College of Engineering, Computer Science, and Technology
Department of Mechanical Engineering

ME 3040-01 Experimental Methods

Spring 2022

Design, Prototyping and Testing of a Bluetooth Mountain bike Suspension Telemetry system

By:
Daniel Buonavita-Haag

Supervised by:
Amir Nankali, PhD.

May 13th, 2022

ABSTRACT

Mountain bike suspension has evolved in the last 20 years, from simple coil springs directly mounted on the bike seat, to full suspension bicycles with adjustable dampers and air springs. As a consequence, there is now a need to measure suspension performance to base the adjustments on quantitative information rather than just ride feeling. Commercially available telemetry systems address this need, however the cost to own such a system is in the range of $800 to $2500. The cost in owning a telemetry system does not make it practical for small race teams or even the enthusiast. According to this need, this paper presents an economical modular solution for mountain bike telemetry. This was accomplished by utilizing commercially available electronic components such as a Particle IOT Argon for the DAQ (data acquisition unit) and a Spectra Symbol TSP-L0200-103 membrane linear potentiometer as a resistive type of transducer. These electrical components were mounted to the fork of a mountain bike (front suspension) utilizing a custom designed 3d printed dust proof and water-resistant housing. As the front suspension cycles (compresses) the telemetry system measures displacement as a function of time. By interpreting this data, it is possible to understand dampening characteristics of the suspension system so that adjustments can be made accordingly. As a validation the prototype sensor was compared to a commercial sensor manufactured by Motion industries. By bench testing both and comparing the data using statistical analysis. A null hypothesis (H_0) *was established as the mean difference between the sensors is equal to zero i.e.*

$$H_0: \mu_{Created\, sensor} - \mu_{Commercial\, sensor} = 0$$

Therefore, the alternative to this hypothesis (H_A) *is stated as*

$$H_A: \mu_{Created\, sensor} - \mu_{Commercial\, sensor} \neq 0$$

By conducting a t-Test for two samples assuming unequal variances a p value of .961625 was obtained, therefore using a standard significance level of 5%; $\alpha = .05$ *it was determined that* $p < \alpha$*, the null hypothesis was rejected.*

Table of Contents

1. Introduction

In certain instances, having underdamped or over damped suspension can raise the risk for injury of the mountain bike rider. For this reason, adjusting suspension to be critically damped is of importance. In more demanding respects of mountain biking such as racing having a properly tuned suspension can give you an advantage over your competitor. Telemetry devices are essential to properly tuning suspension. Most telemetry systems work by tracking displacement as a function of time to gain insight on how the suspension system is reacting. These telemetry systems typically have a sensor and a data logger that are placed on the bicycle. However, with wireless technologies becoming more accessible many sensors transmit data wirelessly to a datalogging devices such as a smartphone or laptop. In this paper, a Motion industries enduro fork tracer will be compared to a DIY (Do It Yourself) prototype sensor built using readily available electronics and a 3d printer. Both sensors will be transmitting data wirelessly to a data logging device. The sensors both work similarly and are typically directly mounted to the suspension as shown in Figure 1. As the suspension compresses a carriage that is fixed to the lower leg of the suspension glides along the guide rails and a wiper applies a localized pressure at some distance along the linear potentiometer. The potentiometer will output a voltage that varies with distance. The varying voltage is read by a DAQ, in this case the Particle IO Argon, this data is then transmitted via Bluetooth and logged on an external device such as a smartphone or computer.

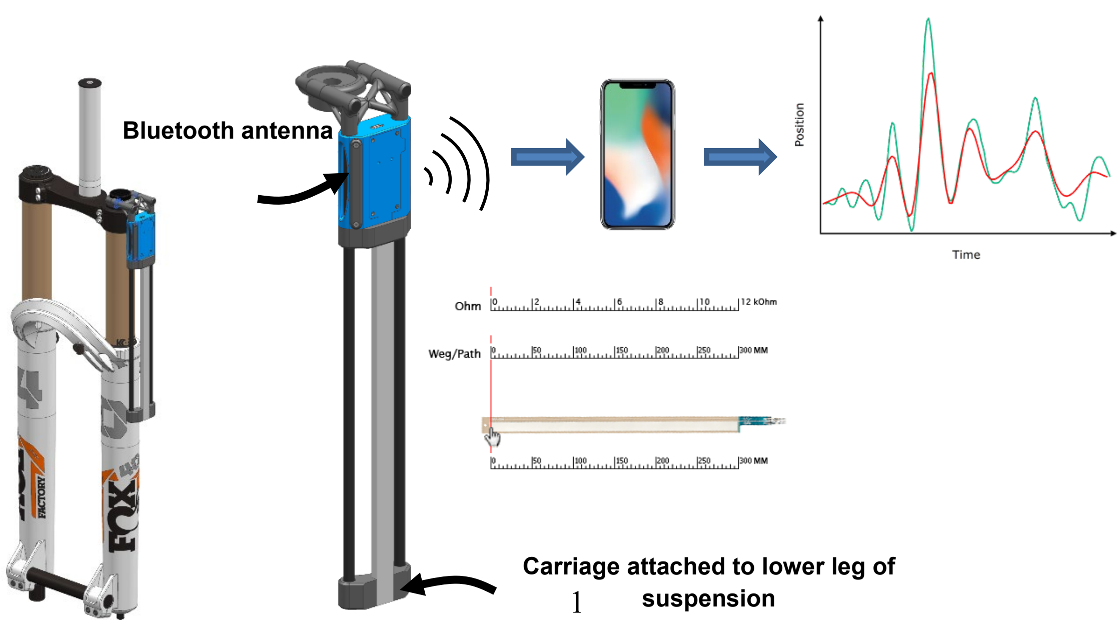

Figure 1 Mountain Bike Telemetry, Prototype Sensor Schematic

2. Prototype Sensor Design

1	Enclosure assy.
2	Mounting bracket assy.
3	Guide rail assy.

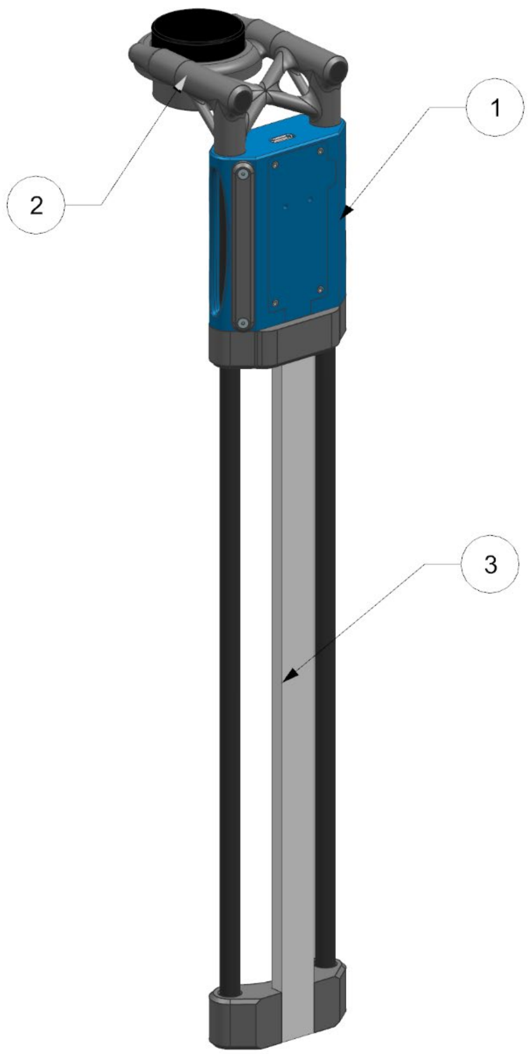

Figure 2 Prototype Sensor Sub-assemblies

The prototype sensor is organized into its three distinctive sub-assemblies labeled in Figure 2. Each sub assembly is comprised of multiple components. Some of the components are custom designed while other components were purchased as raw stock and cut to length or are used for fastening. Further details of each component are addressed in the following sections, enclosure, mounting bracket and guide rail respectively.

I. Enclosure

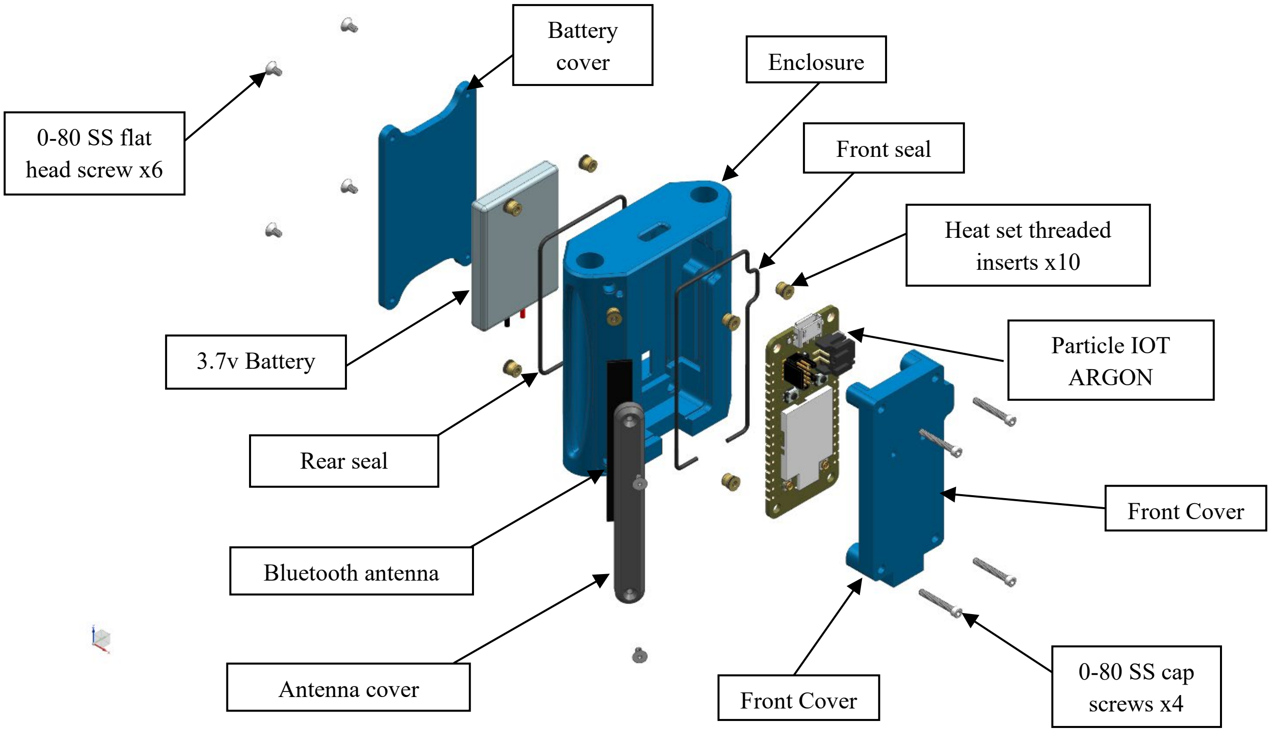

Figure 3 Enclosure assy. exploded view

The electronics enclosure is comprised of 13 unique components labeled in Figure 3. Of the 13 components, the battery cover, enclosure, front cover, and the antenna cover were custom designed to fit the electronics in a dust proof and water-resistant housing. Sealing the device was accomplished by using two seals on the front and rear of the device. On the enclosure a seal gland was designed to accept a .04” diameter EPDM foam rubber O-ring per the design recommendations of the Parker sealing guide [1] to ensure full compression of the seal when the mating piece is attached. The rear seal orientation is such that it functions a “face seal” while the front seal uses a “radial seal” configuration, these seal configurations were done to accommodate the geometrical constraints of the device. Because this device was going to be manufactured using 3d printing technology, designing threads in the body of the device would not be a viable option for fastening. The small size of the device would require the use of 0-80 threads that when printed would fail due to the overwhelming shear stresses from torquing the screw. Therefore, each screw location uses a brass heat set threaded insert as depicted in Figure 3 . The inserts are installed using a soldering iron and placed in a hole designed into the enclosure, as the insert's heat increases the plastic melts and molds around the insert mechanically bonding it to the assembly, thus, metal threads are now available for robust fastening.

II. Mounting Bracket

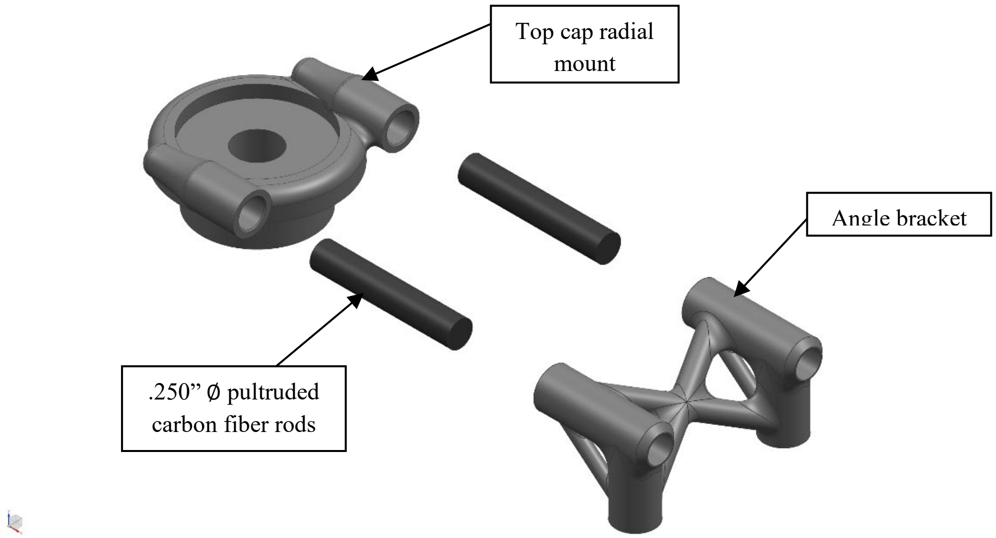

Figure 4 Mounting bracket exploded view

The prototype sensor has been designed to attach to the top air inlet cap of any air spring fork. The attachment configuration is shown in Figure 2. This mounting configuration allows for the telemetry system to be quickly installed and uninstalled by tightening or loosening the top air cap. Due to this radial mount the sensor can be rotated around the leg of the suspension if necessary to avoid becoming obtrusive for tighter trails. In addition, this mounting configuration allows for the modular sensor to be adjusted to different sizes of suspension systems by simply changing the length of the guide rails and transducer. The Top radial mount and angle bracket are both custom designed components for 3d printing, the carbon fiber pultruded rods were purchased and cut to the prescribed length. The components were assembled by bonding the carbon pultruded rods to both 3d printed pieces using Loctite EA9340 epoxy adhesive with 0.0015" glass beads additive as a bond line controller. The entire sub assembly was placed in a heated UV cure chamber for 60 minutes at 60°c and post cured for 90 minutes at 40°c per the manufacture's specifications [2].

III. Guide rails

The guide rail assembly is comprised of 4 components. The carbon fiber rods serve as linear guide rails and the mounting system for the entire sensor assembly. The Spectra Symbol TSP-L0200-103 is an adhesive backed linear membrane potentiometer and is attached to a stock sized high strength 2024 aluminum bar as show in Figure 5. The aluminum bar and sensor components are both mounted to guide rail mounts Figure 5. When fully assembled, the Spectra Symbol TSP-L0200-103 connector is fed into the enclosure and wired to the Particle IOT Argon using 3 connections, voltage in, voltage out and ground seen in Figure 6.

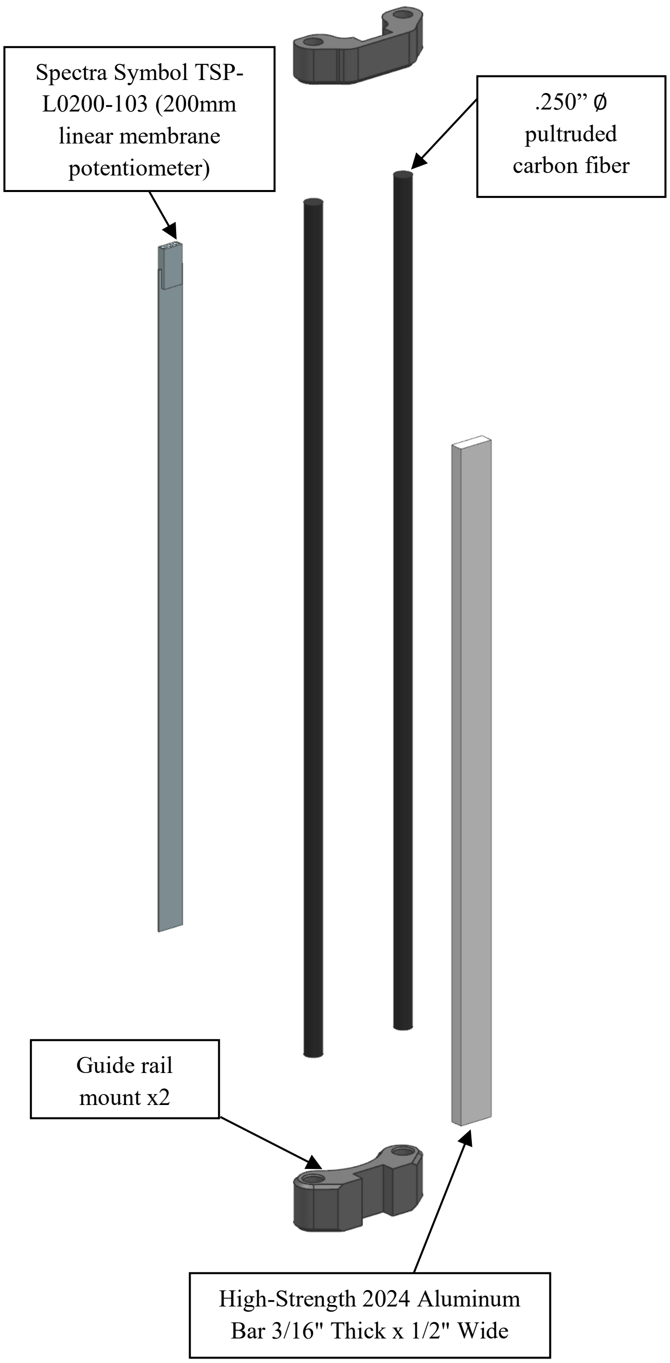

Figure 5 guide rail exploded view

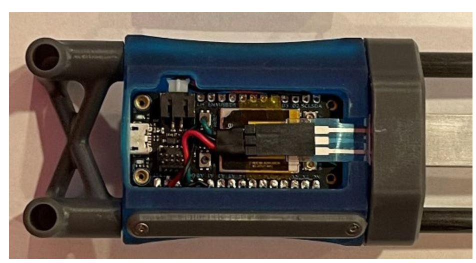

Figure 6 Transducer electrical connections

3. Technical specifications

I. Transducer electrical specifications

Table 1 Spectra Symbol TSP-L0200-103

Natural Resistance	10k Ohms
Resistance Tolerance	+/- 20%
Effective Electrical Travel	8-2000mm
Linearity (independent)	+/- 1%
Repeatability	No hysteresis, dependent on wiper
Resolution	Analog output, dependent on DAQ

II. Calibration procedure

Using the technical specifications provided by the manufacture for the Symbol TSP-L0200-103, it is assumed that the transducers linearity as seen in Table 1 has negligible influence on the recorded data. Therefore, the calibration procedure chosen for the prototype sensor is a one-point calibration. The calibration was performed by setting the carriage of the device to a "zero point" i.e., the suspension of the bicycle would be unloaded or at static equilibrium. At the zero point, a corresponding voltage is read and recorded. The voltage is recorded as a "reference low value" and the distance mapped to this voltage is set as 0mm. The "reference high point" is simply the maximum displacement of the suspension also known as the maximum travel. Using the proportionality of the distance and voltage, a distance as a function of voltage can be derived, this is shown visually in Figure 7.

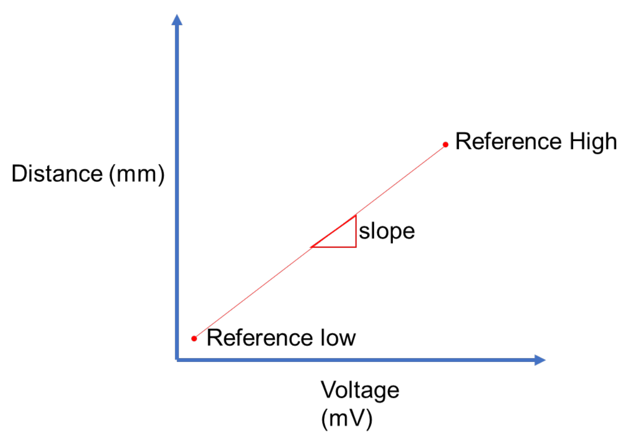

Figure 7 Graphical representation of calibration function

Similarly, the calibration procedure for the commercial sensor prompts the user via their mobile app to "unweight the suspension" and records the zero point. The app then prompts the user to specify the maximum travel of the suspension being measured.

4. Manufacturing and Cost analysis

I. Manufacturing

To manufacture the custom designed sensor housing components, Stereolithography (SLA) additive manufacturing process was used. SLA manufacturing was used due to its high-resolution and broad engineering material choice that was required for this application. The machine used for manufacturing was the FormLabs Form3b+ shown in Figure 8

Figure 8 FormLabs Form3b+ medical

II. Cost analysis (Bill of materials)

Table 2 Itemized Bill of Materials Costs

Component	Cost (USD)
Enclosure	4.40
Battery cover	0.69
Front cover	1.15
Antenna cover	0.47
Top cap radial mount	2.44
Angle bracket	1.38
Guide rail mount x2	3.6
Symbol TSP-L0200-103	19.39
Particle IOT Argon	36.19
3.7v Battery	7.39
.250” Ø pultruded carbon fiber rod	34.98
High-Strength 2024 Aluminum Bar	8.81
0-80 SS cap screws x4	1.12
0-80 SS flat head screw x6	1.28
Heat set threaded inserts x10	1.55
Viton® Fluoroelastomer O-Ring gasket	1.24
TOTAL	**126.08**

From Table 2, each component is itemized with its individual cost to manufacture or to purchase. This analysis is just material cost, it does not include labor, energy, or machine costs. In comparison to the Motion industries enduro fork tracer priced at $799.99, the prototype device is $672.92 less expensive. In addition, the Motion Industries device requires a subscription of $30/month to use the iPhone app to export the data. This is completely avoided with the prototype device that is a one-time cost.

5. Software

I. Device Firmware

The firmware for the device was written in C++ using the Particle IO plugin for visual studio code. Using this plug-in allowed for wireless firmware flashing over Wi-Fi and the use of the Bluetooth low energy 5.0 (BLE) protocols. Due to the BLE protocol limitations the firmware was designed to stream data at a maximum sampling rate of 200hz. The other hardware limitation encountered when designing the firmware was tracking time. The Particle IO device does not have hardware to accurately measure time, therefore time tracking was recorded when the data was received by the data logging device.

II. Data logging and visualization

Because the Particle IO Argon does not have sufficient onboard memory it is merely used as a data transceiver. All of the data was received by a M1 MacBook pro via a HTML web app inside of google chrome web browser with a web BLE plug in. The Html web app serves as a GUI (graphical user interface) that shows the data points on a Distance vs Time graph that updates live as seen in Figure 9. The chart was created using JavaScript open-source charts from Chart.js [3]. The green loading bar at the top is a model of the potentiometer and displays live at which position the potentiometer is experiencing a pressure.

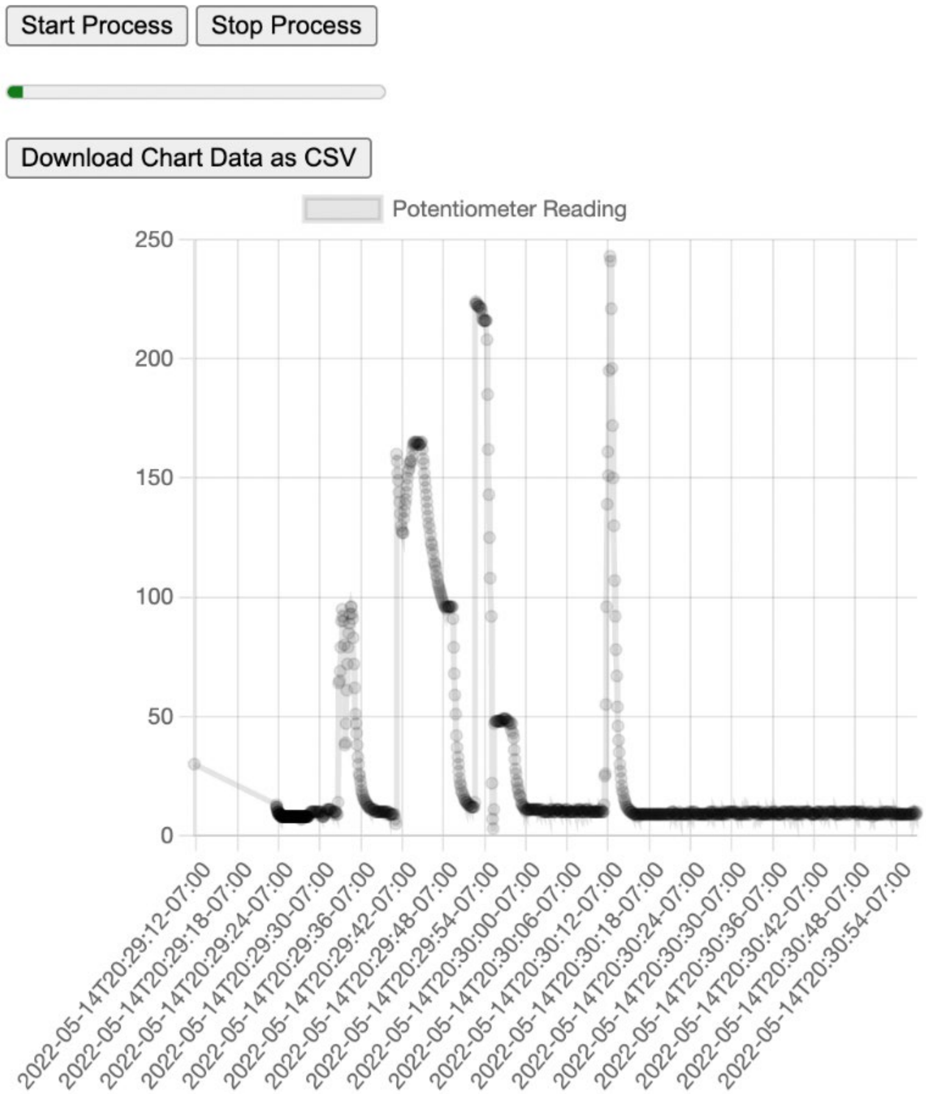

Figure 9 Html web app GUI

6. Experimental Setup

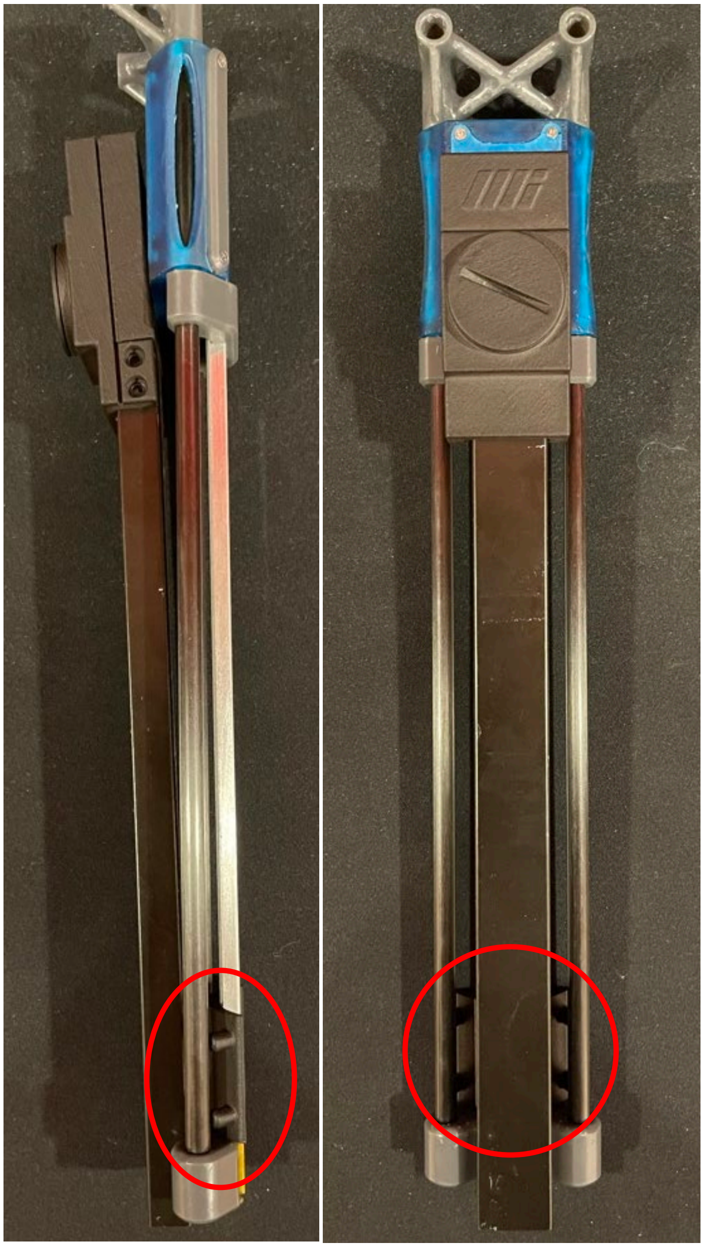

Figure 10 Experimental setup both devices

To compare the two devices a custom carriage and wiper was made to constrain both devices along their guide rails seen highlighted by the red circle in Figure 10 Experimental setup both devices. The wiper was sandwiched between both devices but was able to translate axially along the guide rails as shown in Figure 10. However, due to the thickness of each devices electronic enclosure this caused a slight deviation in parallelism between the transducers of each device. Therefore, as the wiper translated towards the enclosure it would lift away from the prototype device. This caused some anomalies during data collection that will be further discussed in the Results and Discussion section of this paper. Due to time limitations a proper experimental fixture was not created, however to further test and validate the prototype device a more robust method of constraining the devices together is required to avoid any systemic errors in the data as well as any hysteresis caused by inadequate wiper contact with the transducer. Both devices were set to record data simultaneously while the wiper carriage assembly was manually translated up and down the guide rails for approximately 25 seconds. In the 25 seconds of data collection 4660 data points were collected.

7. Results and Discussion

Once the data was collected it was exported in a CSV format and then imported into Microsoft excel. Both data sets were synced in time to correct for the difference in initially setting up the recording. As a preliminary analysis the two data sets were plotted on a Displacement vs Time graph seen in Figure 11. The data plotted shows a generally similar agreement between the two sensors. However, there is a systemic error or offset between both data sets. The offset was determined to occur due to a shift between both sensors after calibration of approximately 3-4mm and continued shifting during the analysis by 1mm. This occurred because of the experimental setup of both sensors. The sensors were not properly constrained to each other and only constrained by the carriage wiper assembly. The Prototype sensor depicted in orange on Figure 11 has some random error in comparison to the commercial sensor, this random error or noise has an undetermined cause, however it is likely that the commercial sensor has some hardware filtering that cleans the signal before transmitting the data where the prototype sensor does not. As previously stated, the lack of parallelism between the two transducers caused the wiper to lift off of the prototype sensor which in turn recorded some erroneous data points seen in Figure 11 at approximately 15 seconds into data collection. The data recorded in this interval is the natural resistance of the device without pressure of the wiper which translates to a distance value over 160mm therefore was not displayed. Using the data visualization tool during the recording process allowed for the correction of this by squeezing the two sensors together to ensure pressure on both transducers and thus data values normalized. Both sensors were further compared by checking proportionality of the displacement data between both sensors in Figure 12. The expectation for Figure 12 is that both data sets will have a proportionality indicating the data sets agree with each other. From Figure 12 there is generally a good agreement however the outliers caused during the data collection anomaly skews the trend line to have an R^2 value of 0.78. Visually the proportionality of the data sets agree, therefore further analysis of the data without the outliers would improve the proportionality.

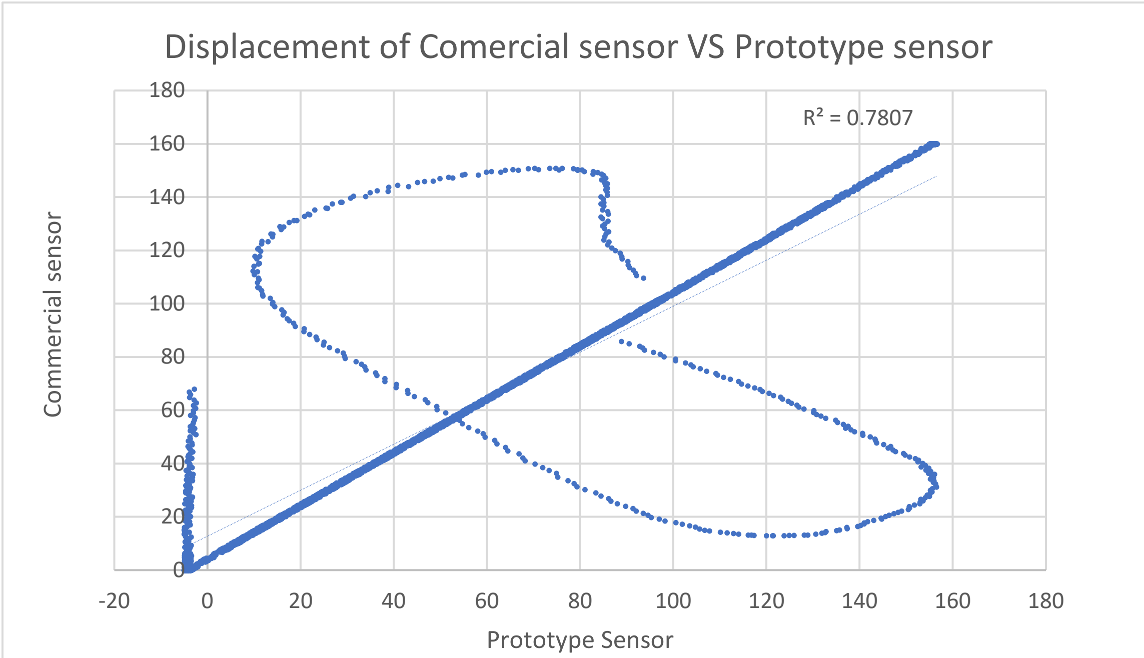

Figure 11 Proportionality of commercial sensor data vs prototype sensor data

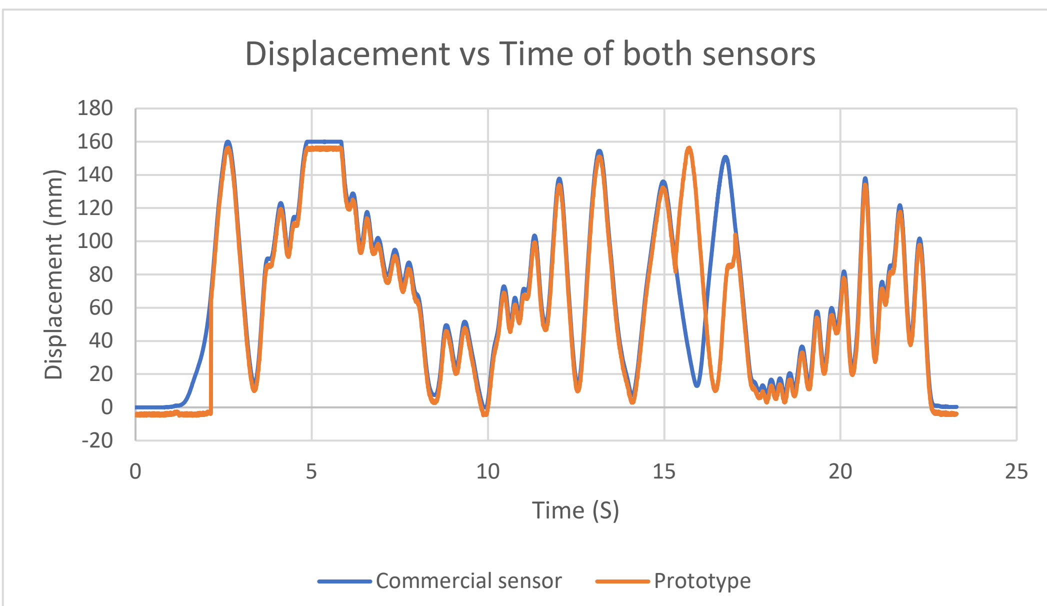

Figure 12 Displacement vs Time comparison of both sensors

8. Statistical analysis

I. Hypothesis

To align with the objective of this paper, a null hypothesis is stated as the mean difference between the sensors is equal to zero. Analytically the null hypothesis is stated as follows.

$$H_0: \mu_{Prototype\,sensor} - \mu_{Commercial\,sensor} = 0$$

$$H_A: \mu_{Created\,sensor} - \mu_{Commercial\,sensor} \neq 0$$

Therefore, the alternative hypothesis is stated as the mean difference between the sensors is not equal to zero.

II. Data distribution

Before hypothesis testing can be conducted it is required to analyze how the data is distributed so that a proper hypothesis test may be selected. Hence, two quantile-quantile (QQ) plots were constructed to visualize the data sets to check for normal distribution. Figure 13 is the QQ plot for the Motion industries enduro tracer and Figure 14 is the QQ plot for the Prototype sensor. Both plots have heavy tails due to the quantity of data points that were taken during the initialization of the data recording and the end of the recording where the data was a constant value. Ignoring the tails as non-significant data points a linear regression of the QQ plots was done to check for proportionality. If the trend line is proportional with a slope of 1 it can be determined that the data is normally distributed. Both plots have linearity with a R^2 value of .94 with slight deviations due to outliers thus both data sets are determined as normally distributed.

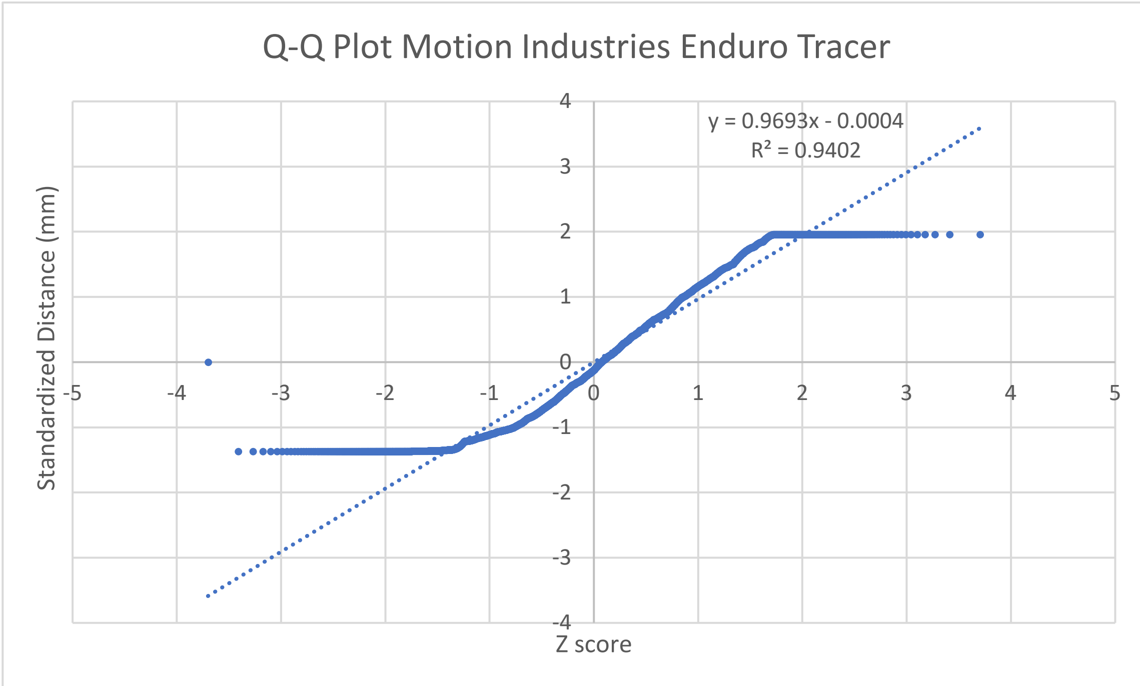

Figure 14 Quantile-Quantile plot for motion industries enduro tracer

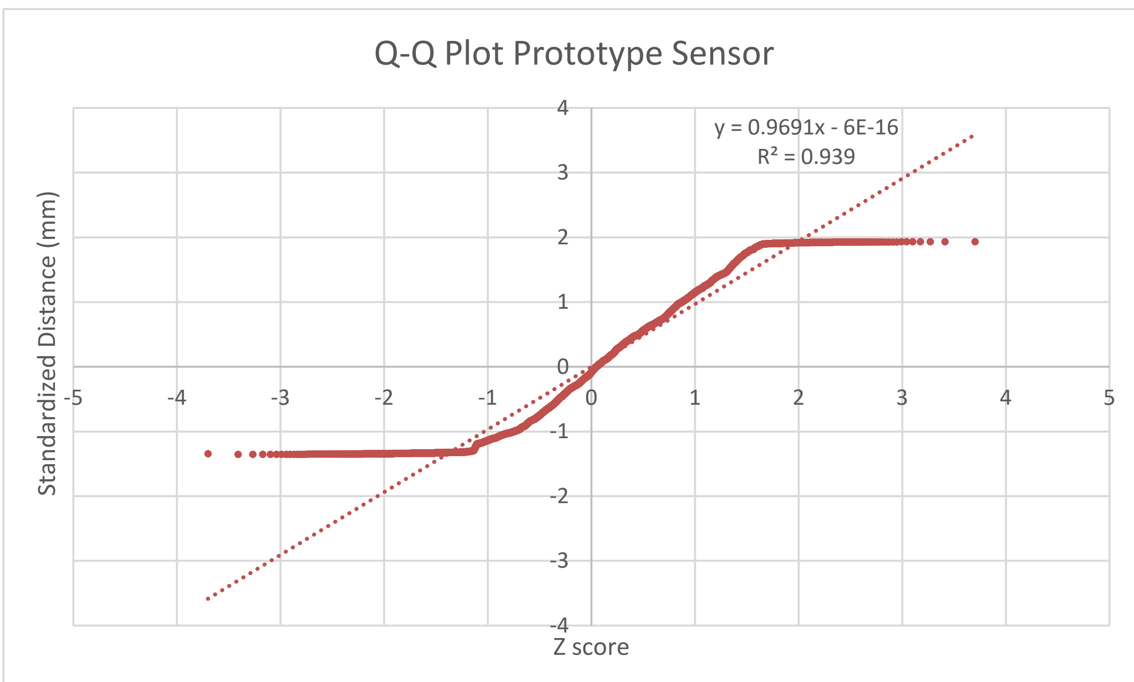

Figure 13 Quantile-Quantile plot for prototype sensor

III. Hypothesis testing

Using the aforementioned parameters for the hypothesis and data distribution it was determined that a paired T-test for means would be conducted. Given by equation 1

$$t = \frac{\sum d}{\sqrt{\frac{n(\sum d^2)-(\sum d^2)}{n-1}}} \qquad (1)$$

Table 3 Paired t-Test for means

t-Test: Paired Two Sample for Means		
	Variable 1	*Variable 2*
Mean	65.80626342	61.48601426
Variance	2312.655737	2423.008608
Observations	4660	4660
Pearson Correlation	0.883579309	
Hypothesized Mean Difference	0	
df	4659	
t Stat	12.54726759	
P(T<=t) one-tail	7.68082E-36	
t Critical one-tail	1.645180752	
P(T<=t) two-tail	1.53616E-35	
t Critical two-tail	1.960473295	

Using the paired T-test it was determined that the P value for the two tailed test was 1.53616E-35 as shown in Table 3. Consequently, $P < \alpha$ and the null hypothesis must be rejected.

9. Conclusion

Using readily available materials and electronics it was possible to design and manufacture a DIY wireless suspension telemetry system for $672.92 less than the leading competitor. To validate the sensor an experimental fixture was created to constrain both sensors to each other to simultaneously measure paired data values. These data sets were exported and compared using Microsoft excel. The data was then tested for normal distribution and a statistical analysis was performed. The results from the statistical analysis determined that $P < \alpha$ and the null hypothesis was rejected. Hence, the prototype sensor population mean was less than the standard significance level α of .05. The results of the statistical analysis were likely heavily influenced from the offset that occurred due to the shifting of the experimental setup mid data collection. In addition, an anomaly occurred due to a deviation in parallelism during the data collection causing a section of data to be erroneous. The prototype sensor also deviated with some random error in comparison to the Motion industries sensor. Further experiments with an improved experimental setup would address these deviations as well as creating a filter to remove some of the random error. With these improvements it may be possible to retest the null hypothesis for a more accurate comparison of the sensors.

10. References

[1] “Seal solutions guide - parker hannifin.” [Online]. Available: https://promo.parker.com/promotionsite/seal-solutions-guide/us/en/home/Dynamic-Sealing. [Accessed: 11-May-2022].

[2] Yumpu.com, “TDS LOCTITEÂ® Hysol - Scott Technical Sales Company,” *yumpu.com*. [Online]. Available: https://www.yumpu.com/en/document/view/43484845/tds-loctitear-hysol-scott-technical-sales-company. [Accessed: 16-May-2022].

[3] “Chart.js,” *Chart.js | Open source HTML5 Charts for your website*. [Online]. Available: https://www.chartjs.org/. [Accessed: 16-May-2022].

[4] *Ble potentiometer example*. [Online]. Available: https://rickkas7.github.io/ble-potentiometer/. [Accessed: 11-May-2022].

[5] D. Loginov, “How to make a web app for your own Bluetooth Low Energy Device?,” *Medium*, 07-Jun-2019. [Online]. Available: https://loginov-rocks.medium.com/how-to-make-a-web-app-for-your-own-bluetooth-low-energy-device-arduino-2af8d16fdbe8. [Accessed: 11-May-2022].

[6] *Sensor tip continued*. [Online]. Available: https://www.novotechnik.com/newsletters/issue2/sensor-tip-cont.html. [Accessed: 11-May-2022].

[7] Unknown, *Uncertainty calculation*, 01-Jan-1970. [Online]. Available: http://jaufattengineering.blogspot.com/2012/10/just-let-you-know-how-to-do-uncertainty.html. [Accessed: 11-May-2022].

11. Appendix

```
#include "Particle.h"

SYSTEM_MODE(MANUAL);


const BleUuid serviceUuid("4677062c-ad02-4034-9abf-98581772427c");
const BleUuid valueUuid("dc13b36a-3499-46b0-ac11-5ac0173c4cc5");

BleCharacteristic valueCharacteristic("value",
BleCharacteristicProperty::NOTIFY, valueUuid, serviceUuid);

const int ADC_PIN = A1;
const unsigned long UPDATE_PERIOD_MS = 100;
unsigned long lastUpdate = 0;

void setup() {
    Serial.begin();

    BLE.addCharacteristic(valueCharacteristic);

    BleAdvertisingData data;
    data.appendServiceUUID(serviceUuid);
    BLE.advertise(&data);
}

void loop() {
    if (BLE.connected()) {
        if (millis() - lastUpdate >= UPDATE_PERIOD_MS) {
            lastUpdate = millis();

            uint8_t data[1];

            // ADC value is 0-4095 (12 bits) but only put 8 unsigned bits
in the data
            // as we don't need the resolution and it avoid dealing with
byte order issues.
            data[0] = analogRead(ADC_PIN) >> 4;

            valueCharacteristic.setValue(data, sizeof(data));
        }
    }
}
```

Figure 15 Particle IOT firmwa

```html
<html>

<head>
  <title>BLE Potentiometer Example</title>
  <style>
    body {
      background-color: #ffffff;
    }

    p,
    h1,
    h2,
    h3,
    li,
    td,
    div {
      font-family: helvetica, sans-serif;
      color: #000000;
      font-size: medium;
    }
  </style>
  <script src="https://cdn.jsdelivr.net/npm/chart.js@^3"></script>
  <script src="https://cdn.jsdelivr.net/npm/moment@^2"></script>
  <script src="https://cdn.jsdelivr.net/npm/chartjs-adapter-moment@^1"></script>
</head>

<body>

  <button>Start Process</button>
  <button id="stop">Stop Process</button>

  <div id="noBLE" style="display:none">
    This demo requires Web BLE, which is only available on new Chrome browsers on Android Chrome,
Chromeboxes,
    some Macs and Windows PCs. It's not supported on other browsers (Firefox, Safari, Edge, Internet
Explorer)
    and is not supported on Chrome for iOS.
  </div>

  <div id="error"></div>

  <p><meter id="meter" value="0" min="0" max="255" style="width:200"></meter></p>

  <div>
    <button id="downloadCSV">Download Chart Data as CSV</button>
    <canvas id="myChart" width="400" height="400"></canvas>
  </div>
```

```
<script>
    var device;

    var chartRef = document.getElementById('myChart').getContext('2d');

    var chart = new Chart(chartRef, {
        type: 'line',
        data: {
            datasets: [
                {
                    label: 'Potentiometer Reading',
                    data: [
                        {
                            x: Date.now(),
                            y: 30
                        }
                    ]
                }
            ]
        },
        options: {
            scales: {
                x: {
                    type: 'time',
                    time: {
                        unit: 'seconds'
                    }
                }
            }
        }
    })
    // chart.options.animation = false;

    document.querySelector('button').addEventListener('click', function () {
        if (navigator.bluetooth) {
            onButtonClick();
        }
        else {
            document.getElementById('noBLE').style.removeProperty('display');
        }
    });

    document.querySelector('#stop').addEventListener('click', function(){
        disconnectFromDevice();
    })

    async function onButtonClick() {
        try {
```

```
        console.log('requesting bluetooth device...');
        device = await navigator.bluetooth.requestDevice({
            filters: [{ services: ['4677062c-ad02-4034-9abf-98581772427c'] }]
        });

        console.log('connecting to GATT server...');
        const server = await device.gatt.connect();

        console.log('getting private potentiometer service...');
        const service = await server.getPrimaryService('4677062c-ad02-4034-9abf-98581772427c');

        console.log('getting private potentiometer characteristic...');
        const characteristic = await service.getCharacteristic('dc13b36a-3499-46b0-ac11-5ac0173c4cc5');

        console.log('starting notifications...');
        await characteristic.startNotifications();

        console.log('notifications started, adding listener');
        characteristic.addEventListener('characteristicvaluechanged', handleNotifications);
        // device.addEventListener('gattserverdisconnected', onDisconnected);

    } catch (error) {
        console.log('error: ' + error);
        document.getElementById('error').innerHTML = error;
    }
}



function handleNotifications(event) {
    var value = event.target.value.getUint8(0);

    // chart.data.labels.push(Date.now());
    chart.data.datasets.forEach((dataset) => {
            dataset.data.push({
                x: Date.now(),
                y: value
            })
    });
    chart.update();

    console.log(`${Date.now()},${value}`);
    document.getElementById('meter').value = value.toString();
}

function disconnectFromDevice() {
    device.gatt.disconnect();
    console.log('Disconnected from device.')
```

```
    }
</script>

<script>

document.getElementById("downloadCSV").addEventListener("click", function(){
    downloadCSV({ filename: "chart-data.csv", chart: chart })
});

function convertChartDataToCSV(args) {
    var result, columnDelimiter, lineDelimiter, data;

    data = args.data || null;
    if (data == null || !data.length) {
        return null;
    }

    columnDelimiter = args.columnDelimiter || ',';
    lineDelimiter = args.lineDelimiter || '\n';

    result = `time${columnDelimiter}value`;
    result += lineDelimiter;

    data.forEach(function(item) {
        result += item.x;
        result += columnDelimiter;
        result += item.y;
        result += lineDelimiter;
    });

    return result;
}

function downloadCSV(args) {
    var data, filename, link;
    var csv = "";


    for(var i = 0; i < args.chart.data.datasets[0].data.length; i++){
        csv += convertChartDataToCSV({
            data: args.chart.data.datasets[0].data
        });
    }
    if (csv == null) return;

    filename = args.filename || 'chart-data.csv';

    if (!csv.match(/^data:text\/csv/i)) {
```

```
            csv = 'data:text/csv;charset=utf-8,' + csv;
        }

        data = encodeURI(csv);
        link = document.createElement('a');
        link.setAttribute('href', data);
        link.setAttribute('download', filename);
        document.body.appendChild(link); // Required for FF
        link.click();
        document.body.removeChild(link);
    }
    </script>
</body>

</html>
```

Figure 16 HTML web app