    RMS_Val_Loc(floor,:) = [val,loc];
end

```

Figure 3-19 Minimum RMS Difference Value and Location for Each Floor

Ideally, the second column in the “RMS_Val_Loc”, which corresponds to the location in the “time_shift” vector for each floor, will all contain the same number. If all the values are the same in one single test, it means that each floor of the smartphone data would need to be shifted the same distance to best match the Microstrain accelerometer data. With every value the same, it could be concluded the method of phone time synchronization works. Comparing between tests is not necessary, because the location only shows how close the Microstrain data was to the smartphone data during the manual shift process. Values in Table 3-4 were determined using the above method for all five tests in Configuration 1: Healthy Structure.

Time Shift Location by Floor to Align Smartphone to Microstrain Data (Data Point)					
	Test 1	Test 2	Test 3	Test 4	Test 5
	Min. RMS_{diff} Location	Min. RMS_{diff} Location	Min. RMS_{diff} Location	Min. RMS_{diff} Location	Min. RMS_{diff} Location
Floor 1	100	70	104	107	89
Floor 2	101	71	105	107	90
Floor 3	99	69	102	105	88
Floor 4	100	71	104	109	88
Floor 5	100	69	103	106	88
Floor 6	101	74	104	107	89

Table 3-4 Location in shift_range to Align Smartphone and Microstrain Data

To simplify, and compare data obtained from difference tests, every value in one tests vector is subtracted from the first floor value and then divided by 256 to obtain the time difference between the raw synchronized data and the correct shift. If the normalized time shift was applied to the smartphone data, it should result in the “most accurate” response to the structure. In Table 3-5, a value of zero denotes an exact synchronization between the floor of interest, and the first floor of that test. A value of plus/minus 0.0039 seconds or 1/256 is a 1 data point shift.

Time Synchronization Error in Smartphones (Seconds)					
	Test 1	Test 2	Test 3	Test 4	Test 5
Floor 1	0	0	0	0	0
Floor 2	-0.0039	-0.0039	-0.0039	0	-0.0039
Floor 3	0.0039	0.0039	0.0078	0.0078	0.0039
Floor 4	0	-0.0039	0	-0.0078	0.0039
Floor 5	0	0.0039	0.0039	0.0039	0.0039
Floor 6	-0.0039	-0.0156	0	0	0

Table 3-5 Time Synchronization Error in Smartphones

From the data above, the greatest shift for any floor in the five tests occurred on test two, floor six, and corresponded to a four data point shift. Of 25 floors that could be shifted (Floors 2 through 6 of Tests 1 through 5) 18 floor shifts were necessary. 13 of the 18 floor shifts occurred at the smallest shift used in the experiment, 0.0039 seconds (or one time step in the 256Hz sampling rate of the Microstrain data). From this data, it appears that the synchronization method is fairly accurate compared to the method used for the Microstrain accelerometers.

3.10.2 Synchronization Error Affect on Smartphone Stiffness Values

The next step was to determine how the time synchronization error would affect the stiffness values obtained from testing. Each normalized time shift was applied to the corresponding floor of the smartphone data, the data was the interpolated to a 100Hz time stamp, and plugged into the ERA code similar to section “Post Processing Procedure”. After calculating the data, the results are shown in Table 3-6.

Stiffness Values with Adjusted Synchronization - Healthy Configuration (lbs/in)					
	Test 1	Test 2	Test 3	Test 4	Test 5
Floor 1	263	274	294	288	278
Floor 2	269	260	259	261	262
Floor 3	268	281	274	273	270
Floor 4	272	268	277	278	277
Floor 5	288	285	284	266	279
Floor 6	282	291	283	297	292

Table 3-6 Stiffness Values with Adjusted Synchronization – Healthy Structure

These values were then averaged together and compared to stiffness values obtained without the synchronization shift. The percent change from the Microstrain accelerations was also taken to show the difference between Microstrain signals and smartphones. Table 3-7 and Table 3-8 show the smartphones data would better resemble the Microstrain stiffness values if the smartphone synchronization method was improved over the current method. The maximum percent difference with the adjusted time data was 6 percent. While the maximum percent difference of the old data was 9 percent. If all the absolute value of all the percent differences were averaged together, the difference for adjusted values would be 2% where the non-adjusted values would be 4% different than the Microstrain data.

Stiffness Values with RMS Adjusted Time (lbs/in)			
	Microstrain	Smartphone	% Difference
Floor 1	263	279	6
Floor 2	265	262	-1
Floor 3	275	273	-1
Floor 4	270	275	2
Floor 5	279	280	0
Floor 6	291	289	-1

Table 3-7 Microstrain to Adjusted Smartphone Stiffness Comparison

Stiffness Values without RMS Adjusted Time (lbs/in)			
	Microstrain	Smartphone	% Difference
Floor 1	263	249	-5
Floor 2	265	274	3
Floor 3	275	251	-9
Floor 4	270	268	-1
Floor 5	279	269	-4
Floor 6	291	288	-1

Table 3-8 Microstrain to non-adjusted Smartphone Stiffness Comparison

3.10.3 Synchronization Error Affect on Microstrain Data

After understanding there was some error involved in the stiffness values from the smartphone accelerometers, the next step was to see how that error would affect the Microstrain accelerometer system. The Microstrain accelerometers were assumed to have a zero error when considering time synchronization. This allowed for a “Time Synchronization Error” to be added to floors in three configurations. Configuration A forced the top 3 floors of the structure to have a $1/256$ error in their synchronization. Configuration B forced the top three floors to have a $3/256$. The value $3/256$ was chosen because it best represented the error associated with smartphone application time synchronization method. Configuration C imposed a $1/256$ second time shift to every other floor. After shifting the data, for the configurations and taking the stiffness values using the ERA and Stiffness algorithms in Appendix the stiffness values are shown in Table 3-9.

Stiffness Values with Synchronization Error (lbs/inch)				
Configuration	Normal	A	B	C
Error	None	Top 3- 1/256	Top 3 - 3/256	1/256 - Alt
Floor 1	265	266	265	250
Floor 2	276	275	274	279
Floor 3	272	278	289	265
Floor 4	272	271	244	281
Floor 5	280	285	296	275
Floor 6	289	289	288	291

Table 3-9 Stiffness Values with Synchronization Error

After calculating all stiffness values for the different synchronization error the average difference between the error and normal configurations are shown in Table 3-10, Table 3-11, and Table 3-12

Top 3 Floors 1/256 Sync Error Vs Normal (lbs/inch)			
	Normal	Time Sync Error	Difference
Floor 1	265	266	0.66
Floor 2	276	275	1.11
Floor 3	272	278	6.09
Floor 4	272	271	1.01
Floor 5	280	285	5.26
Floor 6	289	289	0.62
	Average Difference		2.46

Table 3-10 Top 3 Floors 1/256 Sync Error Vs Normal

Top 3 Floors 3/256 Sync Error Vs Normal			
	Normal	Time Sync Error	Difference
Floor 1	265	265	0.13
Floor 2	276	274	1.68
Floor 3	272	289	17.28
Floor 4	272	244	28.07
Floor 5	280	296	16.19
Floor 6	289	288	1.54
	Average Difference		10.81

Table 3-11 Top 3 Floors 3/256 Sync Error Vs Normal

Every Other Floor Shifted 1/256 Sync Error Vs Normal			
	Normal	Time Sync Error	Difference
Floor 1	265	250	14.47
Floor 2	276	279	3.72
Floor 3	272	265	7.31
Floor 4	272	281	8.93
Floor 5	280	275	4.46
Floor 6	289	291	1.69
	Average Difference		6.76

Table 3-12 Every Other Floor Shifted 1/256 Sync Error Vs Normal

Through the analysis of the smartphone data, the threshold for smartphone data is 3/256 seconds. A difference of 3/256 seconds synchronization error at Configuration B yielded an average error of 10.81 lbs/in, which accounts for about 4% difference from the expected stiffness values. In experiments using smartphone devices with the application's synchronization method, it is recommended to account for at least a 5% error in stiffness values, if initial value of the structure is above 200 lbs/in.

3.10.4 Smartphone Mounting Techniques Compared to Microstrain

After confirming the synchronization method was adequate for experimental purposes, an experiment was setup to prove sticky pads would not add error to the recorded acceleration data and prove sticky pads were a viable method of attaching smartphones to a structure. If the data recorded using sticky pads as a mounting method could be closely replicated by both another method and the Microstrain accelerometers, the sticky pads would prove to provide enough friction to reliably gather structural response data for a lab or experimental experiment.

The setup for the experiment can be seen in Figure 3-20, used four smartphone devices, and two Microstrain accelerometers all secured to the 4th floor of the structure in three different methods. The structure was mounted to the shake table using four bolts and the shake table underwent a square wave with 0.5 inch amplitude. There was one Microstrain accelerometer mounted to two opposite corners of the floor. These were the only two locations on the floor with pre drilled holes for the Microstrain accelerometers. As seen in Figure 3-21, a smartphone was placed on a sticky pad and another smartphone placed on a Velcro strip above the Microstrain accelerometers. The sticky pads were used during structural testing and the Velcro strip, capable of holding up 20 pounds, was used as a possible alternative. As displayed in Figure 3-22, one half of the Velcro strip was attached to the structure and the other half to the back side of the smartphone so both would align with the structure's motion.



Figure 3-20 Overall Setup - Sticky Pad Test

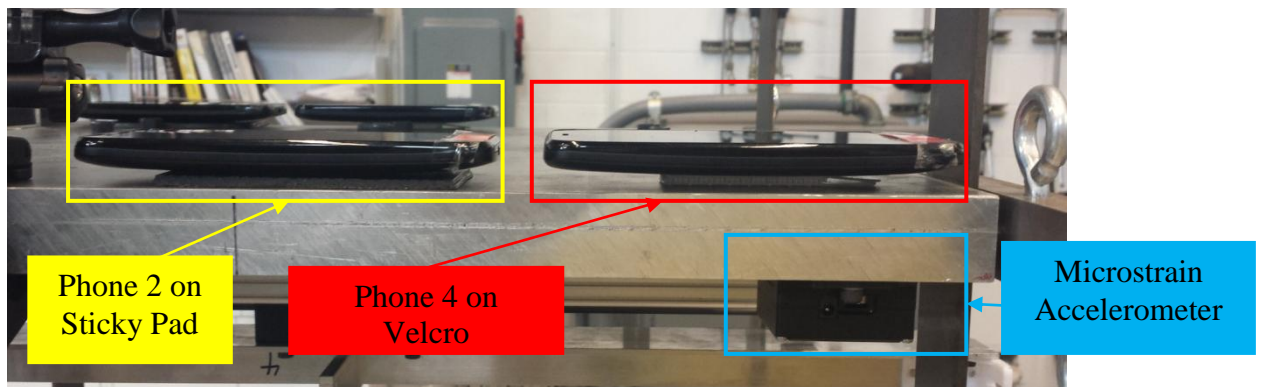


Figure 3-21 Sticky Pad Testing - Side View

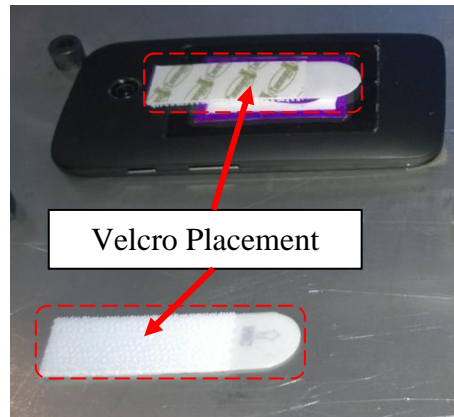


Figure 3-22 Velcro Placement

Sensors were divided by their location on two different sides of the structure. The breakdown can be seen in Table 3-13.

Sticky Pad Testing Sensor Locations		
Sensor Name	Side of Surface	Mounting Method
557 (Microstrain)	Left	Screwed
Phone 2 (Moto G)	Left	Sticky Pad
Phone 4 (Moto G)	Left	Velcro
555 (Microstrain)	Right	Screwed
Phone 1 (Moto G)	Right	Sticky Pad
Phone 3 (Moto G)	Right	Velcro

Table 3-13 Sticky Pad Testing Sensor Locations

The tests were conducted and the results were graphed by which side of the structure the data was coming from to eliminate any error due to torsion, and to also mimic how the data was collected in the structural testing. During structural testing, a phone on a sticky pad was placed directly over the Microstrain accelerometer. The response of the three devices on the left side and right side of the structure, lined up by eye, can be seen in Figure 3-23 and Figure 3-24 respectively.

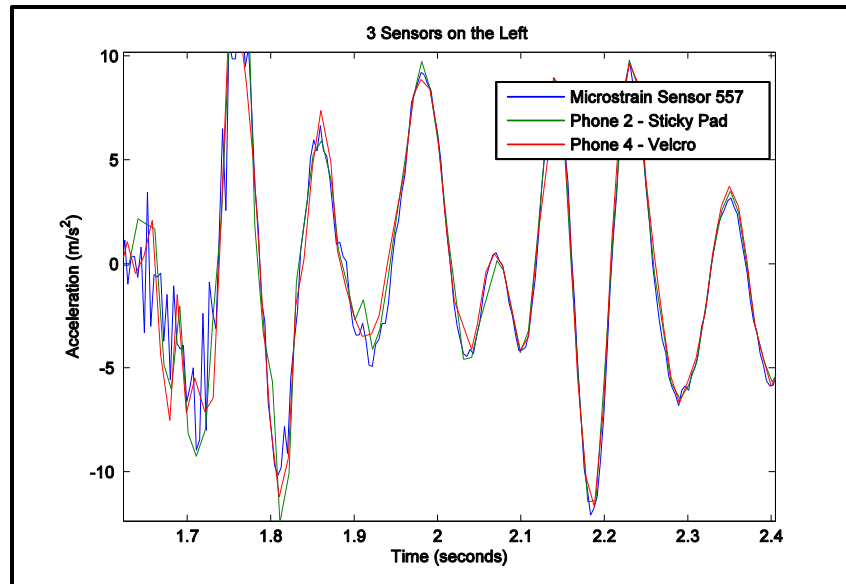


Figure 3-23 Sensors on Left (Synchronized)

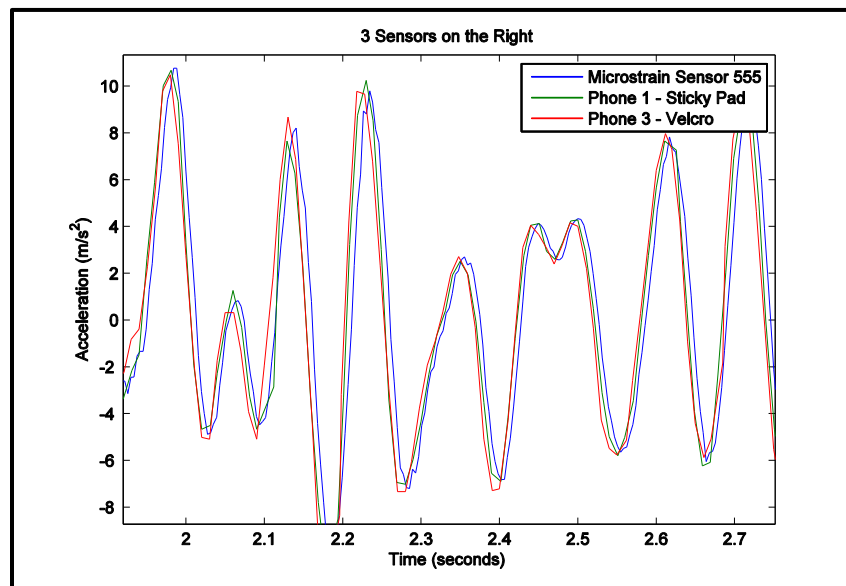


Figure 3-24 Sensors Right Side (Synchronized)

Graphically, accelerations from the smartphone sensors matched up very well with the accelerations from the Microstrain accelerometers. To quantify how well the signals between the devices matched up, Equation 7 in Section 3.10.1 was used to find the RMS of the difference between the two signals. Using this method, it would best line up the data from the Microstrain

accelerometer to each smartphone on the respective side. Sensor 557 was best matched with Phone 2, interpolated to the same timestamp as Phone 2, and then the difference in RMS value was found. This process was repeated until the RMS difference for both Microstrain accelerometers was calculated on each side. After completing the process for both Microstrain accelerometers, it was repeated for the four tests and the data is represented in Table 3-14.

Average RMS_{x-y} (Average of four tests; units in m/s^2)		x					
		P1 (Sticky Pad)	P2 (Velcro)	P3 (Sticky Pad)	P4 (Velcro)	M557	M555
y	P1 (Sticky Pad)	0.000	0.595	0.735	0.704	0.700	0.389
	P2 (Velcro)	0.595	0.000	0.632	0.442	0.621	0.549
	P3 (Sticky Pad)	0.735	0.632	0.000	0.631	0.734	0.455
	P4 (Velcro)	0.704	0.442	0.631	0.000	0.698	0.529
	M557	0.700	0.621	0.734	0.698	0.000	0.519
	M555	0.389	0.549	0.455	0.529	0.519	0.000

Table 3-14 RMS Sensor Comparison

When comparing the RMS Motorola G smartphone data to the Microstrain data in Table 3-14, the results showed similar RMS differences for all three methods, Microstrain, Sticky Pad and Velcro. When comparing the two Microstrain sensors, M555 and M557, that were attached to the structure with screws (*i.e.*, the most secured attachments of the three methods), they showed an average RMS value of $0.519m/s^2$. This helped established the baseline of expected difference between the measurements of two sensors that were securely attached on the same floor. The last two rows in Table 3-14 show comparisons of the two phone attachment methods of using sticky pads and Velcro to the Microstrain sensors. The last row (M555) had similar to slightly smaller RMS differences than the baseline value of $0.519m/s^2$, while the second last row (M557) showed slightly larger RMS differences than the baseline value. These observations implied that the sticky pad and Velcro methods were good alternatives to the screw attachments.

When comparing sticky pad and Velcro data to one another, the RMS values range from 0.442m/s^2 (below 0.519m/s^2) to 0.735m/s^2 (above 0.519m/s^2). In other words, data obtained from Sticky pads and Velcro methods were at most 0.235 m/s^2 worse than comparing the baseline value while some results were better. Using the data in Table 3-14 it was possible to conclude the smartphones recorded similar accelerations on the sticky pads and the Velcro strips as Microstrain accelerometers, therefore, both Sticky pads and Velcro would be sufficient methods of attaching smartphones to a structure for experimental purposes.

3.11 Stiffness Results Compared to a Dedicated Sensing Network

Comparing the results from smartphone sensing system to a dedicated SHM sensing system was important to validate results gathered from the sensors. The University of New Hampshire had Microstrain® accelerometers with the specifications that are listed in Table 1-1. These Microstrain® wireless sensors were attached to the structure while testing was completed with the smartphone devices. Structural vibration data were collected for both sensing systems, the 6th floor data for each sensing system were plotted together, stiffness values of the damaged structures were calculated, and the standard deviation of each floor's stiffness was calculated.

3.11.1 Stiffness from Microstrain Sensors and Smartphone Sensors

Structural analysis was performed as discussed in Section 3.8 “Structural Analysis Results” to calculate experimental stiffness for the experimental structure in the high bay at UNH. Multiple tests were conducted per configuration and the results of each floor were averaged for each configuration. Results from the tests conducted for both smartphone and

Microstrain sensors are displayed below in Table 3-15 and Table 3-16, respectively. Both figures display the stiffness of each floor in pounds per inch.

Structure Results – Smartphones (lb/inch)						
Configuration	1	2	3	4	5	6
Damaged floors	None (Healthy)	6th	4th	1st	4 th & 6th	1 st & 4 th
Floor 1	249	237	253	195	267	209
Floor 2	274	266	269	278	257	262
Floor 3	251	264	266	248	272	266
Floor 4	268	252	228	253	216	213
Floor 5	269	280	279	260	285	283
Floor 6	288	253	287	288	250	285

Table 3-15 Smartphones Stiffness Results

Structure Results – Microstrain Sensors (lb/inch)						
Configuration	1	2	3	4	5	6
Damaged floors	None (Healthy)	6th	4th	1st	4 th & 6th	1 st & 4 th
Floor 1	263	262	284	208	253	216
Floor 2	265	271	264	269	264	263
Floor 3	275	268	275	267	278	277
Floor 4	270	272	236	270	231	232
Floor 5	279	281	278	272	282	279
Floor 6	291	252	292	294	254	293

Table 3-16 Microstrain Stiffness Results

Table 3-17 and Table 3-18 show the percent decrease of stiffness per floor when comparing each configuration to the healthy structure. Damaged floors are highlighted in green. Both methods were capable of detecting the damage locations when comparing the healthy and the damage configurations. Generally, the Microstrain sensor estimates had less errors in the non-damaged locations compared to the estimates from the smartphone sensors.

% Change from Healthy Structure – Smartphone Accelerometer					
Configuration	2	3	4	5	6
Damaged floors	6th	4th	1st	4th & 6th	1st & 4th
Floor 1	-4.81	1.82	-21.84	7.36	-16.02
Floor 2	-2.87	-1.77	1.50	-6.28	-4.48
Floor 3	5.00	5.70	-1.39	8.27	5.98
Floor 4	-5.95	-14.82	-5.63	-19.28	-20.28
Floor 5	4.06	3.96	-3.36	6.06	5.33
Floor 6	-12.04	-0.15	0.00	-12.90	-1.05

Table 3-17 Percent Stiffness Change - Smartphone Accelerometer

% Change from Healthy Structure – Microstrain Accelerometer					
Configuration	2	3	4	5	6
Damaged floors	6th	4th	1st	4th & 6th	1st & 4th
Floor 1	-0.40	8.01	-20.80	-3.66	-17.67
Floor 2	2.36	-0.18	1.64	-0.20	-0.84
Floor 3	-2.73	-0.28	-3.04	0.82	0.60
Floor 4	0.78	-12.44	0.12	-14.50	-14.10
Floor 5	0.47	-0.56	-2.73	1.01	-0.11
Floor 6	-13.44	0.41	0.96	-12.66	0.81

Table 3-18 Percent Stiffness Change - Microstrain Accelerometer

3.12 Summary: Structural Analysis with Smartphones

As shown in Section 3.8 “Analysis of Results,” smartphones, using the developed application, are capable of identifying damage in a structure and could be used in a lab and/or classroom setting as a less expensive alternative to dedicated sensing systems.

The time synchronization error allowance for any system is dependent on the system and structure used. This synchronization method developed for the use on the smartphone application proved to have a maximum synchronization error of 0.0117 seconds. Using this maximum error threshold (0.0117 seconds) and applying it to data recorded by the Microstrain accelerometers in

structural health testing, an error due to the application synchronization process could be as much as 4% of the 200 lb/inch inter-story stiffness.

To confirm Sticky pads were a reliable method to secure a smartphone device to an aluminum structure, accelerations were recorded with sensors mounted in three methods; Sticky pads, Velcro, and screwed into the structure. Testing confirmed both Sticky pads and Velcro were a dependable method of recording comparable accelerations to the Microstrain accelerometers because both methods calculated similar RMS difference values within 0.25 m/s^2 of the baseline set by comparing Microstrain accelerometers.

Testing was conducted using the smartphones on a scaled six story structure. Six different damage configurations were applied to the structure by removing cross bracing. For each testing configuration smartphones were mounted to the structure using sticky pads and structural accelerations were recorded using a developed smartphone application. A dedicated sensing system was also mounted to the structure to obtain accelerations and allow the smartphone sensors to compare their data to the dedicated system.

Both systems were capable of returning similar accelerations which lead to post processing to compare how smartphones could evaluate damage to a structure. After post processing data from both the smartphone sensors and the dedicated sensing system damaged locations on the structure were successfully located.

Smartphones, with the programmed application could be included into an educational curriculum to add an alternative means of determining a stiffness change in a structure. As the accelerometer in smartphones improves, the method could only be improved with faster processors and more sensitive accelerometers.

Chapter 4 Outreach Program

On two separate occasions, April 14th 2014 and December 11th 2014, an educational outreach was conducted at the Hillside Middle School in Manchester, New Hampshire that coincided with 7th grade teacher Mr. Baron Richardson's curriculum. The outreach program was constructed to educate the students on how stiffness and damping would affect a structure, and also to allow the students to gain hands on experience with structures. Smartphones were used to test the structures in Mr. Baron Richardson's classroom, and supplement his structures curriculum. The second outreach program in December was an improved program based on the experience and feedback from the first outreach program in April.

	Outreach 1- Pilot (April 14 th , 2014)	Outreach 2 - Improved (Dec 11 th , 2014)
Activities Conducted	Pre-outreach survey Activity 1 and Activity 3 Post-outreach Survey	Pre-outreach Survey Activity 1 and Activity 2 Post-outreach Survey
# of students	87	96
Changes Between Outreach Programs	Issues discovered: Phone Code –Activity 1 (Hardcoded) Not Enough Smartphones (4) wanted smaller groups No Questions to Quantify Interest in STEM	Improvements: Ability to choose Structure Stiffness and Damping Brought 8 smartphones, school provided 30 Interest in STEM on Pre and Post Activities

Table 4-1 Outreach Differences

On April 14th, 2014, the date of the pilot outreach, the following activities were conducted: A pre-outreach survey (described in Section 4.2 Pre-outreach Survey) was given on the prior day; on the day of the outreach, a presentation discussing stiffness and damping was given, followed by an in-class activity, and ending with a post-outreach survey. The in-class activity included “Activity 1” and “Activity 3” described in Section 4.5 and 4.7, respectively. Using the experience from the pilot outreach a few modifications were made to the primary outreach. The in-class activity was modified (“Activity 1”). The original code had to hardcode the stiffness and damping combination for the structures, the second outreach provided the user (*e.g.*, the teacher) the option to select the combination of stiffness and damping. Pre- and post-outreach surveys added a question quantifying the student’s interest in Science, Technology, Engineering, and Math (STEM). Also, through the post-outreach survey questions, it was evident students enjoyed the hands on activity, and more than four phones were needed to allow more students to have an interactive experience with the structures. The first outreach confirmed the activity and presentation helped increase the students’ knowledge of stiffness and damping in structures.

The December 11th outreach began the day before with pre-outreach survey. The purpose of the survey was to set a baseline of student understanding in regards to stiffness and damping, as well as gauge interest in STEM. On the day of the outreach, a presentation was given that discussed the effects of stiffness and damping on a structure. After the presentation two activities were conducted and are outlined in Sections 4.5 “Activity 1” and 4.6 “Activity 2”. At the end of the session, a post-outreach survey was given to the students which contained the same questions as the pre-outreach survey.

4.1 Student Breakdown

For the outreach conducted on December 11th 2014, the students broke down the following way:

	Morning		Afternoon	
	Session 1	Session 2	Session 3	Session 4
Boys	12	12	16	14
Girls	14	14	10	6
Total	26	26	26	20

Table 4-2 Student Breakdown by Class

In Activity 1, the morning class only had eight smartphones available to them. The school had recently purchased new phones, and they did not have the new application installed, so eight smartphones from UNH were used in the morning. The eight smartphones in the morning class only allowed for two separate groups of students during Activity 1. Between the morning and afternoon class, 16 smartphones became available because the application was installed in these phones purchased by the Hillside Middle School. Thus, in the afternoon sessions, there were considerably more smartphones (24) for similar class size. Table 4-3 shows the phone-to-student ratio drop from 3.25 to about 1 between the morning and afternoon classes. A smaller phone-to-student ratio allowed the students to more directly interact with the phones.

	Morning		Afternoon	
	Class 1	Class 2	Class 3	Class 4
Smartphones	8	8	24	24
Students	26	24	26	20
Smartphones per Student	3.25	3.25	1.08	.833

Table 4-3 Activity 1 - Smartphone Breakdown per Class

Figure 4-1 displays the large difference between having eight smartphones for the class (Right) versus having 24 smartphones for the class (Left). The group on the left only has four students playing with four smartphones. Limiting the amount of students per group allows each student to participate. The right image shows several students who are not able to fit into the group.



Figure 4-1 Comparing Group Sizes (Small-Left, Large-Right)

4.2 Pre-outreach Survey

The day before conducting the outreach program at the middle school, a few questions were distributed to the students and the answers collected by Mr. Baron Richardson. The questions are as follows:

1. Which structure is more stiff
a. Structure that sways more (moves side to side wider)
b. Structure that sways less
c. Not sure what a stiff structure means
2. Which would typically move more in an earthquake or windstorm?
a. Tall Structure
b. Short Structure
c. They would move the same
d. Not sure
3. Will a structure with higher damping, sway for a ____
a. Longer time
b. Shorter time
c. Equal time
d. Not sure what damping is
4. On a scale from 1 to 10 (1 being no interest, 10 being very interested), how interested are you in Science, Technology, Engineering, and Math?
1 2 3 4 5 6 7 8 9 10

Figure 4-2 Outreach Quiz

The purpose of the questions was to gain an initial understanding of the student's knowledge in regards to stiffness, damping, and structural behaviors. As well as gauge their interest in STEM. Answers for these questions were collected and recorded and used to compare to the post-outreach survey results.

4.3 Post-outreach Survey

At the conclusion of the presentation, and activities (discussed in Section 4.4), a post-outreach survey was distributed to students. The post-outreach survey was given to the students to quantify the success of the outreach program through the knowledge gained by the students evident by the change from the pre-outreach survey.

4.4 The Presentation

The outreach began with a presentation that explained two variables that are used to quantify structural vibrations. The first variable explained to the seventh grade audience was stiffness. Stiffness, was defined as a variable used to measure the rigidity of an object. In relation to structures, a building with a larger stiffness, K , would move less when excited. As an example for the students to better understand K , we used videos of previous tests conducted on the University of New Hampshire shake table. One video displayed a structure with no bracing and the other video had a braced structure with springs along the diagonal Figure 4-3. The videos were played directly following each other so the seventh graders could visually see the difference in K between the two structures. In regards to the stiffness of a structure, the students were taught to associate structures with a larger sway to have a lower stiffness and structures with a small sway to have a larger stiffness coefficient.



Figure 4-3 Braced Structure

The second variable explained to the students was damping or C . Damping was defined to the 7th graders as the ability to remove energy from the system. To give the students a practical implementation of dampers in action, it was decided to use a door damper. It was explained that a door damper's responsibility was to take energy away from the door making it come to a stop and prevent the door from slamming shut. After representing damping from an object that they could find in the school, damping was then related to moving a hand through water. Water sliding between each of your fingers was compared to the internal mechanism of a damper sliding through the viscous fluid inside a damper Figure 4-4. A video involving dampers' effect on a structure was then shown to the students. The video displayed two structures, one structure had a low C value and the other had a very high C value. Both structures were displaced, the low C value structure swaying until the person stopped the motion, and the high C value structure coming back to an immediate stop. Both structures were excited by a shake table until the low C value structure began to collide with the high C value structure. One of the students expressed he really enjoyed this video, and it helped him in the later activities that day. From the explanation

of damping, students were encouraged to relate a structure that vibrates for a longer period of time to have a lower damping value, and a structure with a shorter vibration period to have a higher damping value. The explanations of both stiffness and damping were used in the following activity.

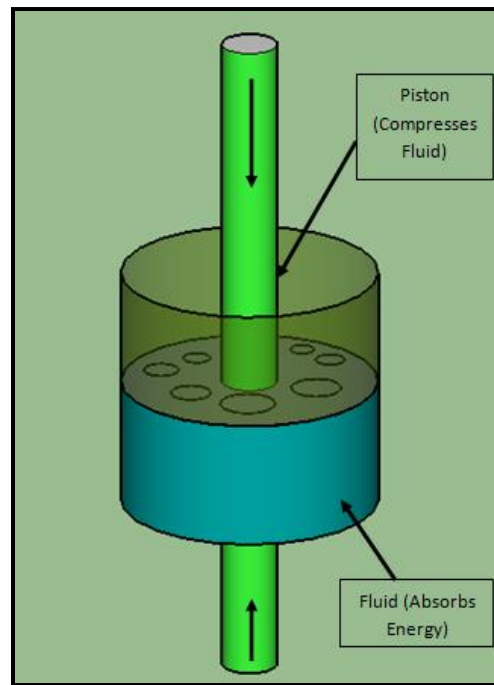


Figure 4-4 Damper Example

4.5 Activity 1 – Determining Stiffness and Damping

An activity was created to gauge the students understanding of damping and stiffness impact on a structure. A smartphone application was developed that would allow users to excite a simple shear structure based on user inputs. More specifically, users could use the touch screen to move every floor to the left or right and click “animate” Figure 2-20 Displace the Structure to simulate and animate floor motions based on input floor. Under the “Mr. Richardson’s Class”

portion of the application, each phone was assigned to display a structure with a different combination of stiffness and damping.

Name: _____				
Phone (circle one)				Structure with
1	2	3	4	High Stiffness & High Damping
1	2	3	4	High Stiffness & Low Damping
1	2	3	4	Low Stiffness & High Damping
1	2	3	4	Low Stiffness & Low Damping

Figure 4-5 Blank Phone Activity

The four combinations used in the activity were; (i) high stiffness and high damping, (ii) high stiffness and low damping, (iii) low stiffness and high damping, and (iv) low stiffness and low damping. The smartphones were split into groups of four phones and placed on a table in the classroom and each phone was given a number from 1 to 4 (Figure 4-6 – left). Students in each class were divided into smaller groups to interact directly with the application, displace the floors, and animate the structure (Figure 4-6 - right).



Figure 4-6 Activity 1 Phone Layout

Figure 4-8 demonstrates the interactive activity conducted in the educational outreach. Each group of students would displace the structure to look like Time 0 (initial condition), and click the “animate” button on the touch screens. The structures would then react to the initial displacements in a manner represented by Figure 4-8. The structures with the higher damping ratios, Phones 3 and 4, came to rest at Time 4 while the structures with low damping ratios, Phones 1 and 2, were still reacting to the initial displacements. Stiffness, K , is the other variable affecting the structures in Figure 4-8. The structures with lower K , Phones 1 and 4, would displace further from the initial position compared to Phones 2 and 3 which had higher K . The correct answers to the activity are shown in Figure 4-7.

Name: _____

Phone (circle one)				Structure with
1	2	3	4	High Stiffness & High Damping
1	2	3	4	High Stiffness & Low Damping
1	2	3	4	Low Stiffness & High Damping
1	2	3	4	Low Stiffness & Low Damping

Figure 4-7 Phone Activity Answers

Each individual row in Figure 4-8 depicts a phones screen at four separate time intervals during the structural response. Time 0 corresponds to the initial impulse loads being applied to the structure. Time 1 is the instant immediately after the user clicks the “animate” button and the structure begins its response. Time 2, was determined to be the time it took for Phone 1 to make half an oscillation. Time 2 corresponded to 0.25 seconds. Time 3 was chosen as the time corresponding to when the structures with higher damping ratios, Phones 3 and 4, came to rest. The total time lapse of this figure is 0.5 seconds.

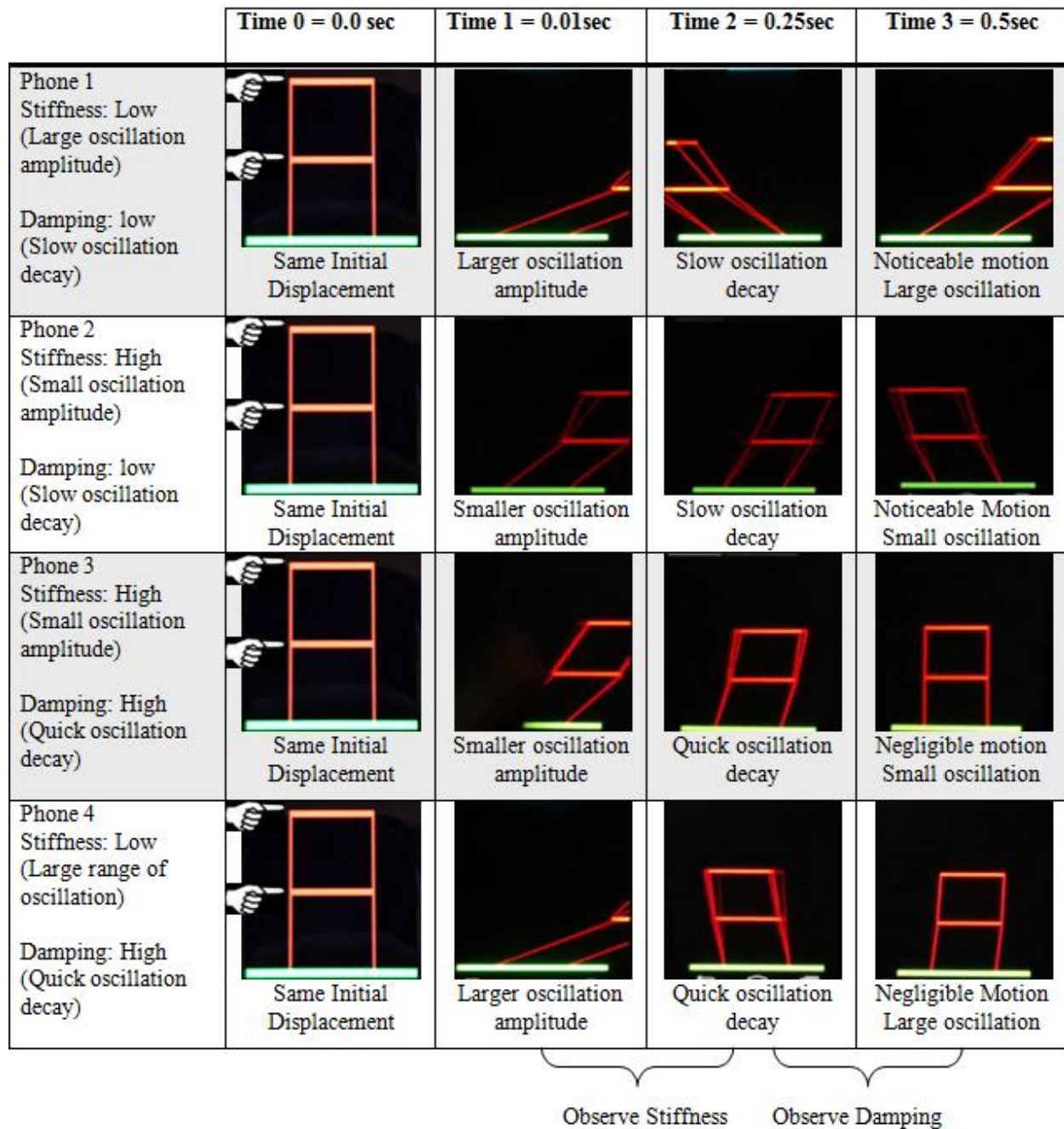


Figure 4-8 Phones Structural Response (Fingers Represent an Applied Point Load)

4.6 Activity 2 – Counting Oscillations

In this activity, two structures (Figure 4-9) were created by Mr. Baron Richardson and mounted to the counter of his classroom. The structure on the right was created to be the control structure; however it also has a higher stiffness because of a thicker column. The structure on the left is equipped with a liquid damper created by Mr. Baron Richardson and a smaller column than the right structure. The liquid damper could be turned off by tuning a valve in the center of the damper.

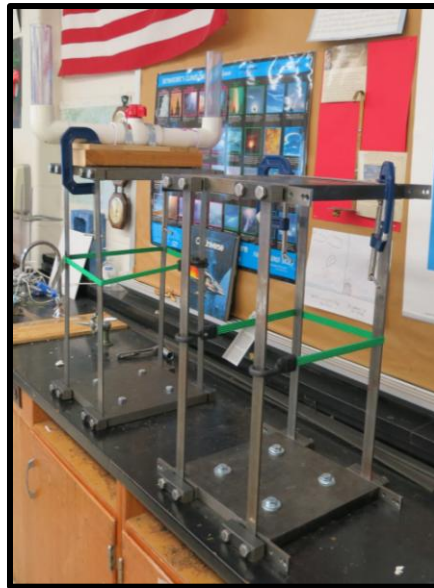


Figure 4-9 Class Metal Structures

For this activity, two smartphones were synchronized using the developed application. One smartphone was mounted to each structure using a sticky pad as outlined in Section 3.2 Smartphone Setup. Both structures were displaced (Figure 4-10) and released at the same time and allowed to vibrate. During the vibration phase, half the students counted the number of oscillations from the left structure and the other half, the right.

The purpose of this activity was to show the students the difference in oscillation cycles between the two structures. Due to the higher mass of the structure on the left, and the thicker columns on the right structure, the right structure was presented to the students as the more stiff structure. This was evident with the quicker oscillations cycles in Figure 4-11 and the students counting a larger number of oscillations for the structure on the right compared to the left structure.



Figure 4-10 Displacing Class Structures

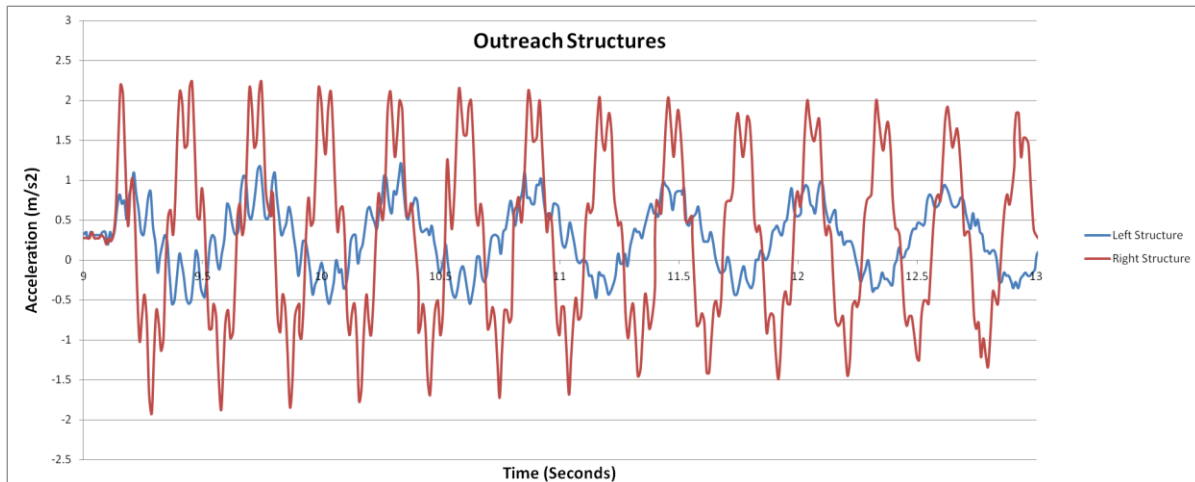


Figure 4-11 Structure Responses (as shown in class)

4.7 Activity 3 - Spaghetti Structure Test

A portion of the outreach program only conducted in the pilot was to find which of two spaghetti structures had a higher stiffness value; the two spaghetti structures, Uni Tatos and Bob's Builders, were built by the students. The goal of this test was to show the students by using smartphones, they could perform a small scale structural analysis similar to ones we could conduct at UNH. After relating the setup at UNH to the classroom setting, relative acceleration ratio was explained. When conducting the experiment, there needed to be a quick and simple method to measure the acceleration of the shake table and compare with the accelerations of the phone on the structure. By taking the root mean square (RMS) of the accelerations received by the phone on the structure and dividing it by the RMS of the shake table, it provided the relative acceleration received by both phones.

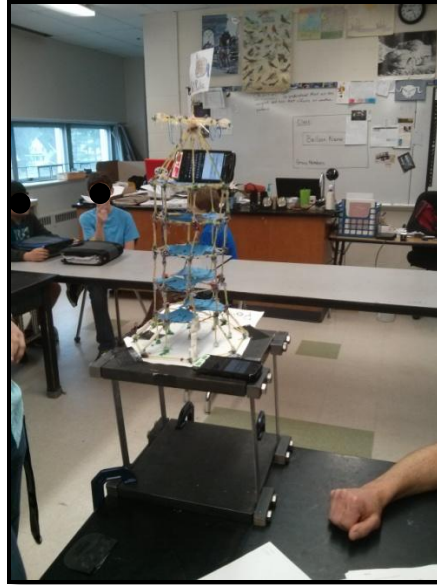


Figure 4-12 Spaghetti Structure "Uni Tatos"

The setup for the spaghetti structure test included; one spaghetti structure mounted to the shake table by attaching the structure's base using duct tape, one smartphone secured to the shake table using a sticky pad, and another smartphone placed on the top story of the structure. The setup of the tests can be seen in Figure 4-12. The structure was excited by giving the shake table an initial displacement and releasing.

Figure 4-13 shows two instances of the structural motion. The picture on the left shows the shake table at rest with the structure, Uni Tatos on top. The picture on the right depicts the process used to apply a ground motion to the structure. Since a person would apply the initial displacement of the shake table before release, it was almost impossible to replicate the same test with the exact excitation. To remedy this, the root mean squared responses of the structures and of the shake table response were used to determine the relative accelerations of the structure caused by the shake table.

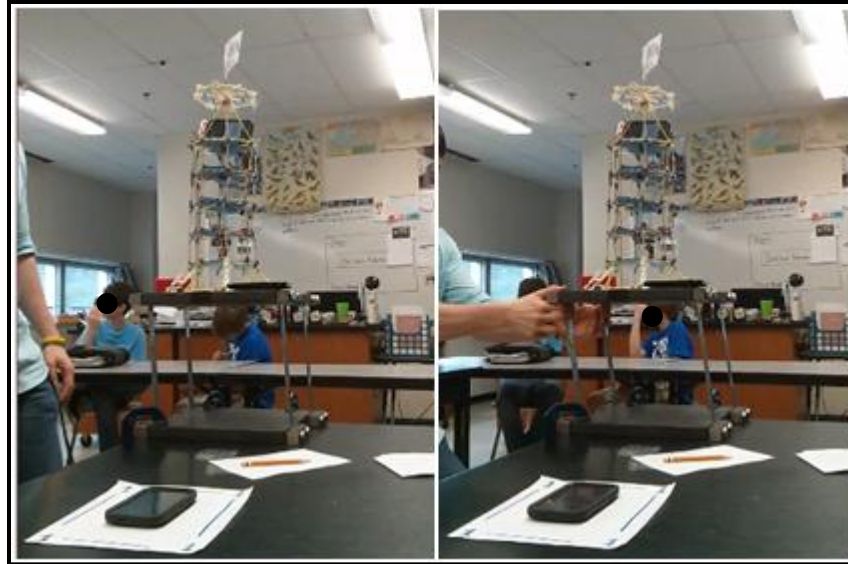


Figure 4-13 Displacing the “Shake table”

The developed smartphone application gave a greater advantage than present techniques to evaluate responses for the spaghetti structure test in a classroom setting. Currently, the classroom would use only visual cues of the structure to explain increased stiffness or damping. The application allowed for the class to: record data from both the shake table and the structure, combine both data files into one file, post process the data (outside of class), animate the structure using the combined data file and display the response ratio of that structure.

Upon completion of each test, what had occurred on the application (listed above) was shown to the students. It was explained that once the animate button on this portion of the application was clicked, the phone would take the file name, read the file, animate the recorded responses and calculate the difference in accelerations received from both phones. After both structures had been tested, the different response ratios were explained to the students.

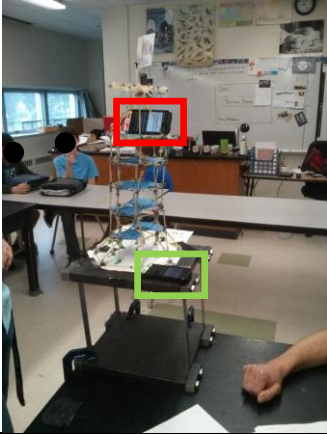
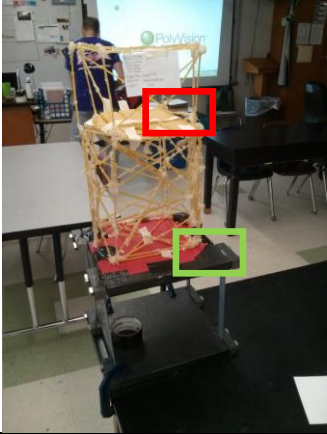
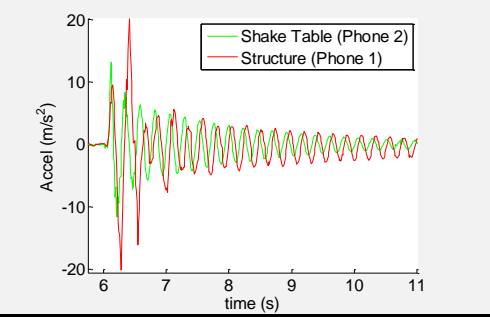
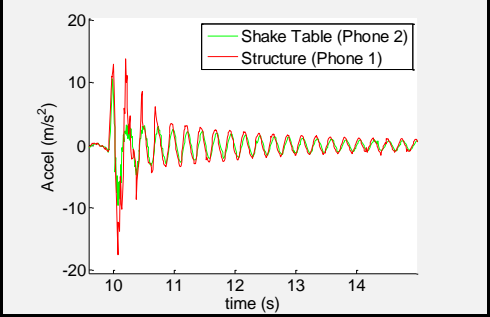
	Uni Tatos (Flexible Structure)	Bob's Builders (Stiff Structure)
Structure		
Acceleration Response		
Response Ratio:	1.4829705	1.533828

Figure 4-14 Spaghetti Structure Results

Figure 4-14 summarizes the spaghetti structures tests used during the outreach program. Prior to the outreach, the students were challenged to construct spaghetti structures they would later subject to a simple shake table test. For the outreach, the students were supervised as they picked a groups structure with a higher stiffness, Bobs Builders, and less stiff structure, Uni Tatos. In Figure 4-14, Uni Tatos is in the middle column and Bobs Builders is in the left column. Phone locations are boxed and the colors are coordinated with the graphs below. The graphs show the accelerations of the structure and of the shake table during the test. Bobs Builders structure was confirmed to be the structure with a higher stiffness due to the higher amplitudes in the acceleration plot and the response ratio being higher than the one seen by Uni Tatos.

4.8 Results

Correct answers	Both Classes – Enter	Both Classes – Exit
Question 1 and 2 (Stiffness)	86.74	87.03
Question 3 (Damping)	50.60	86.42

Table 4-4 Enter and Exit Quiz Results

To gain a better understanding of the data collected from the pre and post surveys, it was determined separate the responses by question. After looking at each question, it was determined that the first two questions related directly with stiffness of the structure, while the third question related to how damping affects the structure. The questions were split to determine which variable, stiffness or damping, the students understood more clearly, and to see how the presentation and activity helped. Questions 1 and 2 would gauge the students' knowledge on stiffness while question 3 would gauge the students understanding of damping. Responses from Questions 1 and 2 in the pre-outreach survey demonstrated that the class had a prior knowledge of affects stiffness has on the structure. 86.75% of students correctly answered question 1 and question 2 on the pre-survey. When the pre-outreach survey percentages were compared to the post-outreach survey percentages there was an increase of 0.29% points from 86.75% before to the 87.04% after. Question 3 of the pre-outreach survey data showed that 50.6% of the class got the question regarding damping correct, compared to the 86.75% on the stiffness questions. The post survey showed an increase from 50.6% to 86.4% or 32 more students who understood how damping would affect a structures response. From these results it was evident that the in class activity coupled with the presentation helped the students better grasp the effects of damping on a structure.

	Students (87 total)	Class1 (33)	Class2(54)
Correct answer	42	22	20
Incorrect answer	45	11	34

Table 4-5 Quiz Results

Table 4-5 splits up the results of Activity 2 by class. Class 1 was considered to have a higher academic standing compared to Class 2. Based on the results in Table 4-5, it showed that approximately 2/3 of the Class 1 students correctly answered the activity problems while only 37% of the students got the activity correct in Class 2.

4.8.1 Fisher's Exact Test

Fisher's exact test was used to determine the statistical significance of the two classes' data from the quiz, and if the results were random or if we should expect the same results going forward. With a significant correlation, the test would prove the contingency between the two variables of interest. If the results of the Fishers Exact Test were significantly correlated, it would prove that the activity is a good representation of the students overall knowledge. In the experiment, Class 1 was the more academically advanced group, where Class 2 was in the lowest math and science classes offered by the middle school.

	# Correct	# Incorrect	Total
Class 1	22	11	33
Class 2	20	34	54
Total	42	45	87

Table 4-6 Fisher Test – Amount Correct/Incorrect

a	b	$a+b$
c	d	$c+d$
$a+c$	$b+d$	N

Table 4-7 Fisher Test – Variable Assignment

After gathering the data, and calculating using

$$p_{outcome} = \frac{(a+b)!(c+d)!(a+c)!(b+d)!}{N!a!b!c!d!} = 0.004959 \quad (6)$$

the formula above, the result of 0.004959 is less than .05 which means the class data shows a significant difference (“Fishers Exact Test”, 2015). Since $p_{outcome}$ was significant, less than .05, it shows that the educational outreach conducted was a good representation of the students’ knowledge on structural stiffness and damping. If this was conducted with a different middle school the same results would be seen.

4.9 Conclusion

After completing the outreach at the Hillside Middle School in Manchester, New Hampshire, it was evident that the students took away some knowledge of stiffness and damping. Damping had the largest jump in result from the pre- to the post-quiz, with an increase from 50% to 86% of students getting the third question correct (Table 4-5). The outreach program proved to be a successful method in teaching damping and stiffness’s effect on a structure while also providing the students with a hands on activity to learn about structural engineering. The outreach also helped determine using more smartphones per student in Activity 1 showed an increase in scores. Students in the afternoon sessions, one phone per student, did 0.3 points better than students who were grouped three students per phone (See Appendix C).

Chapter 5 Thesis Summary

A smartphone application was developed for Android devices using the Java programming language to create a new method of time synchronization and data acquisition for experimental structural analysis. The application has the capabilities to (i) synchronize its time with another smartphone, (ii) record acceleration data and save it to a “.csv” file, (iii) combine more than two “.csv” files into one file, and (iv) animate a virtual structure and its motions on the screen. Ideally the application would be integrated into educational programs to a cost effective method of recording accelerations from multiple devices for classrooms from a K-college level.

The main portion of the application was created to time-synchronize two or more smartphones for use in an experimental SHM system. Incorporating smartphones into engineering methods can assist less affluent programs in teaching structural concepts. Smartphones used Bluetooth to send and receive messages to calculate the time difference between two devices before testing. Testing showed phones could stay synchronized for as long as 45 minutes, however, it is recommended to synchronize devices every 10 to 15 minutes. According to testing conducted, the synchronization method has also shown to have a maximum error of 0.0117 seconds. Using the time synchronization method, and the ability to record acceleration data, it was possible to apply the application to testing an experimental structure.

Testing was conducted on a six-DOF experimental structure in the UNH high bay. Both smartphones and a dedicated sensor system were used to record structural accelerations. Multiple

tests were conducted using different configurations of damage. Damage was applied to the structure by removing cross bracings from the structure. Through post processing, it was evident smartphones could record similar accelerations when compared to dedicated sensor systems. Also, by comparing stiffness values estimated from healthy and damaged structures, it was possible to identify structural damage. Structural damage could only be determined when stiffness change in a story exceeded 10%. Testing proved incorporating smartphones into experimental structural analysis situations could be possible, and improved upon going forward.

An outreach program was developed and conducted at a local middle school to educate students on two structural properties, stiffness and damping. One activity used four smartphones and asked the students to determine which phone's structure had the following combinations of stiffness and damping: (i) high stiffness and high damping, (ii) high stiffness and low damping, (iii) low stiffness and high damping, and (iv) low stiffness and low damping. Pre- and post-outreach survey showed that this smartphone activity positively improved students' interests in STEM (science, technology, engineering and mathematics).

Smartphones, with the programmed application, were capable of synchronizing with multiple devices, recording accelerations, and correctly identifying a change in stiffness of over 10%. The application was also able to be used as a tool to improve upon current educational curriculums. This system could be applied to a school curriculum to replace expensive dedicated sensor systems and provide a greater insight to structures for children.

5.1 Errors and Limitation

Currently there are several aspects to this application and process that hinder its expansion from a lab and experimental usage. From an experimental standpoint, the method of mounting the smartphones using sticky pads could introduce some error. Although the sticky pads were shown to be nearly as effective as attaching the sensors with screws, the phones' alignment to the structure may not be perfectly consistent from test to test. Additionally, if the surface of the sticky pad was not thoroughly cleaned, it could decrease "stickiness" and alter the received accelerations. The method was chosen to be simple to include in a K-12 setting.

From a software point of view, the smartphone application is lacking the ability to record data on set time stamps with an exact time interval between sampling points. Lacking a set sampling rate requires interpolation to a set time stamp and this introduces error because the raw data is not being directly analyzed. Also, with any error in the time synchronization process, it could affect accelerations from a single floor, leading to errors in structural stiffness that could propagate throughout the structure.

The application's synchronization method was determined to have a synchronization error of up to 0.0117 seconds. Applying this error to Microstrain accelerometer data, a 5% error in stiffness value could be seen if the initial stiffness value of the structure is above 200 lbs/in.

Another source of error in this set up is the method that we are using for post processing. After the interpolation is conducted, data is then put into the ERA code, which can compound the errors introduced through the both the synchronization process and the interpolation process.

Overall, the method appears to work well, however, there are clear aspects of the application that need to be improved or studied further when it comes to introducing error.

Currently this method of recording and post processing data for a structural analysis purpose is limited to using in a lab or experimental setup for a K-12 level. It is by no means, at this point in time, a viable alternative to currently accepted methods of SHM. It could be proposed as a cheaper alternative to estimate approximate inter story stiffness values, and enhance structural engineering knowledge in a K-12 classroom setting.

5.2 Future Work

Although the application has proven its capabilities of SHM, it could still be further developed. Currently, the application is programmed to connect each phone in pairs. Although this method is reliable and accurate, it could be made more efficient by altering the code and having it connect and synchronize to multiple devices sequentially. Time synchronization would take significantly less time and make the application easier to use.

A potential capability of the Smartphones could include self location between the master device and the node devices. This would allow the smartphones to identify where they were if laid out in a large scale structure. Some possible ideas would include Bluetooth signal strength or a Global Positioning System

The application currently has the ability to display structures in a three dimensional view. By changing the viewport, the application could display torsion, and acceleration response in the x, y, and z, plane.

Further trials could be conducted testing techniques of securing the smartphones to the structure. Sticky pads are reliable, however, they might not keep the smartphones as still as

necessary. Improving upon the current attachment method could improve results from the smartphones.

Regarding the outreach efforts, it would be ideal to conduct a long term study to track the 7th grade students' STEM interest over time. Continuous, periodic surveys can help address this concern. Additional outreach/testing to other K-12 students will also examine how the developed smartphone application influences students at different age groups.

It would be possible to add some features and additional math to the outreach program. Features that could be added to the application in order to make the in class activity easier, would be to include a timer so the students could time when the structure came to rest. Also, a faint outline of the structure could be laid into the background to show the maximum sways of the structure. This would assist the students in determining structural stiffness. The final addition to the outreach could include some basic structural equations. Introducing the equation " $\omega_n = \sqrt{k/m}$ " into the outreach will allow the students to get a better understanding of structural analysis while keeping the equation at a simple level.

APPENDIX

Appendix A: Eigen-system Realization Algorithm (ERA) with Impulse Excitations

(This appendix is from Dr. Tat Fu's dissertation)

The ERA is developed by Juang and Pappa (1985) and is one of the well-known schemes for estimating modal characteristics. ERA uses singular value decomposition on the Hankel matrix,

$$\mathbf{H}(k-1) = \begin{bmatrix} \mathbf{Y}(k) & \mathbf{Y}(k+1) & \cdots & \mathbf{Y}(k+p) \\ \mathbf{Y}(k+1) & \mathbf{Y}(k+2) & \cdots & \mathbf{Y}(k+p+1) \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{Y}(k+r) & \mathbf{Y}(k+r+1) & \cdots & \mathbf{Y}(k+p+r) \end{bmatrix} \quad (\text{A1})$$

where $\mathbf{Y}(k)$ is the pulse response matrix such that $\mathbf{Y}_{ij}(k)$ is the impulse response at the k^{th} time instant collected at the i^{th} location due to an impulsive excitation at the j^{th} location in the structure. The singular value decomposition of $\mathbf{H}(0)$ is denoted by

$$\mathbf{H}(0) = \mathbf{P}\mathbf{D}\mathbf{Q}^T. \quad (\text{A2})$$

Here, \mathbf{P} and \mathbf{Q}^T are unitary matrices formed by left and right singular vectors respectively and \mathbf{D} is the diagonal matrix formed by the singular values. Singular vectors corresponding to "low" singular values are attributed to noise and the reduced order matrices \mathbf{P}_n , \mathbf{Q}_n and \mathbf{D}_n are generated by using only the singular vectors corresponding to the "high" singular values. The linear system parameters corresponding to the reduced order system can now be estimated using the equations:

$$\mathbf{A} = \mathbf{D}_n^{-1/2} \mathbf{P}_n^T \mathbf{H}(1) \mathbf{Q}_n \mathbf{D}_n^{-1/2} \quad (\text{A3})$$

$$\mathbf{B} = \mathbf{D}_n^{-1/2} \mathbf{Q}_n^T \mathbf{E}_m \quad (\text{A4})$$

$$\mathbf{C}_i = \mathbf{E}_n^T \mathbf{P}_n \mathbf{D}_n^{-1/2} \quad (\text{A5})$$

where $\mathbf{E}_p^T = [\mathbf{I}_p \ \mathbf{0}]$ with \mathbf{I}_p being the identity matrix of order p . The mode shapes of the structure correspond to the columns in the matrix $\mathbf{V} = \mathbf{C}_i \mathbf{\Phi}$, where $\mathbf{\Phi}$ contains the eigenvectors of \mathbf{A} . And the modal frequencies of the structure correspond to the eigenvalues of \mathbf{A} .

Appendix B: Least Square Estimate for Structural Stiffness

(This appendix is from Dr. Tat Fu's dissertation)

After finding the modal parameters of the structure using either ERA or the distributed algorithm, the structural stiffness can be estimated using a least square estimate. In Figure 2, the 4-story shear structure has mass and stiffness matrices

$$\mathbf{M} = \begin{bmatrix} m_1 & 0 & 0 & 0 \\ 0 & m_2 & 0 & 0 \\ 0 & 0 & m_3 & 0 \\ 0 & 0 & 0 & m_4 \end{bmatrix}, \quad \mathbf{K} = \begin{bmatrix} k_1+k_2 & -k_2 & 0 & 0 \\ -k_2 & k_2+k_3 & -k_3 & 0 \\ 0 & -k_3 & k_3+k_4 & -k_4 \\ 0 & 0 & -k_4 & k_4 \end{bmatrix} \quad (\text{B1})$$

By rearranging the eigenvalue problem [B1],

$$(\mathbf{K} - \lambda_j \mathbf{M})\phi_j = 0 \quad \text{or} \quad \mathbf{K}\phi_j = \lambda_j \mathbf{M}\phi_j \quad (\text{B2})$$

with λ_j and ϕ_j being the j^{th} eigenvalue and eigenvector of the system, respectively, (B2) becomes

$$\begin{bmatrix} \phi_{1,j} & \phi_{1,j} - \phi_{2,j} & 0 & 0 \\ 0 & \phi_{2,j} - \phi_{1,j} & \phi_{2,j} - \phi_{3,j} & 0 \\ 0 & 0 & \phi_{3,j} - \phi_{2,j} & \phi_{3,j} - \phi_{4,j} \\ 0 & 0 & 0 & \phi_{4,j} - \phi_{3,j} \end{bmatrix} \begin{bmatrix} k_1 \\ k_2 \\ k_3 \\ k_4 \end{bmatrix} = \begin{bmatrix} \phi_{1,j} \lambda_j m_1 \\ \phi_{2,j} \lambda_j m_2 \\ \phi_{3,j} \lambda_j m_3 \\ \phi_{4,j} \lambda_j m_4 \end{bmatrix} \quad (\text{B3})$$

where $\phi_{i,j}$ is the i^{th} element of ϕ_j . Knowing the mass matrix \mathbf{M} of the structure, the stiffness can be solved for any particular eigenvalue and eigenvector by pre-multiply both sides of (B3) with the inverse of the matrix of ϕ values. Using a least square approach, the overall structural stiffness can be estimated from all the eigenvalues and corresponding eigenvectors computed by ERA or the distributed algorithm.

Appendix C: Multiple Regression Analysis

(This appendix is from a journal paper being prepared by Kyle Wyatt, Drs. Tat Fu and Lewina Lee)

Multiple regression (Allison, 1999) was used to determine if the outreach program influenced students interest in Science, Technology, Engineering, and Math (“STEM interest”), after adjusting for pre-survey interest in STEM and class time (morning vs. afternoon). We also examined whether performance on an in-class smartphone activity was associated in change in STEM interest pre- and post-outreach program.

STEM interest was assessed using a questionnaire item asking, “How interested are you in Science, Technology, Engineering, and Math”, with scores ranging from 0 (“not at all”) to 10 (“very interested”). Pre-outreach STEM interest was measured one day prior to the outreach program, and post-outreach STEM interest was measured immediately after the program. Class time was a binary variable to differentiate between the morning (=0) vs. afternoon class (=1). It was important to adjust for class time because the morning class had 8 phones (i.e., approximately 3 students per phone) whereas the afternoon class had 24 phones (i.e., 1 student per phone). Because students in the afternoon class had more opportunities to interact with the smart phone applications, we expected that they would have a better understanding of structural dynamics concepts, and thus a potentially higher level of STEM interest as a result of the outreach program. Performance on the in-class smartphone activity was used to indicate students’ understanding of structural dynamics concepts. Scores ranged from 0 to 4, with higher scores indicating better performance. Independent variables were entered in three steps. In the first step, pre-outreach STEM interest was entered. This allowed us to examine the residualized change in STEM interest, that is, the magnitude of change in STEM interest after the outreach

program, after adjusting for pre-outreach STEM interest. Class timing was entered in the second step, and performance on the smartphone activity was entered in the final step.

Table C-1 summarizes the findings from the regression analysis. In the first step, students with greater STEM interest during the pre-outreach survey also expressed greater STEM interest during the post-outreach survey. In the second step, students in the afternoon class scored 0.30 points higher on the post-outreach survey than those in the morning class. Difference in class time explained an additional 1.9% in the variance in post-outreach STEM interest. In the third step, better performance on the smartphone activity was linked to greater change in STEM interest after the outreach program and accounted for an additional 1.7% variance in the outcome. Performance on the smartphone activity appeared to explain differences in the morning vs. afternoon classes in their post-outreach STEM interest – after adjusting for performance on the smartphone activity, difference in class time on post-outreach STEM interest was only marginally significant ($p=.07$).

Predicting Post-outreach STEM interest with:	(R^2)	Pre-outreach STEM interest		Class		Phone Activity	
		B (SE)	P	B (SE)	P	B (SE)	P
Step 1: Pre STEM interest	0.7545	.881 (0.053)	<0.0001	--	n/a	n/a	n/a
Step 2: Pre-outreach STEM interest and Class	0.7736	0.8715 (.0501)	<0.0001	0.30183 (.1094)	0.007	n/a	n/a
Step 3: Pre-outreach STEM interest and Phone	0.7831	0.8705 (0.05)	<0.0001	--	n/a	0.14788 (0.0429)	0.0009
Step 4: Pre-outreach STEM interest, Phone, Class	0.7909	0.8658 (0.049)	<0.0001	0.12155 (0.0448)	0.0079	0.20442 (0.1116)	0.0704

B = Parameter Estimate

SE = Standard Error

R^2 = R-Square

Table C-1 Multiple Regression Tests

Appendix D: MATLAB Code

Code below is used to Configuration 1, Test 1, from testing date 03/13/15:

```
clc;
clear all;

%% Developed by Kyle Wyatt
% University of New Hampshire
start = 6;

%% test date 03/13/15
Directory = 'C:\Users\Owner\Documents\Grad School\Thesis\Testing\031315\';

%% 031315 Test 1 - Full structure- Heavy Springs - synced before
% real time sync
timeSync = [0, -3602812, -3936, -4833, -3712, -2544];
% units = ms %[delayPhone1, delayPhone2] %look at the master phones csv to
get these values
test = 'Test1\'; % folder
testTitle = 'Test1';
phone1 = 'p1t103_13_2015_121912.csv';
phone2 = 'p2t103_13_2015_131716.csv';
phone3 = 'p3t103_13_2015_121806.csv';
phone4 = 'p4t103_13_2015_121814.csv';
phone5 = 'p5t103_13_2015_121839.csv';
phone6 = 'p6t103_13_2015_121901.csv';
% int1 = 25 seconds to seconds
    intStart = 25 * 100; % second to start * data points per second
    intEnd = intStart + (60*100); % 60 seconds of data points after the end
    testTitle = ' Test1'; %title of graph

%% code
timeSync = timeSync/1000; % convert tiem differences to seconds

m1 = csvread(strcat(Directory,test,phone1),start,0);
m2 = csvread(strcat(Directory,test,phone2),start,0);
m3 = csvread(strcat(Directory,test,phone3),start,0);
m4 = csvread(strcat(Directory,test,phone4),start,0);
m5 = csvread(strcat(Directory,test,phone5),start,0);
m6 = csvread(strcat(Directory,test,phone6),start,0);

%% Synchronize

t1 = (m1(:,1)-m1(1,1))/1000; % time difference in seconds
t2 = (m2(:,1)-m2(1,1))/1000;
t3 = (m3(:,1)-m3(1,1))/1000;
t4 = (m4(:,1)-m4(1,1))/1000;
t5 = (m5(:,1)-m5(1,1))/1000;
t6 = (m6(:,1)-m6(1,1))/1000;
```



```

% lines up data
t2 = t2 + timeSync(1,2) + (m2(1,1)-m1(1,1))/1000;
t3 = t3 + timeSync(1,3) + (m3(1,1)-m1(1,1))/1000;
t4 = t4 + timeSync(1,4) + (m4(1,1)-m1(1,1))/1000;
t5 = t5 + timeSync(1,5) + (m5(1,1)-m1(1,1))/1000;
t6 = t6 + timeSync(1,6) + (m6(1,1)-m1(1,1))/1000;

%      [time(seconds), x(m/s2), y(m/s2), z(m/s2)]
data1 = [t1, m1(:,2),m1(:,3),m1(:,4)];
data2 = [t2, m2(:,2),m2(:,3),m2(:,4)];
data3 = [t3, m3(:,2),m3(:,3),m3(:,4)];
data4 = [t4, m4(:,2),m4(:,3),m4(:,4)];
data5 = [t5, m5(:,2),m5(:,3),m5(:,4)];
data6 = [t6, m6(:,2),m6(:,3),m6(:,4)];

if intEnd == 0;
    intEnd = length(data5)-1000;
end

data1 = data1(intStart:intEnd,:);
data2 = data2(intStart:intEnd,:);
data3 = data3(intStart:intEnd,:);
data4 = data4(intStart:intEnd,:);
data5 = data5(intStart:intEnd,:);
data6 = data6(intStart:intEnd,:);

% done individually for each data set because the time vectors arent the same
size yet.
%% Data1 - Remove duplicate data points
for ii = 1:(length(data1)-10)
    a=data1(ii,1); %first time
    b=data1(ii+1,1); % second time
    c=b-a; % difference
    data1c(ii+1,1)=c; % all the differences
end
ind1 = find(~data1c); % find the differences that are zero
data1(ind1,:)=[];
%% Data2
for ii = 1:(length(data2)-10)
    a=data2(ii,1); %first time
    b=data2(ii+1,1); % secondtime
    c=b-a; % difference
    data2c(ii+1,1)=c; % all the differences
end
ind2 = find(~data2c); % find the differences that are zero
data2(ind2,:)=[];
%% Data3
for ii = 1:(length(data3)-10)
    a=data3(ii,1); %first time
    b=data3(ii+1,1); % second time
    c=b-a; % difference
    data3c(ii+1,1)=c; % all the differences
end
ind3 = find(~data3c); % find the differences that are zero
data3(ind3,:)=[];
%% Data4

```

```

for ii = 1:(length(data4)-10)
    a=data4(ii,1); %first time
    b=data4(ii+1,1); % second time
    c=b-a; % difference
    data4c(ii+1,1)=c; % all the differences
end
ind4 = find(~data4c); % find the differences that are zero
data4(ind4,:)=[];
%% Data5
for ii = 1:(length(data5)-10)
    a=data5(ii,1); %first time
    b=data5(ii+1,1); % second time
    c=b-a; % difference
    data5c(ii+1,1)=c; % all the differences
end
ind5 = find(~data5c); % find the differences that are zero
data5(ind5,:)=[];
%% Data6
for ii = 1:(length(data6)-10)
    a=data6(ii,1); %first time
    b=data6(ii+1,1); % second time
    c=b-a; % difference
    data6c(ii+1,1)=c; % all the differences
end
ind6 = find(~data6c); % find the differences that are zero
data6(ind6,:)=[];

%% Set the Interval of data points used
% finds the first and last point in each array
% finds the max
firstTimes =
[data1(1,1),data2(1,1),data3(1,1),data4(1,1),data5(1,1),data6(1,1)]
lastTimes =
[data1(end,1),data2(end,1),data3(end,1),data4(end,1),data5(end,1),data6(end,1)
]
[M,I] = max(firstTimes); % finds the latest first start time
[mEnd,iEnd] = min(lastTimes); % finds the first end time

[c, index] = min(abs(data1(:,1)-M)); % finds the closest value to the max
start time found earlier
[cEnd, indexEnd] = min(abs(data1(:,1)-mEnd)); % finds the closest value to
the min end time found earlier
% c = difference from value index = where it is
data1 = data1(index:indexEnd,:);

[c, index] = min(abs(data2(:,1)-M)) ;
[cEnd, indexEnd] = min(abs(data2(:,1)-mEnd));
data2 = data2(index:indexEnd,:);

[c, index] = min(abs(data3(:,1)-M));
[cEnd, indexEnd] = min(abs(data3(:,1)-mEnd));
data3 = data3(index:indexEnd,:);

[c, index] = min(abs(data4(:,1)-M));
[cEnd, indexEnd] = min(abs(data4(:,1)-mEnd));
data4 = data4(index:indexEnd,:);

```

```

[c, index] = min(abs(data5(:,1)-M)) ;
[cEnd, indexEnd] = min(abs(data5(:,1)-mEnd));
data5 = data5(index:indexEnd,:);

[c, index] = min(abs(data6(:,1)-M));
[cEnd, indexEnd] = min(abs(data6(:,1)-mEnd));
data6 = data6(index:indexEnd,:);

lengthData
=length(data1),length(data2),length(data3),length(data4),length(data5),length(data6)];

last=min(lengthData);

% make each data set the same length :); %t,x,y,z
data1 = data1(1:last,:);
data2 = data2(1:last,:);
data3 = data3(1:last,:);
data4 = data4(1:last,:);
data5 = data5(1:last,:);
data6 = data6(1:last,:);

%combine data
dataCombined(:, :, 1) = data1; %row %column %depth`
dataCombined(:, :, 2) = data2;
dataCombined(:, :, 3) = data3;
dataCombined(:, :, 4) = data4;
dataCombined(:, :, 5) = data5;
dataCombined(:, :, 6) = data6;

%% reduce amount of data points
for ii=1:size(dataCombined,3);
    [c, index] = max(abs(dataCombined(:,3,ii)));
    accel_index(ii) = index;
    index
end

start = min(accel_index)-300;%300=3seconds before the impact
last = 2400; %amount of data points after the impact

dataCombinedShift = dataCombined(start:start+(last-1),:,:);

%% Interp the data
Fs = 100; % hz
T = 1/Fs;
interpMethod = 'linear'
%% Phone1
Tstart = ceil(dataCombinedShift(1,1,1)); %rounds up
Tend = floor(dataCombinedShift(end,1,1));
t = (Tstart+.5):T:(Tend-1); % time Vector

X = dataCombinedShift(:,1,1); % time

```

```

Y = dataCombinedShift(:,3,1); %accelerations
dataInterp1 = interp1(X,Y,t,interpMethod);

X = dataCombinedShift(:,1,2); % time
Y = dataCombinedShift(:,3,2); %accelerations
dataInterp2 = interp1(X,Y,t,interpMethod);

X = dataCombinedShift(:,1,3); % time
Y = dataCombinedShift(:,3,3); %accelerations
dataInterp3 = interp1(X,Y,t,interpMethod);

X = dataCombinedShift(:,1,4); % time
Y = dataCombinedShift(:,3,4); %accelerations
dataInterp4 = interp1(X,Y,t,interpMethod);

X = dataCombinedShift(:,1,5); % time
Y = dataCombinedShift(:,3,5); %accelerations
dataInterp5 = interp1(X,Y,t,interpMethod);

X = dataCombinedShift(:,1,6); % time
Y = dataCombinedShift(:,3,6); %accelerations
dataInterp6 = interp1(X,Y,t,interpMethod);

firstInterpTimes =
[dataCombinedShift(1,1,1),dataCombinedShift(1,1,2),dataCombinedShift(1,1,3)..
.
dataCombinedShift(1,1,4),dataCombinedShift(1,1,5),dataCombinedShift(1,1,6)];

interpData(:,1) = dataInterp1;
interpData(:,2) = dataInterp2;
interpData(:,3) = dataInterp3;
interpData(:,4) = dataInterp4;
interpData(:,5) = dataInterp5;
interpData(:,6) = dataInterp6;

%% run ERA Code
dt = 1/Fs;
n = 12;

[sys,pole,mshape,wn,zn,mac] = era(interpData,'T',dt,'N',n);

pole1=pole
pole = log(pole(1:2:end))/dt;
% convert eigenvalues to the correct time domain
mshape = sys.c*mshape(:,1:2:end); %convert eigenvectors as well

f=wn/2/pi

wn2=wn; % allows the stiffness script to still access wn

```

Appendix E: Java Code – MainActivity Class

This is just one class in the application, others available upon request.

```

package com.KWyatt.accelerometer;

import java.io.File;
import java.io.IOException;
import java.util.List;

import android.os.Bundle;
import android.os.Environment;
import android.app.Activity;
import android.content.Intent;
import android.view.Menu;
import android.view.MenuInflater;
import android.view.MenuItem;
import android.view.View;
import android.hardware.Sensor;
import android.hardware.SensorEvent;
import android.hardware.SensorEventListener;
import android.hardware.SensorManager;
import android.widget.Button;
import android.widget.TextView;
import android.widget.Toast;

/**This is the Code for the first screen. Brings up buttons
 * and the X,Y,Z accelerations with the associated timestep
 * RecordData is next
 * Epoch code taken from http://www.epochconverter.com/
 */

public class MainActivity extends Activity implements SensorEventListener

{
    private SensorManager sensorManager;
    //-----declares X, Y, Z Axis and time -----
    TextView xCoord;
    TextView yCoord;
    TextView zCoord;
    TextView timeStep;
    // sensor information
    TextView sensorInfo;
    List<Sensor> ss;
    //-----Strings for Toast commands-----
    String mRecordButton = "Data is now Recording"+"\\n"+"Hit Back to Stop";
    // Used in Toast Commands
    String mAnalyzeData = "Analyze Data Button Hit";// Used in Toast Commands
    String mBTLocator = "Bluetooth Button Hit";// Used in Toast Commands
    String mDoesNothing = "Currently has no functionality";
    // Used in Toast Commands

```

```

String folder = "Accelerometer_App"; // folder that data is stored in.

//-----onCreate Function-----
//-----Calls the Layout "ActivityMain into the APP-----
//-----Also starts up the app functionality-----
@Override
public void onCreate(Bundle savedInstanceState)
{
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main); //finds activity main layout

    File root = Environment.getExternalStorageDirectory();
    if(!root.canWrite())
        try {
            throw new IOException("Cannot Write/Access External
Storage");
        } catch (IOException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }

    File directory = new File (root, folder);
    directory.mkdir();
    // -----creates X,Y,Z axis objects-----
    xCoor=(TextView)findViewById(R.id.xcoor);
    yCoor=(TextView)findViewById(R.id.ycoor);
    zCoor=(TextView)findViewById(R.id.zcoor);
    timeStep=(TextView)findViewById(R.id.timestep);

    sensorInfo=(TextView)findViewById(R.id.sensorInfo);

    //-----Sensormanager-----

    sensorManager=(SensorManager) getSystemService(SENSOR_SERVICE);

    sensorManager.registerListener(this, sensorManager.getDefaultSensor(Sensor.TYPE
_ACCELEROMETER),
        SensorManager.SENSOR_DELAY_NORMAL);
    //can add delay in microseconds

    ss=sensorManager.getSensorList(Sensor.TYPE_ACCELEROMETER);

    for (Sensor s:ss) {
        sensorInfo.append(s.getName()
            + "\n Resolution:" + s.getResolution()
            + "\n Max.Range" + s.getMaximumRange());
    }

    /* More sensor speeds (taken from api docs)
    SENSOR_DELAY_FASTEST get sensor data as fast as possible

```

```

        SENSOR_DELAY_GAME    rate suitable for games
        SENSOR_DELAY_NORMAL  rate (default) suitable for screen orientation
changes
    */
//-----Creates a Button to initialize Activity record data -----
    Button bRecord = (Button) findViewById(R.id.record);
    bRecord.setOnClickListener(new View.OnClickListener()
    {

//-----Sets Functionality of the Button bRecord-----
        @Override
        public void onClick(View v)
        {
            // TODO Auto-generated method stub
            startActivity(new
Intent("com.KWyatt.accelerometer.MakeCSVFile"));
        }
    });

//Allows phones to use BT features Scan, pair, and list paired, become discoverable

    Button bBlueTooth = (Button) findViewById(R.id.connect);
    bBlueTooth.setOnClickListener(new View.OnClickListener()
    {

//-----Sets Functionality of the Button bRecord-----
        @Override
        public void onClick(View v)
        {
            // TODO Auto-generated method stub
            startActivity(new
Intent("com.KWyatt.accelerometer.BluetoothLocator"));
        }
    });

//-----Create combined CSV file from 2 csv files-----
    Button bCombine = (Button) findViewById(R.id.combinecsv);
    bCombine.setOnClickListener(new View.OnClickListener()
    {

//-----Sets Functionality of the Button bGraph-----
        @Override
        public void onClick(View v)
        {
            // TODO Auto-generated method stub
            startActivity(new
Intent("com.KWyatt.accelerometer.CombineMenu"));
        }
    });

//-----Animate structure by test data -----

```

```

        Button bAnalyze = (Button) findViewById(R.id.testfile);
        bAnalyze.setOnClickListener(new View.OnClickListener()
        {

//-----Sets Functionality of the Button bRecord-----
            @Override
            public void onClick(View v)
            {
                // TODO Auto-generated method stub
                startActivity(new
Intent("com.KWyatt.accelerometer.AnimateStructure"));
            }
        });

//---Animate structure by displacement/earthquake- move create a structure here. ----
        Button bStructure = (Button) findViewById(R.id.structure);
        bStructure.setOnClickListener(new View.OnClickListener()
        {

//-----Sets Functionality of the Button bStructure-----
            @Override
            public void onClick(View v)
            {
                // TODO Auto-generated method stub
                startActivity(new
Intent("com.KWyatt.accelerometer.StructureMenu"));
            }
        });

//-----Animate structure by Different C and K values For Baron presentation -----

        Button bBaron = (Button) findViewById(R.id.baronactivity);
        bBaron.setOnClickListener(new View.OnClickListener()
        {

//-----Sets Functionality of the Button bStructure-----
            @Override
            public void onClick(View v)
            {
                // TODO Auto-generated method stub
                startActivity(new
Intent("com.KWyatt.accelerometer.BaronChooseStructure"));
            }
        });

        }//OnCreate

//-----onAccuracy Changed Event-----
        @Override
        public void onAccuracyChanged(Sensor sensor, int accuracy)
        {
            // TODO Auto-generated method stub

```



```

    }

//-----onSensor Changed Event-----
@Override
public void onSensorChanged(SensorEvent event)
{
    if(event.sensor.getType()==Sensor.TYPE_ACCELEROMETER)
    {
        // every time the sensor changes
        long epoch = System.currentTimeMillis();// / 1000
        float x = event.values[0];
        float y = event.values[1];
        float z = event.values[2];

        // display those changes on the screen
        timeStep.setText("EpochTime (ms): " + epoch);
        xCoor.setText("X acceleration: " + x);
        yCoor.setText("Y acceleration: " + y);
        zCoor.setText("Z acceleration: " + z);
    }
}

//-----onPause Event-----
@Override
protected void onPause()
{
    super.onPause();
    sensorManager.unregisterListener(this);
}

//-----onResume Event-----
@Override
protected void onResume()
{
    super.onResume();
    sensorManager.registerListener(this,
        sensorManager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER),
        SensorManager.SENSOR_DELAY_NORMAL);
}

//-----onStop Event-----
@Override
protected void onStop()
{
    super.onStop();
    sensorManager.unregisterListener(this);
}

public boolean onCreateOptionsMenu(Menu menu){
    super.onCreateOptionsMenu(menu);

```

```
        MenuInflater graphMenu = getMenuInflater();
//-----shows which menu we want to inflate-----
        graphMenu.inflate(R.menu.activity_main, menu);
        return true;
    }

    public boolean onOptionsItemSelected(MenuItem item){

        switch (item.getItemId())
        {

            case R.id.menu_settings:
                startActivity(new Intent("com.KWyatt.accelerometer.XAxis"));
                return true;

        }

        return false;
    }
}
```

References

Anil K Chopra, Dynamics of Structures: Theory and Application to Earthquake Engineering, Prentice Hall

ASCE (2010). Minimum design loads for buildings and other structures (ASCE/SEI 7-10), American Society of Civil Engineers, Reston, VA.

Carroll, D. R. (1997). Bridge engineering for the elementary grades. *Journal of Engineering Education*, 86(3), 221-226.

Celebi, M. (2002). Seismic Instrumentation of Buildings (with Emphasis of Federal Buildings). Technical Report No. 0-7460-68170, United States Geological Survey, Menlo Park, CA, USA

Chintalapudi, K., Paek, J., Gnawali, O., Fu, T. S., Dantu, K., Caffrey, J., ... & Masri, S. (2006, April). Structural damage detection and localization using NETSHM. In *Proceedings of the 5th international conference on Information processing in sensor networks* (pp. 475-482). ACM.

Cho, S., Yun, C. B., Lynch, J. P., Zimmerman, A. T., Spencer Jr, B. F., & Nagayama, T. (2008). Smart wireless sensor technology for structural health monitoring of civil structures. *Steel Structures*, 8(4), 267-275.

Faber, M., Unfried, A., Wiebe, E.N., Corn, J., Townsend, L.W. & Collins, T.L., (2013). Student Attitudes toward STEM: The Development of Upper Elementary School and Middle/High School Student Surveys. Proceedings of the 2013 American Society for Engineering Education Annual Conference & Exposition, Atlanta, June 23-16.

Farrar, C., & Worden, K. (2007). An introduction to structural health monitoring. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 303-315. Retrieved May 2, 2015, from <http://rsta.royalsocietypublishing.org/content/365/1851/303.short>

Feng, M., Fukuda, Y., Mizuta, M., & Ozer, E. (2015). Citizen Sensors for SHM: Use of Accelerometer Data from Smartphones. *Sensors*, 15(2), 2980-2998.

Fisher's Exact Test. (2015). Retrieved February 19, 2015, from <http://mathworld.wolfram.com/FishersExactTest.html>

G-Link® -LXRS®. (2015). Retrieved April 22, 2015, from <http://www.microstrain.com/wireless/g-link>

Kotsakos, D., Sakkos, P., Kalogeraki, V., & Gunopulos, D. (2013). SmartMonitor: Using smart devices to perform structural health monitoring. *Proceedings of the VLDB Endowment*, Volume 6(Issue 12), Pages 1282-1285. Retrieved from <http://dl.acm.org/citation.cfm?id=2536296>

Krzyzanowski, P. (2000). Clock Synchronization. *Lectures on distributed systems*, 2002.

LIS3DHMEMS digital output motion sensor ultra low-power high performance 3-axes "nano" accelerometer. (n.d.). Retrieved March 8, 2015, from http://www.st.com/web/catalog/sense_power/FM89/SC444/PF250725?icmp=pf250725_pron_pr_feb2014

Madden, M., Lenhart, A., Duggan, M., Cortesi, S., & Gasser, U. (2013, March 12). Teens and Technology 2013. Retrieved April 15, 2015, from <http://www.pewinternet.org/2013/03/13/teens-and-technology-2013/>

Paul D. Allison. (1999). *Multiple regression: A primer*. Pine Forge Press.

Ransford, M. (2014, April 22). Study: College students not embracing tablets as originally predicted. Retrieved May 1, 2015, from <http://cms.bsu.edu/news/articles/2014/4/students-can-live-without-tablets-but-not-smartphones>

Ping, S. (2003). Delay measurement time synchronization for wireless sensor networks. *Intel Research Berkeley Lab*.

Rice, J. A., & Spencer Jr, B. F. (2008, March). Structural health monitoring sensor development for the Imote2 platform. In *The 15th International Symposium on: Smart Structures and Materials & Nondestructive Evaluation and Health Monitoring* (pp. 693234-693234). International Society for Optics and Photonics.

Symans, M. D. (2000). Introducing middle school students to engineering principles using educational bridge design software. *Journal of Engineering Education*, 89(3), 273-278.

Yu, Y., Han, R., Zhao, X., Mao, X., Hu, W., Jiao, D., ... & Ou, J. (2015). Initial Validation of Mobile-Structural Health Monitoring Method Using Smartphones. *International Journal of Distributed Sensor Networks*, 2015.

Zimmerman, A. T., Shiraishi, M., Swartz, R. A., & Lynch, J. P. (2008). Automated modal parameter estimation by parallel processing within wireless monitoring systems. *Journal of Infrastructure Systems*, 14(1), 102-113.