CALIFORNIA STATE UNIVERSITY, NORTHRIDGE

Sensor Fingerprinting of Mobile Devices

A thesis submitted in partial fulfilment of the requirements

For the degree of Master of Science in Computer Science

By

Oscar Casarez-Ruiz

May 2021

The thesis of Oscar Casarez-Ruiz is approved:

_____	_____
Li Liu, PhD	Date
_____	_____
Taehyung Wang, PhD	Date
_____	_____
Ani Nahapetian, PhD, Chair	Date

California State University Northridge

Table of Contents

List of Figures

Abstract

Sensor Fingerprinting of Mobile Devices

By

Oscar Casarez Ruiz

Master of Science in Computer Science

This thesis explores the use of motion sensor data that is available on mobile devices to create a device fingerprint, which can be used to identify the mobile device. In the experimentation, the mobile device sensor data is accessed with an Android application from both a series of 14 smart phones and a series of 4 smart watches. Once the data is collected and processed, the sensor fingerprint is compared with existing data to determine if the device has been previously identified. For the experimentation, both a controlled study--where the mobile device is placed in a consistent position-- and a more realistic study--where the mobile device is stationary but in various positions-- was completed. Each time an experiment is conducted on a mobile device, the application processes the data and returns the sensor fingerprint of the device that is the closest match. The results from the experiments show that sensor data can be used to identify a device.

Smartphones provide more consistent data compared to smartwatches since watches are worn on the arm. Also, the position of the mobile device, particularly with respect to the rotation along the z-axis impacts the classification the accuracy of the identification algorithm.

1. Introduction

Sensor data collected from mobile devices can be used for more than adding functionality, it can also be used to create a sensor fingerprint to identify the device. Mobile devices provide access to sensor data to all mobile applications and web browsers without any permissions, thus allowing for sensor fingerprinting to go undetected by the owner of the device. This thesis explores techniques to convert sensor data into a sensor fingerprint and apply the sensor fingerprint to identify devices.

Device fingerprinting is a common way to identify a device without requesting direct permission to sensitive device information. For mobile devices there are unique identifiers such as serial numbers or international mobile equipment identity, however, they are guarded by permissions. This information about the device, including location, device model, and operating system can be spoofed, thus making sensor data-based fingerprinting a better approach. The sensors data that is on the mobile devices cannot be manipulated it, which allows for a reliable source for creating a device fingerprint. In fact, when a mobile device is factory reset, the sensors are not affected. The sensors continue to provide the same data making the sensor fingerprint persistent.

The intention behind identifying a device with sensor data can be used for innocent or malicious purposes. The sensor data is a powerful source of data that can be collected without the owner of the device noticing. Once the data from a device has been collected it can be used to track the device, regardless of if the owner resets their mobile devices, clears their website history or if they use incognito web browsers. The sensor fingerprint can be used across web browsers and mobile applications.

Sensor fingerprinting can be used as a solution to software metering or authenticating users. For software metering, using a sensor fingerprint allows the application to identify the device preventing users from exploiting free trials by creating new accounts. Including a sensor fingerprint while authenticating a user will add another level of security by preventing logins from devices not recognized.

A system for device identification, called SOFIA, was developed, and its classification accuracy was evaluated. Also, this thesis considers countermeasures for sensor fingerprinting, including varying the positioning of device or reducing the precision of the sensor reading.

Sensor device fingerprinting is only one method for identification, in Chapter 2, additional methods of device identification are mentioned. In Chapter 3, the threat model provides an analyze on how a malicious application can use sensor fingerprinting to exploit mobile devices. Chapter 4 provides a background on what sensors are on mobile devices and which ones can be used for identification. Chapter 5 provides initial data collected from accelerometer sensors from mobile devices, including an analysis on how the sensor data is distinct between devices. Chapter 6 includes details about a mobile application used to collect data and sensor fingerprinting process. Chapter 7 provides the data collected, results from experimenting on multiple mobile devices using a mobile application prototype, and countermeasures. Chapter 8 concludes the thesis.

2. Related Work

Users online have noticed that they receive online ads based on their recent activity. One of the ways this is possible is through device fingerprinting.

2.1. Web Browser Fingerprinting

Device fingerprinting identifies user's devices without traditional methods such as with cookies or with IP address tracking [10], across browsing sessions [7] and even across browsers [4]. IP addresses from devices are typically recycled and used for long periods of time. This allows for web servers to retain IP addresses per device. This technique is often overlooked in comparison to using cookies but can be prevented by using a (VPN). Most used web browsers like Google Chrome, Firefox and Safari do not natively protect against IP address tracking. For tracking across browser sessions and browsers, trackers generate a unique identifier which is stored in multiple storage locations to evade being deleted by users.

Browser fingerprinting is another technique that does not require storing information collected from users, because it is completely stateless. Typically, a script in the browser is run to collect an extensive variety of information available using public APIs and HTTP headers [8]. A browser fingerprint can contain the following information but is not limited to the list of plugins, cookies enabled, time zone, list of fonts, WebGL Vender and WebGL Renderer. Users may believe that using multiple browsers can help prevent third-party trackers from learning about their usage patterns, but the browser fingerprint also collects OS and hardware information. The OS and hardware information are enough to identify users, even if they use multiple browsers.

2.2. Speaker-Microphone and Accelerometer Fingerprinting

Research conducted at Stanford University [1] demonstrated two methods to fingerprint smartphones by using two sensors, the speakerphone-microphone and accelerometer. Using the speakerphone and microphone system on smartphones, the researchers relied on the fact that manufacturing cannot produce identical microphones and loudspeakers, which results in inconsistencies in frequency. As a result, they were able to fingerprint smartphones by playing a series of sounds emitted at different frequencies which are then recorded by microphone. The amplitude and the distortion in the frequency are measured from the recorded signals which provides a fingerprint. Of course, this is intrusive to the user as they need to allow the application permission to use the microphone in order to record the sounds.

The second sensor used was an accelerometer, the key was to identify the imperfections caused by calibration during production which cause scaling and translation in the measurements. These imperfections are unique to each device and were estimated by using an optimization problem with six values, two for each dimension, in order to generate the fingerprint. These measurements can be collected without any action from the user and since the accelerometer is not considered sensitive by Android there is no way to limit via permissions. The results from the mobile web page experiment showcased that the accelerometer data collected from 3583 devices that provide two submissions and the type of device used, the system correctly identified 53% of the device [1]. Along with a microphone-speaker and accelerometer sensor, there are 38 implicit identifiers that can be acquired without requesting any permission [11] in Android devices. There exist unique identifiers on Android phones like International mobile equipment identity (IMEI) or Android ID, however, some of the information can easily be tampered with such as the Android ID. Such

identifiers also require access to sensitive data which can potentially be abused and leaked, violating user privacy.

2.3. Camera Sensor Fingerprinting

In addition to mobile devices, cameras can also be fingerprinted [5]. When a photo is saved from a digital camera, there is information such as exposure, date, time and camera type. Although time information is generic to most cameras, once paired with a fingerprint of the camera sensor pattern noise researchers and particularly investigators can better analyze forensic evidence. The sensor pattern noise is composed of the small changes in intensity between individual pixels when exposed to a well-lit environment. The noise of the first photo is measured, and a shot noise is calculated, if multiple photos are then taken in approximately the same area the pattern noise should be present in each.

2.4. Smart Device Fingerprinting

There has been an increase in demand for smart home devices or Internet-of-Things (IoT) devices, which are also vulnerable to be device fingerprinted [2]. Typically, most of these IoT devices are connected and have internet access via Bluetooth Low Energy (BLE) which is well-suited for pairing with smartphones. There are two ways of fingerprinting IoT devices which are passive and active attacks. For passive attacks, the BLE advertisement packets are sniffed with an external device in order to obtain the UUID (Universally Unique Identifier) within the unencrypted BLE traffic. Once the UUID is known attackers can determine which IoT device it belongs to and the corresponding mobile application. This can be troublesome since some BLE devices have sensitive information such as blood pressure monitors. For active attacks, if the long-term key is discovered while passively sniffing, the attacker can connect to the device and find out the next layer of UUIDs corresponding to secure services.

3. Threat Model

Device fingerprinting can be performed on different types of devices with or without the user's consent. Once the sensor fingerprint has been collected it can be used to exploit or benefit the user. The necessary motion sensor information can be extracted through an installed malicious app, during web browsing on specific websites; or even by accessing previously collected motion sensor data stored on a data server. In the case of apps and web browsers, no explicit permissions are needed to access the motion sensors of a device and so the user need not be notified.

The duration of the attack can vary depending on how long the user leaves their device unattended in the same position. Smartphone users typically leave their device unattended while charging the device or overnight for long periods of time; this is perfect for collecting sensor data. The collection window for smartwatches is going to be shorter, since the data collection requires the user to remain still while the smartwatch screen is face-up.

Collecting data from websites does not need to track cookies, IP addresses, and it not restricted by browsers with privacy protection. Once a device has been identified, it can be used to track a device across browser sessions and even when using incognito mode. The user's privacy is invaded by tracking the mobile device across different websites that the user does not what others to know they visit.

4. Background

Each generation of mobile devices introduces new hardware upgrades such as more cameras, improved displays or the addition of new sensors such as LiDAR. However, each mobile device old or new includes motion sensors. They important for monitoring device movement such as orientation, tilt, shake or rotation. This thesis shows that motion sensors can be used for more than just monitoring a device's movement, but also fingerprinting and identifying the device.

Two of the most prevalent motion sensors on modern mobile devices are the accelerometer and the gyroscope. An *accelerometer* measures the rate of change of the velocity of an object measured in G-force or meters per second squared. Typically, when a device is at rest on Earth, there is a constant force applied which is 1 G or 9.8 m/s^2. Accelerometers on mobile devices are microscope in scale which are known as electromechanical systems (MEMS), a MEMS accelerometer consists of three axes and a combination of static and moving capacitive plates. As acceleration is applied the non-static plates move in respect to the static plates the displacement causes a change in capacitance which is how the acceleration is measured. *Gyroscopes* on mobile devices, are also a MEMS, measure the angular velocity of the device in degrees per second or revolutions per second. The MEMS gyroscope consists of three axes as well, it uses a small resonating mass which shifts as the device rotates. The shifts in the mass are used to calculate the angular rotation of the device. Since MEMS sensors are on the scale of a human hair, when they are manufactured the sensor often requires factory calibration to produce accurate measurements. These imperfections will be key to finding unique characteristics about the mobile device allowing it to be fingerprinted.

For the purpose of this thesis, the measurements collected from the sensors will be referred to as a *reading*. The readings consist of the measurements of acceleration and filtered acceleration values using low-pass and high-pass filters [6]. The low-pass and high-pass filters are applied to the values from the accelerometer to eliminate gravitational forces and reduce sensor noise. In order to ensure the sensors on mobile devices can be used to identify the device, performing an analysis between sensors on different devices would be useful. Since no two sensors are identical due to fabrication methods, comparing measurements can serve as an initial analysis to determine uniqueness among sensors.

5. Initial Analysis

5.1. Sensor Analysis of Smartphones

To analyze the sensors, collecting a large sample of sensor data can provide insight if there exists a unique pattern compared to other devices. For all measurements collected, an application able to log and process measurements was created. In particular, the accelerometer is the key sensor for this research since there exists a baseline measurement. An accelerometer at rest, with the z-axis affected by Earth's gravity in the positive direction, should output a reading of 0 m/s^2 in the x-axis, y-axis, and the z-axis output 9.81 m/s^2. Similarly, when the accelerometer's z-axis is affected by gravity in the negative direction, the z-axis outputs -9.81 m/s^2 with x-axis and y-axis outputting 0 m/s^2. Lastly, when gravity is not applied to any direction on the z-axis the output is 0 m/s^2 with either the x-axis or y-axis being affected by gravity depending on the orientation of the accelerometer. In the next section, the measurements collected from the mobile devices will be compared against the baseline with gravity affecting the z-axis in the positive direction. The analysis will show that accelerometers on mobile devices have measurement error and bias which in this case the imperfections allow sensor fingerprints to exist.

Timestamp	Sensor Type	X-Axis	Y-Axis	Z-Axis
2021-02-27T12:07:15.504-0800	TYPE_ACCELEROMETER	-0.21555	-0.26824	10.021

Figure 5.1 Specifications Table for Log Reading for Sensor Measurement.

The application created for this research runs natively on Android and has a logging feature where a single reading is the timestamp of reading, sensor type, measurement value from the x-axis, y-axis and z-axis as seen in Figure 5.1. When the start button is clicked, the application waits 2 seconds before collecting data in order to avoid registering the button press. The data is written into a csv file which is uploaded to Firebase Storage once the stop button is selected, or when the application auto-stops collecting data. For the purpose of logging sensor measurements, the max

data readings are 10,000 per Sensor Type. These experiments were carried out on 4 smart phones, three Huawei Nexus 6P Smartphones and one Google Pixel 3 Smartphone. This section will include an initial analysis comparing the accelerometer values to the baseline of the Google Pixel 3, Huawei Nexus 6P, a comparison one Huawei Nexus 6P and Google Pixel 3 and lastly a comparison between two Huawei Nexus 6P. The idea between comparing sensor measurements between devices is to show that even with just raw sensor data, there is some unique measurement error even between devices of the same model.

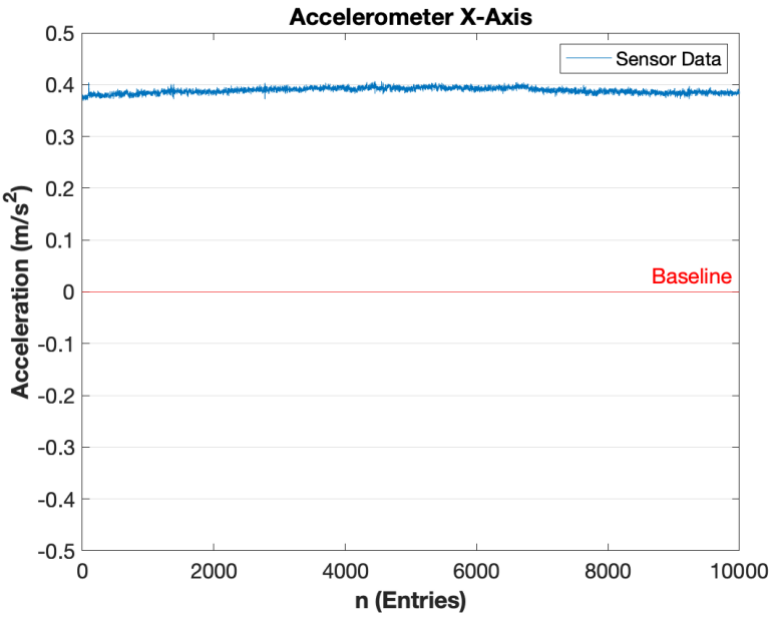


Figure 5.2 Accelerometer X-Axis Measurements from Google Pixel 3.

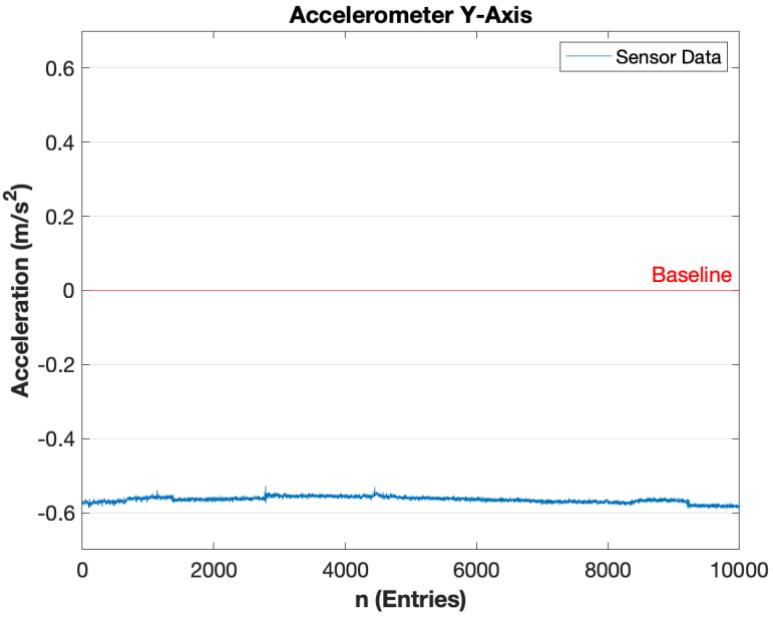


Figure 5.3 Accelerometer Y-Axis Measurements from Google Pixel 3.

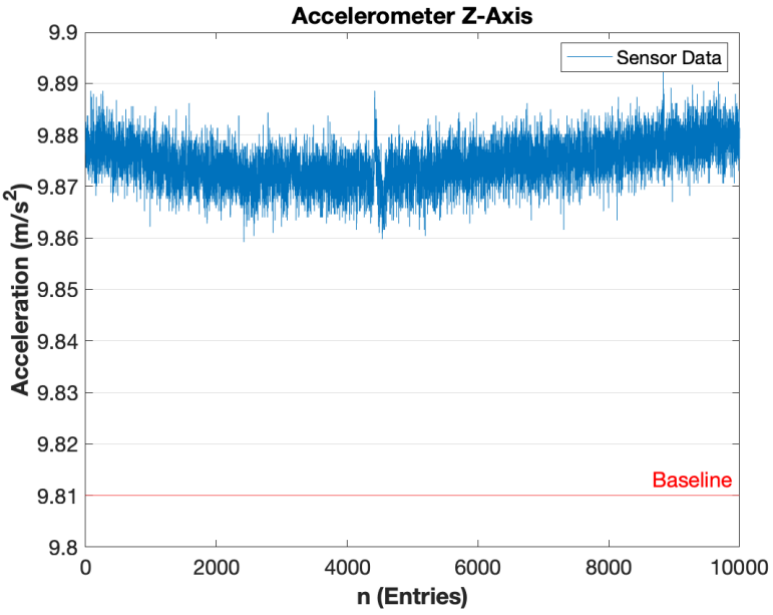


Figure 5.4 Accelerometer Z-Axis Measurements from Google Pixel 3.

Google Pixel 3 Smartphone Specifications		
Platform	OS	Android 11
	Chipset	Qualcomm SDM845 Snapdragon 845
	CPU	Octa-core (4x2.5 GHz Kryo 385 Gold & 4x1.6 GHz Kryo 385 Silver)
	Memory	4 GB
Sensor	Accelerometer	Bosch BMI160 Accelerometer

Figure 5.5 Specification Table for Google Pixel 3 Smartphone.

The analysis shows measurements collected from the Google Pixel 3, in Figure 5.2, Figure 5.3 and Figure 5.4 the x-axis of the figure is *n* for the number of readings where *n* = 10,000 and the y-axis is the acceleration values in m/s^2. The sensor analysis from the Google Pixel 3's accelerometer does have measurement error when compared to the baseline. For the x-axis, the expected value for a device at rest with gravity affecting the z-axis in the positive direction is 0 m/s^2, however the readings for that x-axis values recorded on average 0.388867 m/s^2, with a min of 0.3689 m/s^2 and a max of 0.40663 m/s^2. In the y-axis, the same observation is made where the expected value should be 0 m/s^2 but the readings indicate an average of -0.56473 m/s^2, with a min of -0.59108 m/s^2 and a max of -0.5294 m/s^2. Lastly for the z-axis, which is being affected by gravity in the positive direction, the expected value should be 9.81 m/s^2 but the readings indicate an average of 9.8748 m/s^2 with a min of 9.8592 m/s^2 and a max of 9.8939 m/s^2. This offset between the expected measurement and actual data is promising since it is a good indicator that sensors have imperfections in measurements which can converted into a sensor fingerprint.

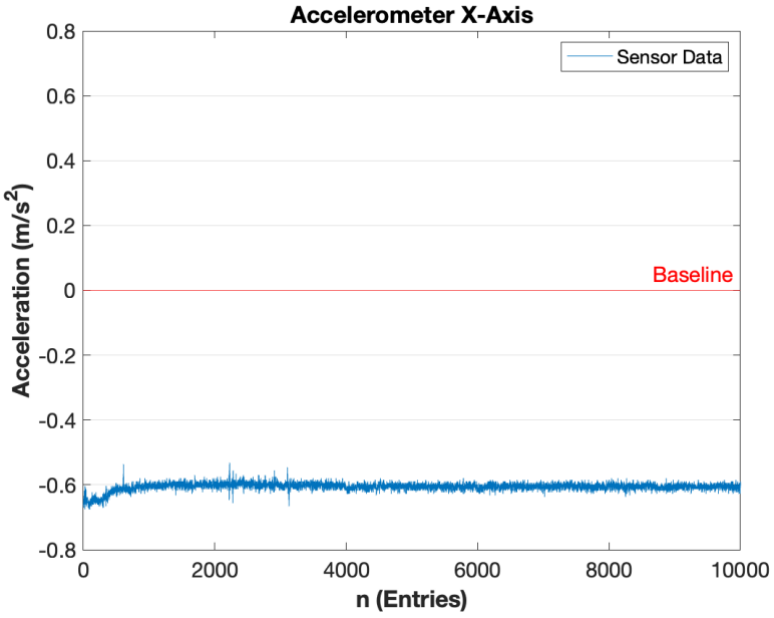


Figure 5.6 Accelerometer X-Axis Measurements from Huawei Nexus 6P.

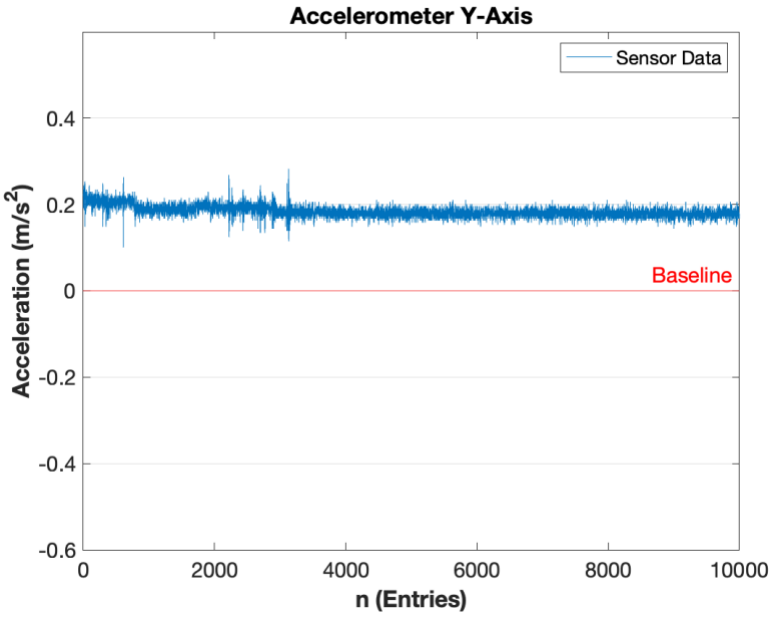


Figure 5.7 Accelerometer Y-Axis measurements from Huawei Nexus 6P.

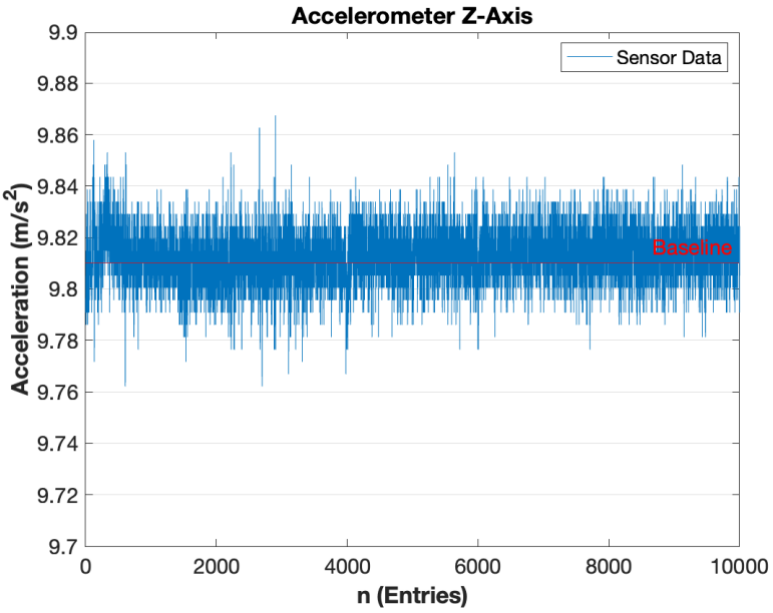


Figure 5.8 Accelerometer Z-Axis Measurements from Huawei Nexus 6P.

Huawei Nexus 6P Smartphone Specifications		
Platform	OS	Android 8.1 (Oreo)
	Chipset	Qualcomm MSM8994 Snapdragon 810
	CPU	Octa-core (4x1.55 GHz Cortex-A53 & 4x2.0 GHz Cortex-A57)
	Memory	3 GB
Sensor	Accelerometer	Bosch BMI160 Accelerometer

Figure 5.9 Specification Table for Huawei Nexus 6P Smartphone.

5.1.1. Sensor Data Comparison – Pixel 3 and Nexus 6P

The analysis for the Huawei Nexus 6P is interesting because the accelerometer chip is the same as the one on the Google Pixel 3, however, the measurement error and offset are different. This is promising because even if the accelerometer chip was made by the same manufacturer there still exists variation. For example, although the z-axis in Figure 5.8 is averaging almost 9.81 m/s^2 the x-axis and y-axis still have an offset from the baseline in Figure 5.6 and Figure 5.7.

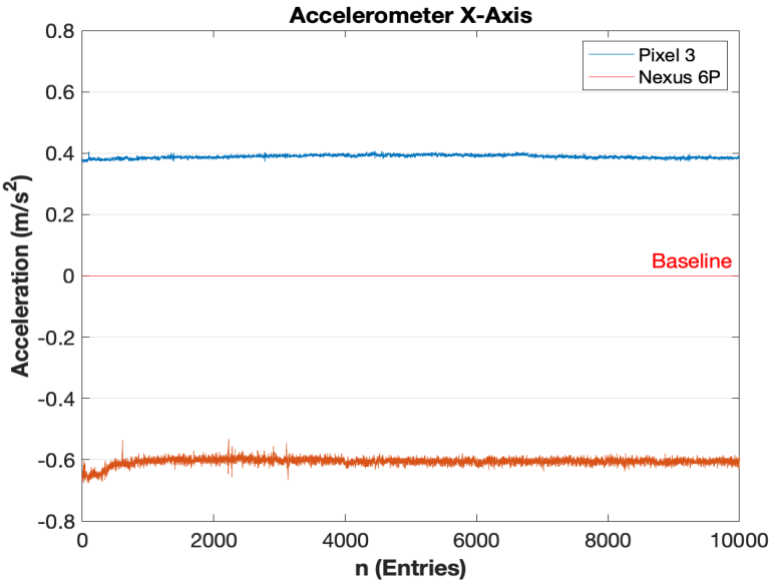


Figure 5.10 Comparing Accelerometer X-Axis - Google Pixel 3 and Huawei Nexus 6P.

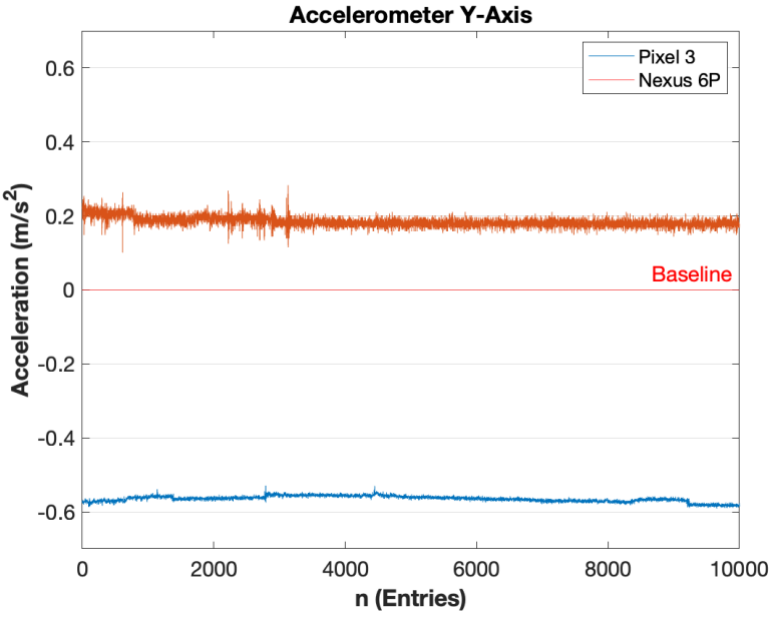


Figure 5.11 Comparing Accelerometer Y-Axis - Google Pixel 3 and Huawei Nexus 6P.

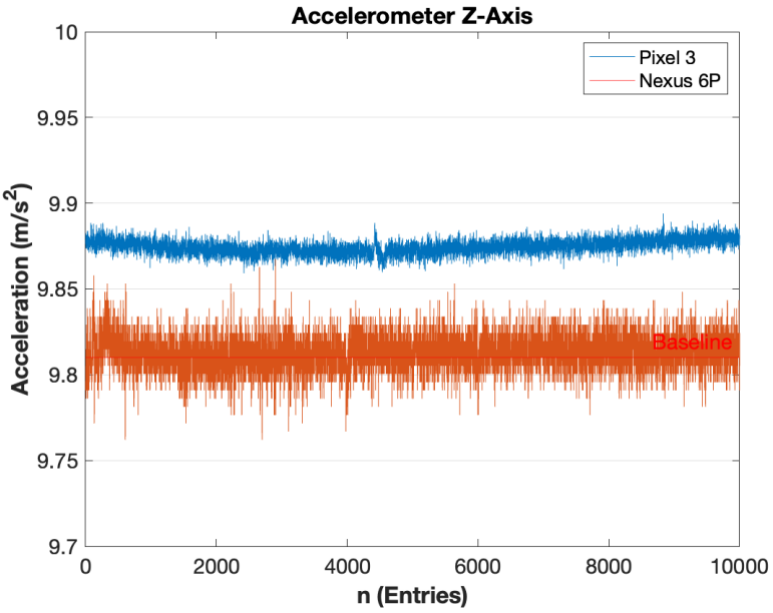


Figure 5.12 Comparing Accelerometer Z-Axis - Google Pixel 3 and Huawei Nexus 6P.

Performing a direct comparison between the raw accelerometer values of the Google Pixel 3 and Huawei Nexus 6P, Figure 5.10, Figure 5.11, and Figure 5.12 clearly demonstrates that the raw value alone could be used to identify the devices. However, in the real-world users do not leave their devices in the same position or orientation which can pose a problem if that is not expected by the application. Some challenges that will need to be overcome are detecting if the device is in the correct position and ensuring that the data collection window is long enough to collect enough information.

5.1.2. Sensor Data Comparison – Three Nexus 6P

Using three Huawei Nexus 6P, the same direct comparison of the raw accelerometer values was completed with results shown in Figure 5.13, Figure 5.14 and Figure 5.15. The devices were placed in the same position and collected 10,000 readings each. Although the devices are the exact same model with the same accelerometer model, the measurement values are distinct enough to identify. This is good news for identifying devices even among the same device model.

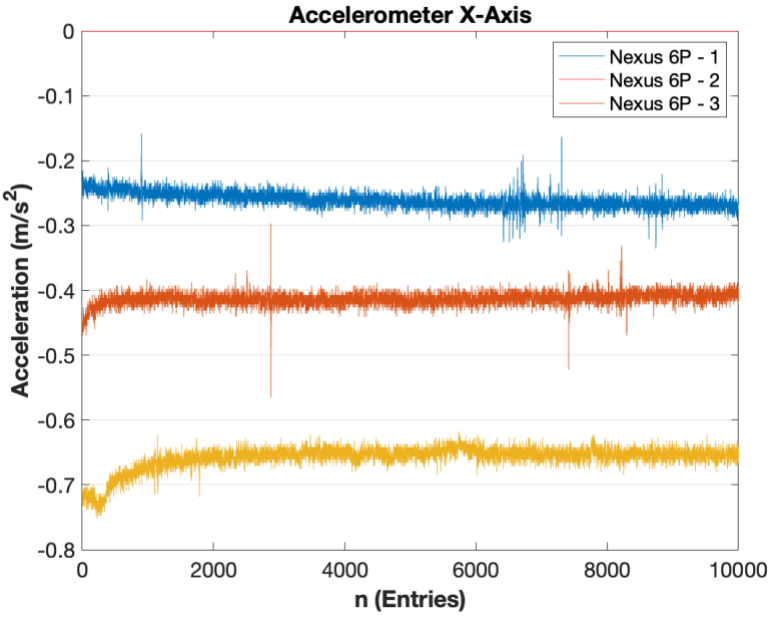


Figure 5.13 Comparing Accelerometer X-Axis – Three Huawei Nexus 6P.

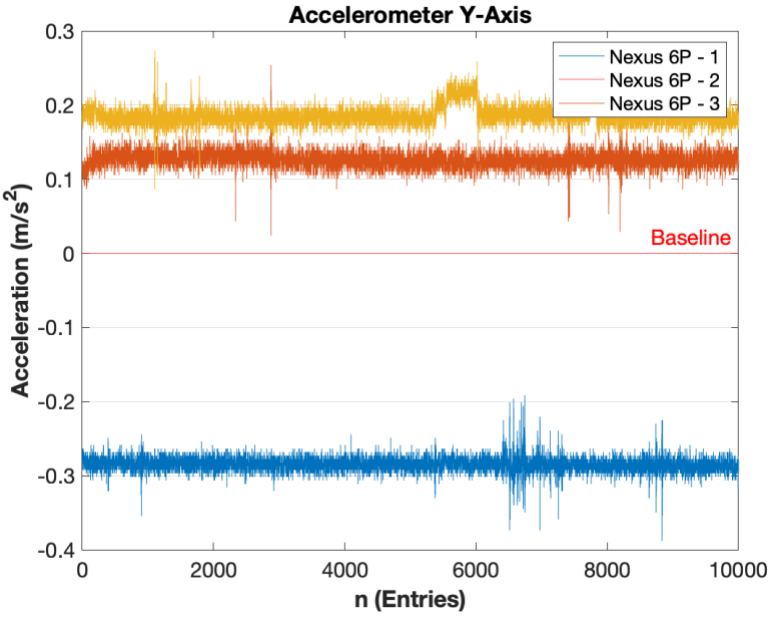


Figure 5.14 Comparing Accelerometer Y-Axis – Three Huawei Nexus 6P.

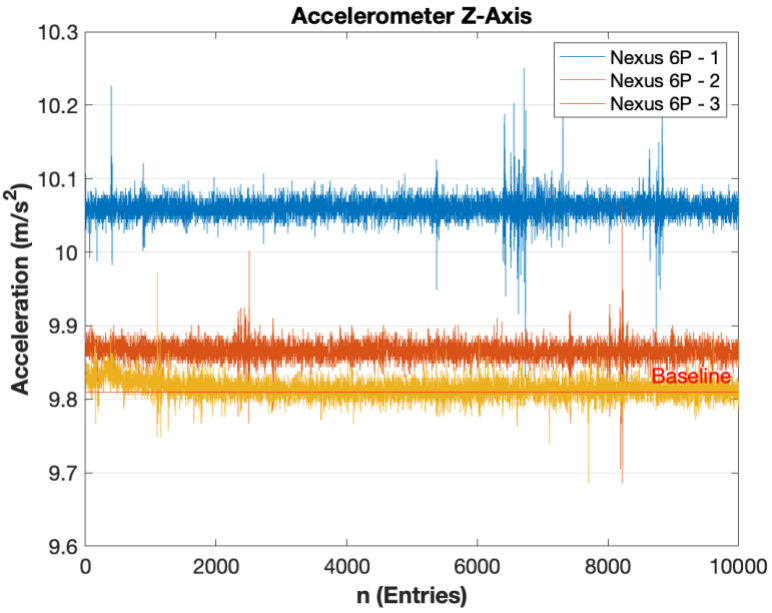


Figure 5.15 Comparing Accelerometer Z-Axis – Three Huawei Nexus 6P.

5.2. Sensor Analysis of Smartwatches

For analyzing smartwatch sensors, the same application was used as the one for the smartphones. The challenges with collecting data for the smartwatches is that in order for the z-axis to be affected by gravity, the watch screen needs to face upward. This can be achieved by wearing the watch and the wearer resting their arm on the table with the screen facing upward. Since the watch is being worn the sensors will detect more movements caused by the wearer which is not observed in smartphones.

5.2.1. Sensor Data Comparison – Sony Smartwatch3 and Motorola Moto360

Comparing the Sony Smartwatch3 and Motorola Moto360 in Figure 5.16, Figure 5.17 and Figure 5.18, some overlap can be observed but the data is still distinct between the two devices. This is still a good indicator that the sensor data is unique enough. The smartwatch sensor data appears to be less consistent compared to the smartphones which is expected.

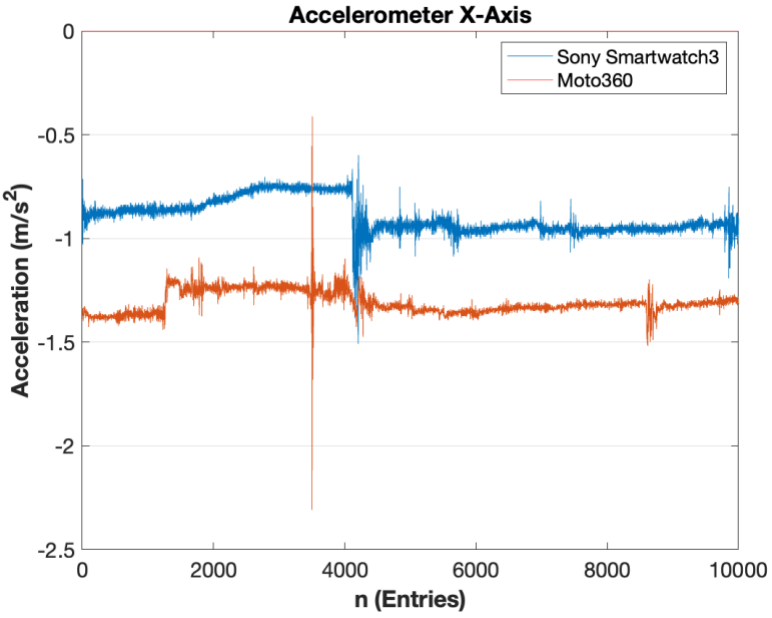


Figure 5.16 Comparing Accelerometer X-Axis – Sony Smartwatch3 and Motorola Moto360.

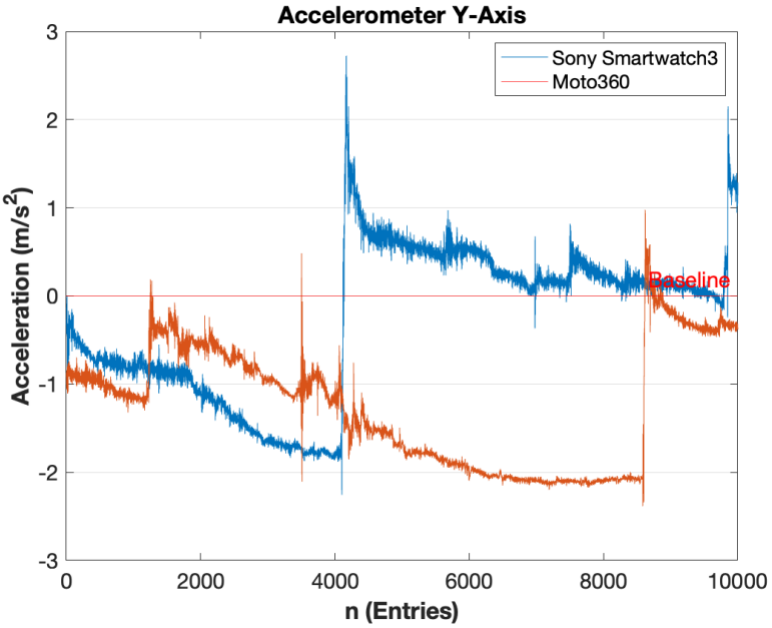


Figure 5.17 Comparing Accelerometer Y-Axis – Sony Smartwatch3 and Motorola Moto360.

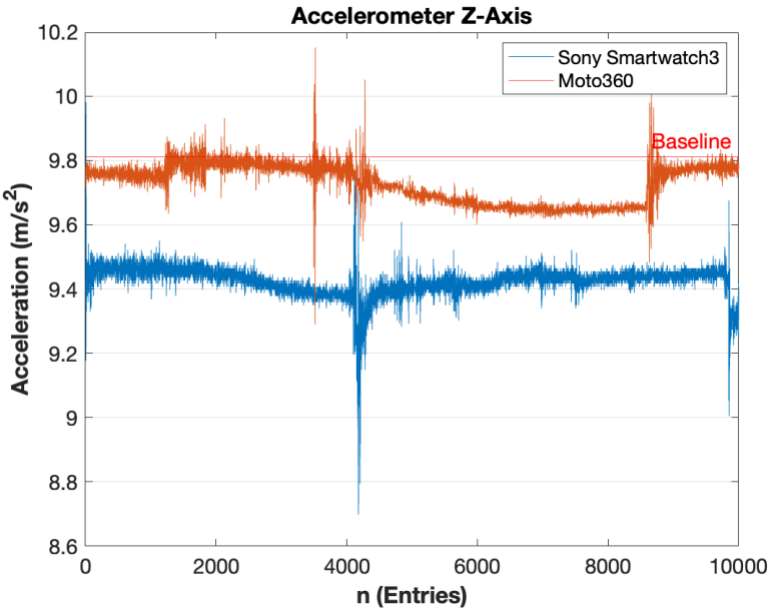


Figure 5.18 Comparing Accelerometer Z-Axis – Sony Smartwatch3 and Motorola Moto360.

5.2.2. Sensor Data Comparison – Three Motorola Moto360

Comparing three Motorola Moto360 smartwatches demonstrates once again, that same model devices still have variances in sensor data values. Although some overlap can be observed in the raw data, the sensor fingerprint will include additional identifiers to compensate.

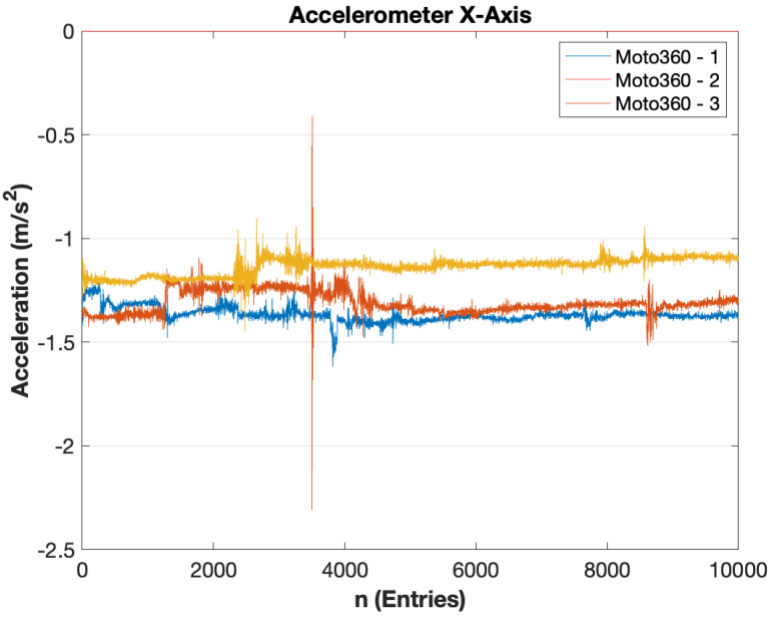


Figure 5.19 Comparing Accelerometer X-Axis – Three Motorola Moto360.

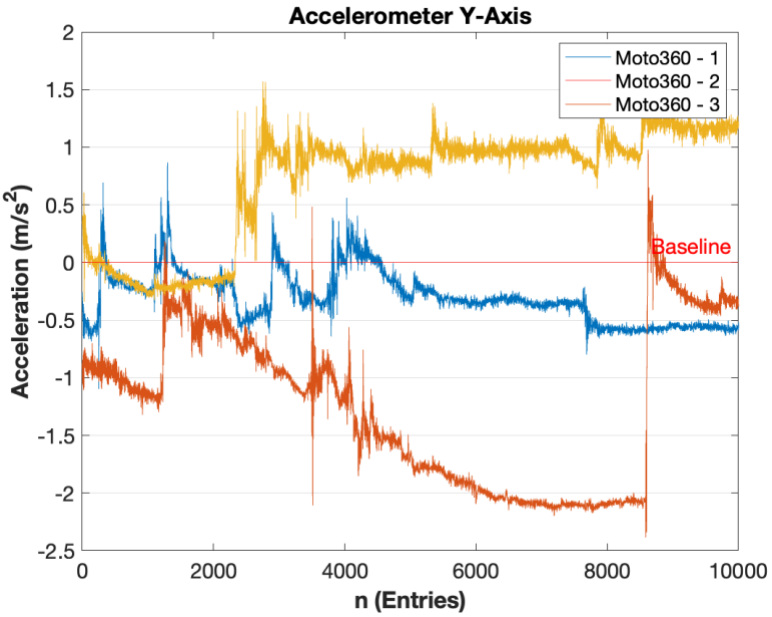


Figure 5.19 Comparing Accelerometer Y-Axis – Three Motorola Moto360.

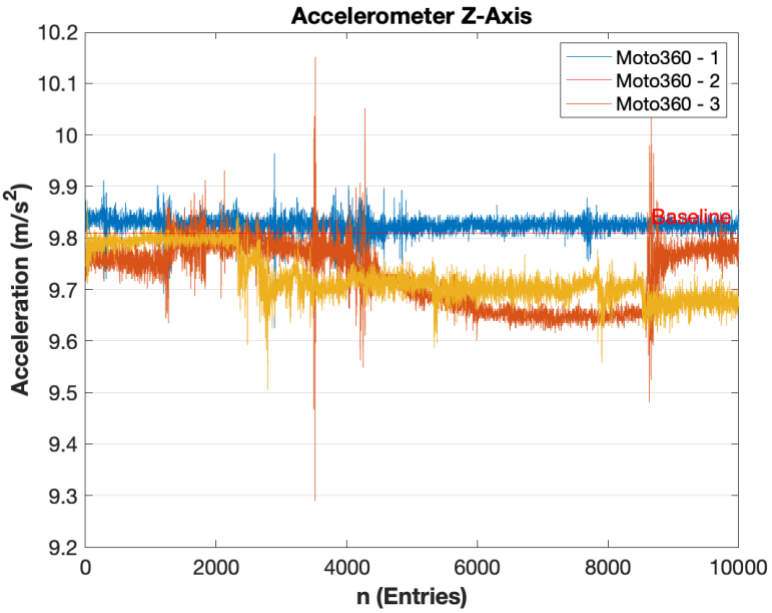


Figure 5.20 Comparing Accelerometer Z-Axis – Three Motorola Moto360.

6. Approach

In the previous chapter, the sensor analysis demonstrates that accelerometer data is a viable option for identifying both smartphones and smartwatches if they are in the correct position.

6.1. Application Requirements

In order for the application to function properly it requires two key features which are implemented for this research work. The first, the application shall be able to collect sensor data and create a sensor fingerprint. The second, the application shall collect data to compare and identify a device with an existing sensor fingerprint. Although the position of the device is important when collecting sensor data currently there is no implementation to automatically detect the position of the device, this could be an enhancement to be added later. Since the position of the phone will not be automatically detected, the device needs to be placed in the position of interest and then manually begin collecting data on the application.

6.1.1. Minimizing Sensor Noise

Although the data having variances is key to ensuring that each device can be uniquely identified having too much noise might lead to inconsistent results. In order to reduce noise in the accelerometer data the average is calculated from sensor readings collected.

6.1.2. Sensor Fingerprint Design

Data	Data Type	Weight
Universally Unique Identifier (UUID)	String	N/A
Device Model	String	[0,1]
Device Manufacturer	String	[0,1]
Accelerometer Model	String	[0,1]
Accelerometer Vendor	String	[0,1]
Accelerometer Z-Axis Maximum	Float	|Expected – Actual|
Accelerometer Z-Axis Minimum	Float	|Expected – Actual|
Accelerometer Z-Axis Bias	Float	|Expected – Actual|
Accelerometer Z-Axis Standard Deviation	Float	|Expected – Actual|
Accelerometer Z-Axis Average	Float	|Expected – Actual|
Accelerometer Z-Axis Linearity	Float	|Expected – Actual|
Accelerometer Z-Axis Sensitivity	Float	|Expected – Actual|

Figure 6.1 Sensor Fingerprint Design Table.

The implementation of the Sensor Fingerprint in Figure 6.1 shows that the Sensor Fingerprint consists of 12 attributes with their respective weights. The weights are used to minimize risk of relying on string identifiers which may be spoofed or emulated while having more weight on the sensor data. All the attributes have weights that are calculated and added together. The lower the weight value the smaller the difference between the sensor fingerprints being compared. The higher the weight value the larger the difference between the sensor fingerprint being compared. The sensor fingerprint with the lowest weight total is selected for the match after comparing the current reading with the existing sensor fingerprints.

6.1.2.1. UUID

The UUID is a 128-bit immutable number that can be used as a unique identifier [3]. The UUID is used in the Sensor Fingerprint, as the Sensor Fingerprint ID. When comparing fingerprints to identify the device, the UUID is not considered.

6.1.2.2. Device Model

The Device Model is a string value that represents the model of the mobile device. When comparing sensor fingerprints. If the Device Model matches, then the weight is 0 and if they do not match then the weight is 1. This approach reduces the likelihood of identifying a mobile device incorrectly by their device model.

6.1.2.3. Device Manufacturer

The Device Manufacturer is a string value that represents the manufacturer that created the mobile device. If the Device Manufacturer matches, then the weight is 0 and if they do not match then the weight is 1. This approach reduces the likelihood of identifying a mobile device incorrectly by their device manufacturer.

6.1.2.4. Accelerometer Model

The Accelerometer Model is a string value that represents the model of accelerometer on the mobile device. If the Accelerometer Model matches, then the weight is 0 and if they do not match then the weight is 1. This approach reduces the likelihood of identifying a mobile device incorrectly by their accelerometer model.

6.1.2.5. Accelerometer Vendor

The Accelerometer Vendor is a string value that represents the Vendor that created the accelerometer on the mobile device. If the accelerometer vendor matches, then the weight is 0 and if they do not match then the weight is 1. This approach reduces the likelihood of identifying a mobile device incorrectly by their accelerometer vendor.

6.1.2.6. Accelerometer Z-Axis Maximum

The Accelerometer Z-Axis Maximum is the maximum initial reading collected to create the sensor fingerprint of the mobile device. When comparing sensor fingerprints, the weight is calculated by taking the maximum from the initial sensor fingerprint subtracted by the maximum from the current reading and returning the absolute value.

6.1.2.7. Accelerometer Z-Axis Minimum

The Accelerometer Z-Axis Minimum is the minimum initial reading collected to create the sensor fingerprint of the mobile device. When comparing sensor fingerprints, the weight is calculated by taking the minimum from the initial sensor fingerprint subtracted by the minimum from the current reading and returning the absolute value.

6.1.2.8. Accelerometer Z-Axis Bias

The Accelerometer Z-Axis Bias is the difference between Gravity and the average reading collected to create the sensor fingerprint of the mobile device. When comparing sensor fingerprints, the weight is calculated by taking the bias from the initial sensor fingerprint subtracted by the bias from the current reading and returning the absolute value.

6.1.2.9. Accelerometer Z-Axis Standard Deviation

The Accelerometer Z-Axis Standard Deviation is the standard deviation of all the readings of the z-axis collected during the initial creation of the sensor fingerprint of the mobile device. When comparing sensor fingerprints, the weight is calculated by taking the standard deviation from the initial sensor fingerprint subtracted by the standard deviation from the current reading and returning the absolute value.

6.1.2.10. Accelerometer Z-Axis Average

The Accelerometer Z-Axis Average is the average of all the readings of the z-axis collected during the initial creation of the sensor fingerprint of the mobile device. When comparing sensor fingerprints, the weight is calculated by taking the average from the initial sensor fingerprint subtracted by the average from the current reading and returning the absolute value.

6.1.2.11. Accelerometer Z-Axis Linearity

$$Linearity = Z_{0g} - \frac{(Z_{+1g} + Z_{-1g})}{2}$$

Figure 6.2 Linearity Equation.

The Accelerometer Z-Axis Linearity is the offset the accelerometer measurement for the Z-Axis. The inputs expected are Z_{0g} when the mobile device is held vertically, Z_{+1g} when the mobile device screen is face up and Z_{-1g} when the mobile device screen is face down. Currently only requires Z_{+1g} while the other two inputs use constants of 0 G for Z_{0g} and -1 G for Z_{-1g}. When comparing sensor fingerprints, the weight is calculated by taking the linearity from the initial sensor fingerprint subtracted by the linearity from the current reading and returning the absolute value.

6.1.2.12. Accelerometer Z-Axis Sensitivity

$$Sensitivity = \frac{Z_{+1g} - Z_{-1g}}{2g}$$

Figure 6.3 Sensitivity Equation.

The Accelerometer Z-Axis Sensitivity measures the sensitivity of the accelerometer sensor based on the measurements when the device is face up and facedown. Currently only Z_{+1g} is required while a constant of -1 G is used for Z_{-1g}. When comparing sensor fingerprints, the weight is

calculated by taking the sensitivity from the initial sensor fingerprint subtracted by the sensitivity from the current reading and returning the absolute value.

6.1.2.13. Attributes Considered

For the Sensor Fingerprint Design, the operating system version of the mobile device was considered but mobile devices have frequent system updates. Relying on the operating system version would not be useful and was therefore excluded. Including additional sensor information and data was considered such as the gyroscope sensor, however, accelerometer was selected since there exists a baseline or expected value to compare.

6.2. Implementation

The prototype is an Android application given the name, Sensor Onboard Fingerprint Identity Authenticator or SOFIA. It can be run on a device that is running Android 7 or later and an internet connect is required to upload the data to the cloud. As stated, native Android applications do not require permission to access the sensors on the mobile device. The prototype uses the same Android APIs to access the sensors on both the smartphones and smartwatches.

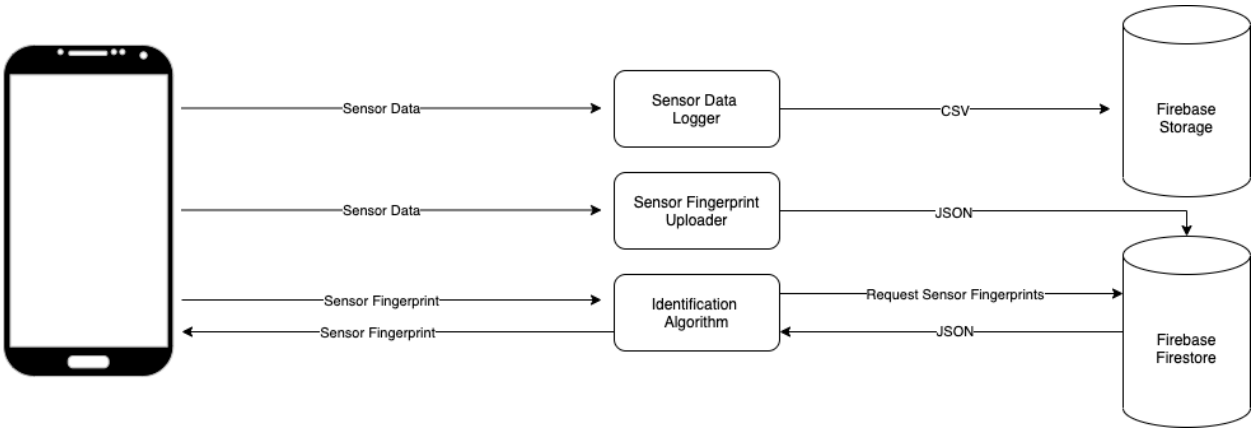
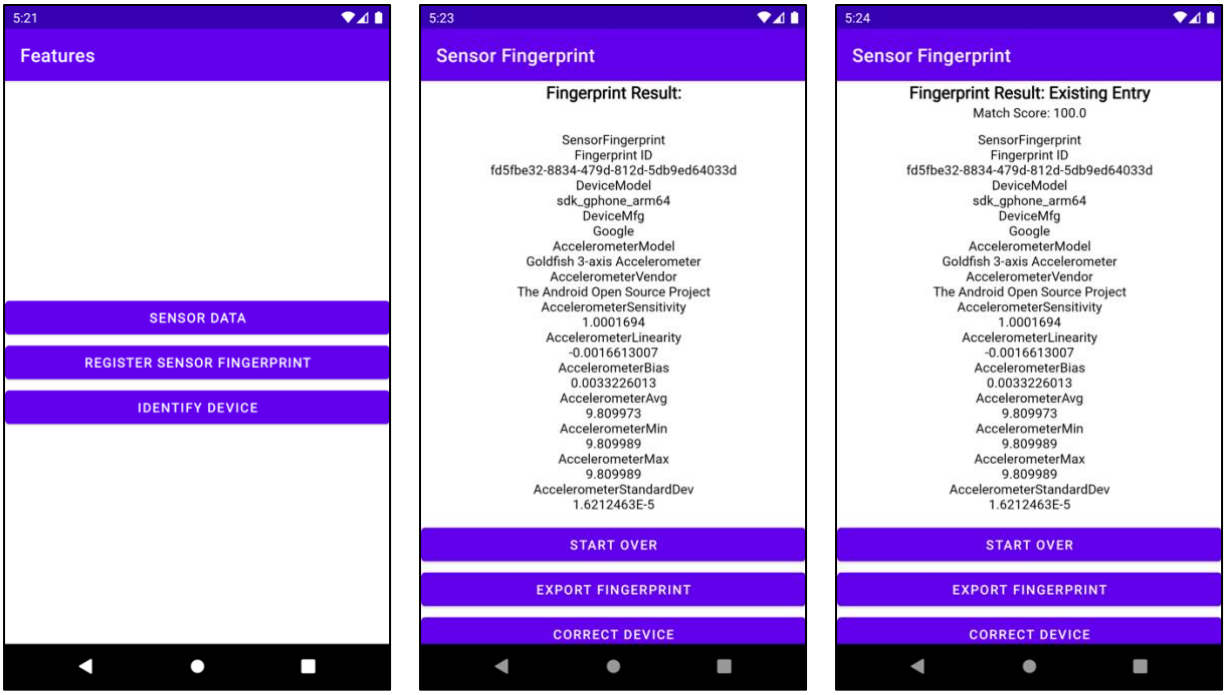


Figure 6.4 SOFIA Feature Diagram.

As shown in Figure 6.4, SOFIA has three main features Sensor Data Logger, Sensor Fingerprint Uploader and Identification Algorithm. Starting with the Sensor Data Logger, the sensor data is collected, formatted and written to a csv file by the Sensor Data Logger. The csv file is then uploaded to the Firebase Storage. The second feature is Sensor Fingerprint Uploader, which takes the sensor data converts it into a sensor fingerprint as seen in Figure 6.1 and after the fingerprint is created it is uploaded to Firebase Firestore as a json. Lastly, the most important part is the Identification Algorithm which take a local sensor fingerprint, compares it with the sensor fingerprints from the cloud and returns sensor fingerprint from the cloud. The sensor fingerprint that is returned is expected to be the closest match to a previously identified device.



(a) (b) (c)

Figure 6.5 SOFIA User Interface Screenshots.

(a) List of buttons that are the main features in SOFIA which are Sensor Data, Register Sensor Fingerprint, and Identify Device. (b) Screenshot of Sensor fingerprint when the device is initially identified using Register Sensor Fingerprint feature. (c) Screenshot of the Sensor fingerprint identified to be the closest match using Identify Device feature.

7. Experimentation

7.1. Experimental Setup

For experimentation, there were two setups one for smartphones and one for smartwatches. The key difference was that the smartphone was placed on the table and the smartwatch was worn on the wrist while the arm is resting on the table.

7.1.1. Smartphone Setup

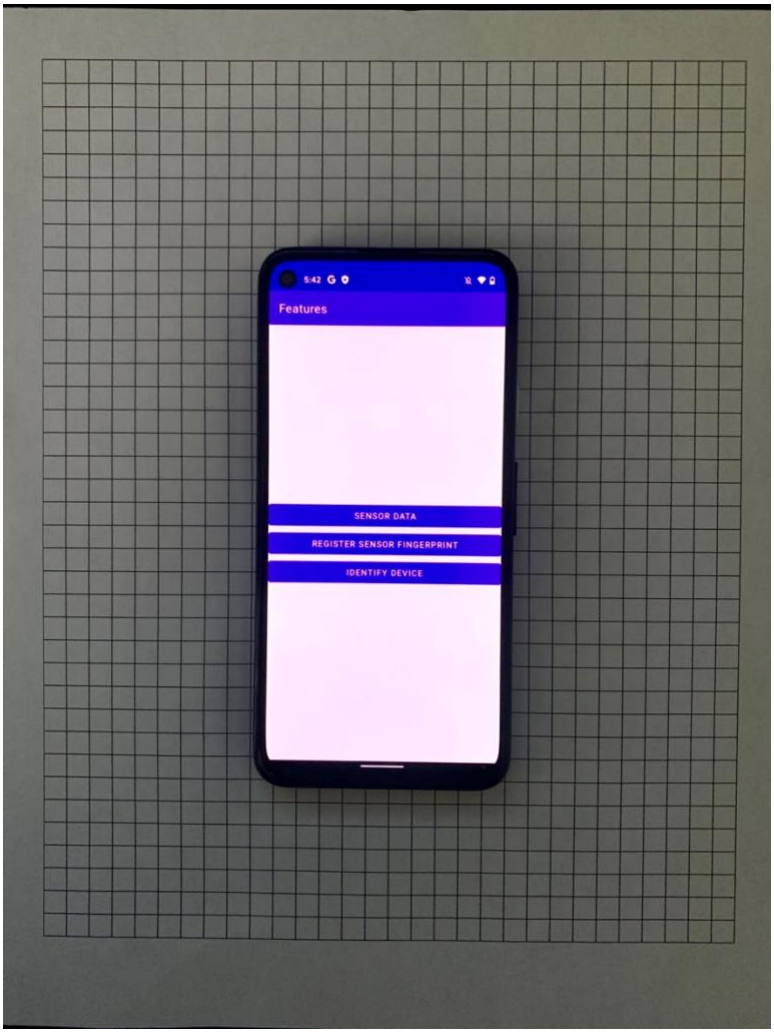


Figure 7.1 Smartphone Data Collection App Photo.

For the smartphone setup the mobile device was placed face-up on a table as shown in Figure 7.1 with no further interaction during data collection. The position of the device will depend on the

scenario that is being tested. The device that will be used for this experiment are Google Pixel 3, three Huawei Nexus 6P and 10 Google Pixel 4a. All of the smartphones come with an accelerometer sensor which is necessary for the identification process.

7.1.2. Smartwatch Setup

Figure 7.2 Smartwatch Data Collection App Photo.

For the smartwatch setup, the smartwatch was be worn on the wrist while the arm was resting on a table with the screen facing up as shown on Figure 7.2. The smartwatches used were three Motorola Moto360 and one Sony Smartwatch3. All the smartwatches used have an accelerometer.

7.2. Controlled Scenario – Same Device Position on Table

For the Controlled Scenario, the mobile devices were placed in the same position as the initial reading was collected for creating the sensor fingerprints. For the smartphones, the position was the device flat on a table with the screen facing up. For the smartwatches, the position was the watch wore on the left wrist with the arm resting on the table. The experiment was performed for 10 iterations on each device in the same position. On each device, only a single fingerprint was created. Using the identification feature in SOFIA, the sensor data was collected, and a sensor fingerprint was created on every iteration. The newly created sensor fingerprint is compared with the initial sensor fingerprints saved on the cloud. After the comparison completed, manual confirmation whether the identification was correct or incorrect on the prototype was entered for logging purposes.

7.2.1. Controlled Scenario – Smartphone Identification Data

The sensor data collected from smartphones during the realistic scenario experiment is showcased in this section. The Pixel 3 and three Nexus 6P devices are consistent in most numerical attributes from the sensor fingerprint model. This can be observed as a horizontal plot of points for each respective device. However, the data collected from the 10 Pixel 4a smartphones showed there was overlap between the devices, even when changing positions which led to device to be identified incorrectly by SOFIA.

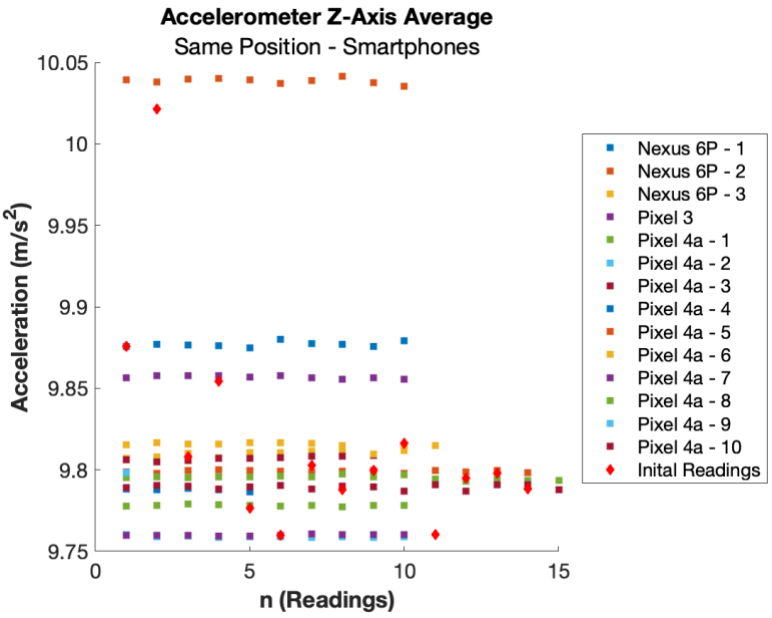


Figure 7.3 Smartphones – Same Position: Z-Axis Average.

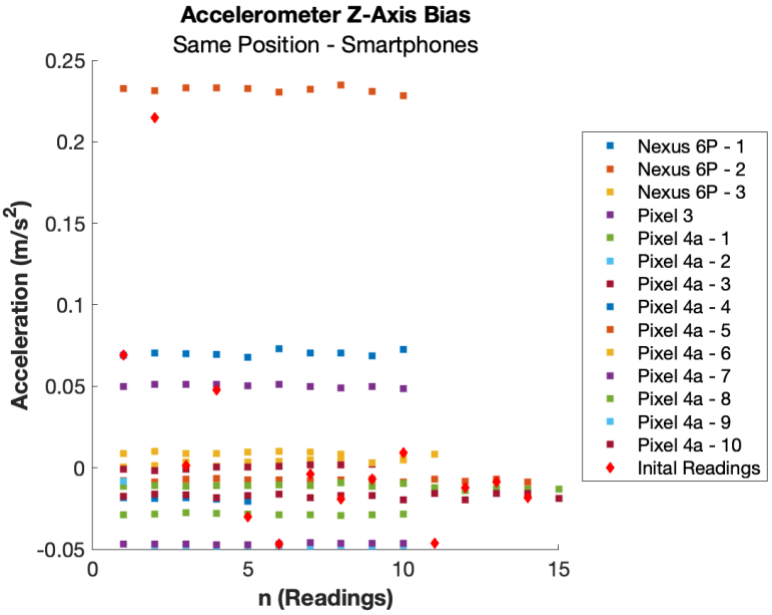


Figure 7.4 Smartphones – Same Position: Z-Axis Bias.

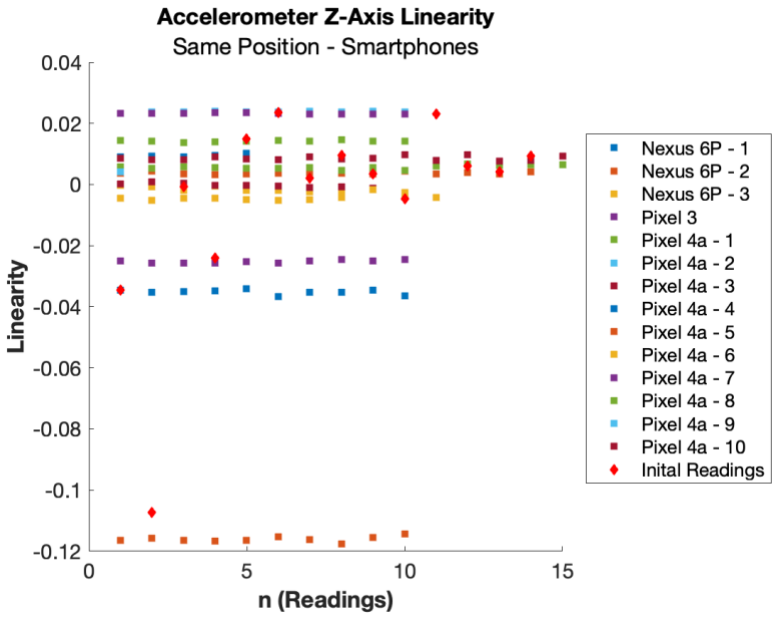


Figure 7.5 Smartphones – Same Position: Z-Axis Linearity.

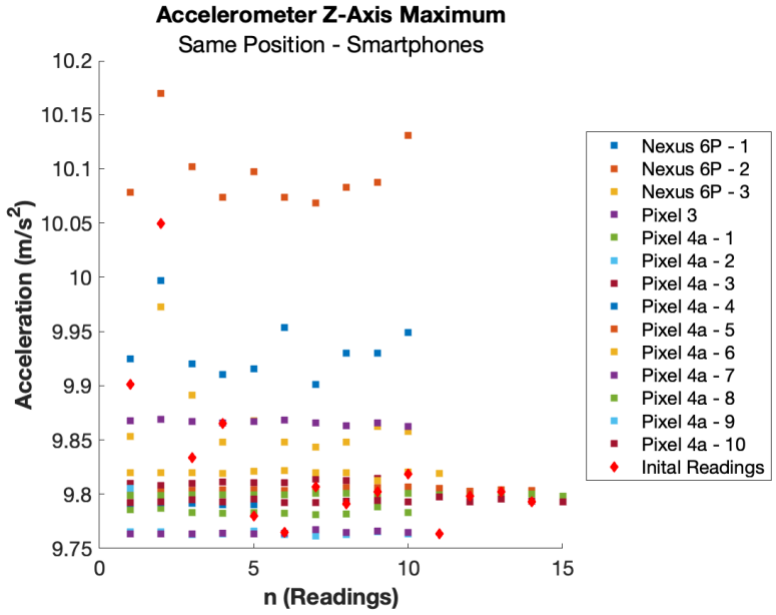


Figure 7.6 Smartphones – Same Position: Z-Axis Maximum.

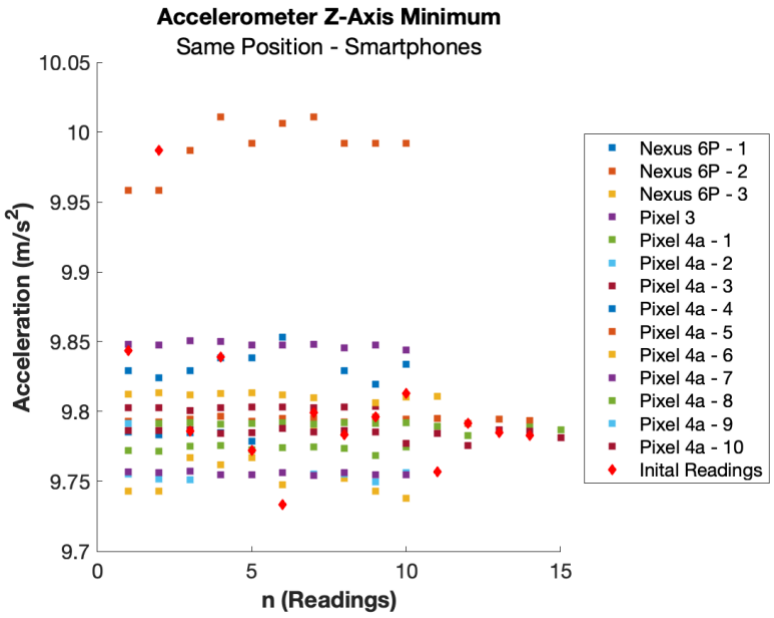


Figure 7.7 Smartphones – Same Position: Z-Axis Minimum.

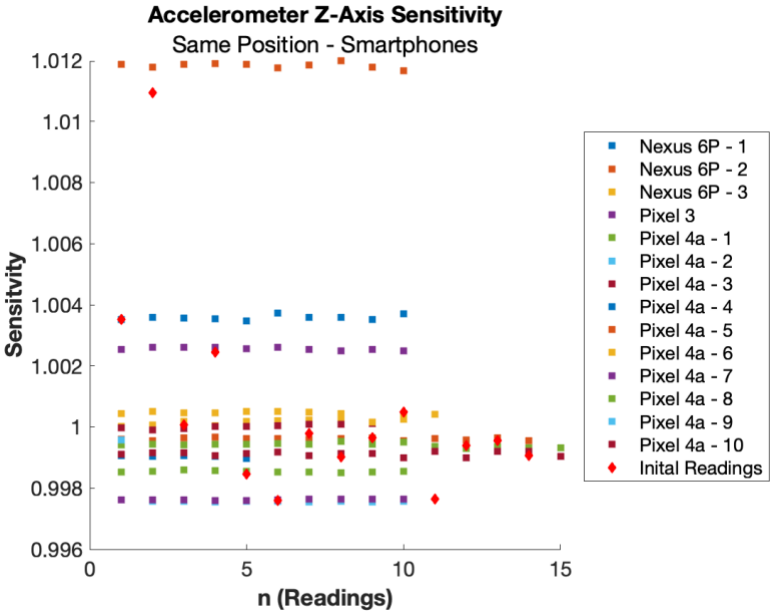


Figure 7.8 Smartphones – Same Position: Z-Axis Sensitivity.

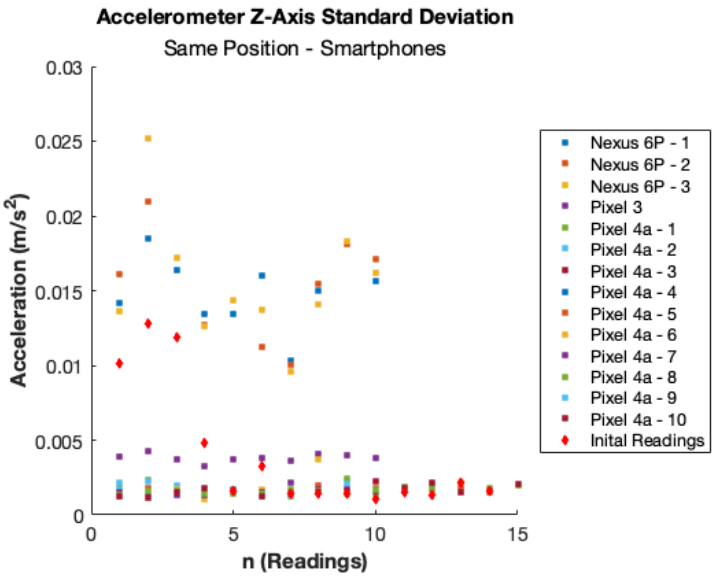


Figure 7.9 Smartphones – Same Position: Z-Axis Standard Deviation.

7.2.2. Controlled Scenario – Smartwatch Identification Data

The sensor data collected from smartwatches during the controlled experiment demonstrated that some devices provide more consistent data than others. The SmartWatch3 demonstrated consistency in most attributes, this can be seen as a horizontal plot of points for the SmartWatch3. However, the data collected from the three Moto360 watches showed there was overlap between the devices which led to device to be identified incorrectly by SOFIA.

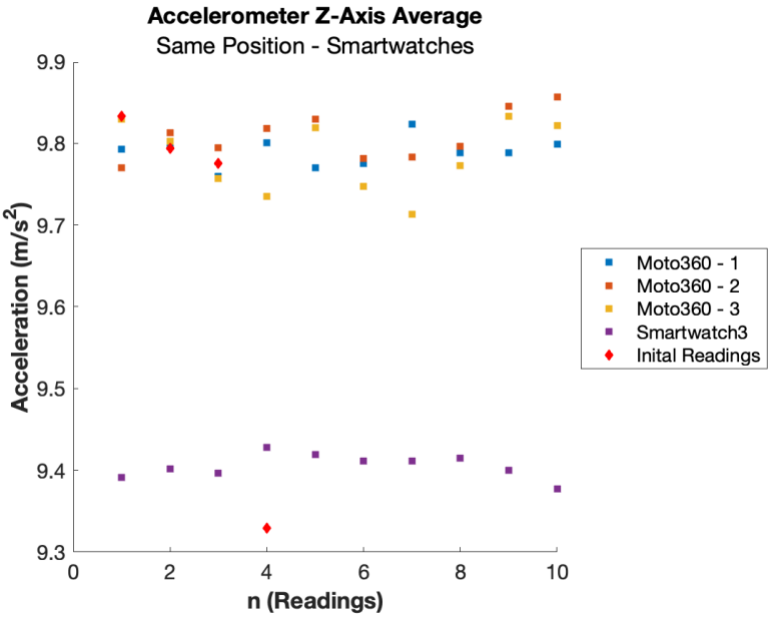


Figure 7.10 Smartwatches – Same Position: Z-Axis Average.

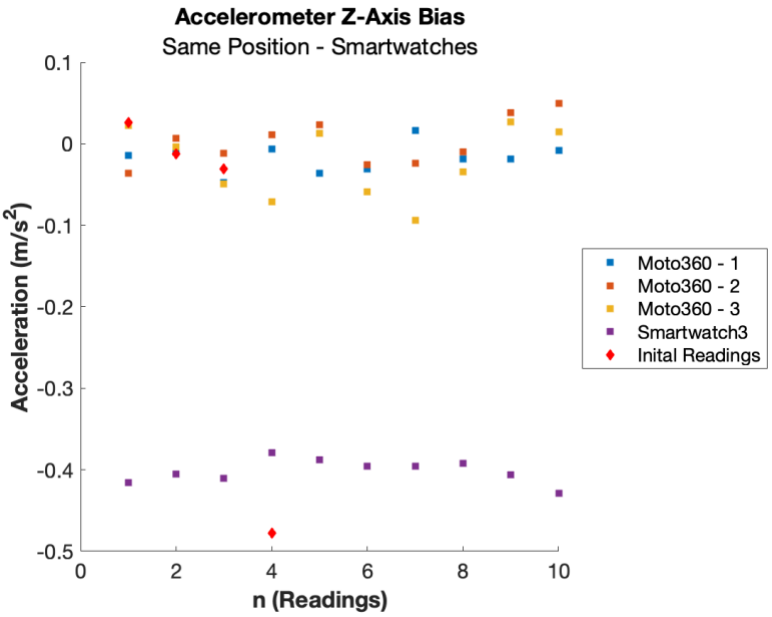


Figure 7.11 Smartwatches – Same Position: Z-Axis Bias.

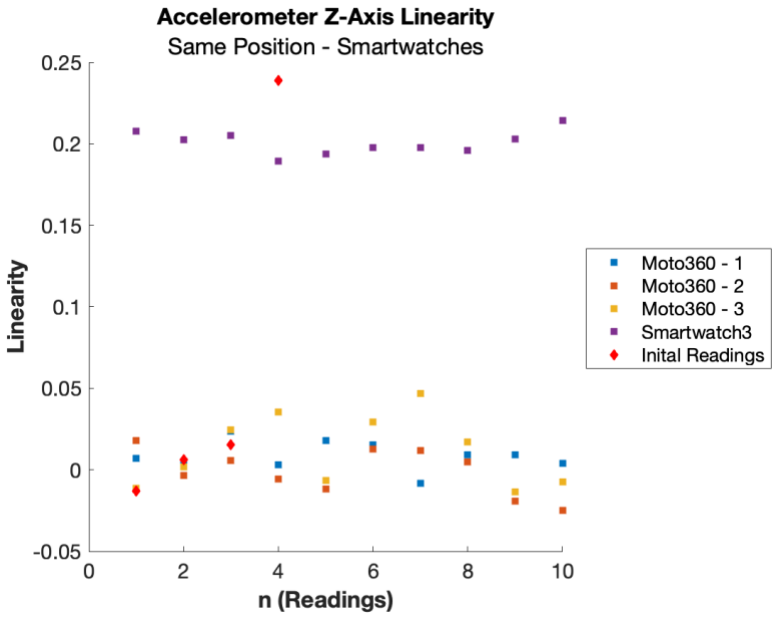


Figure 7.12 Smartwatches – Same Position: Z-Axis Linearity.

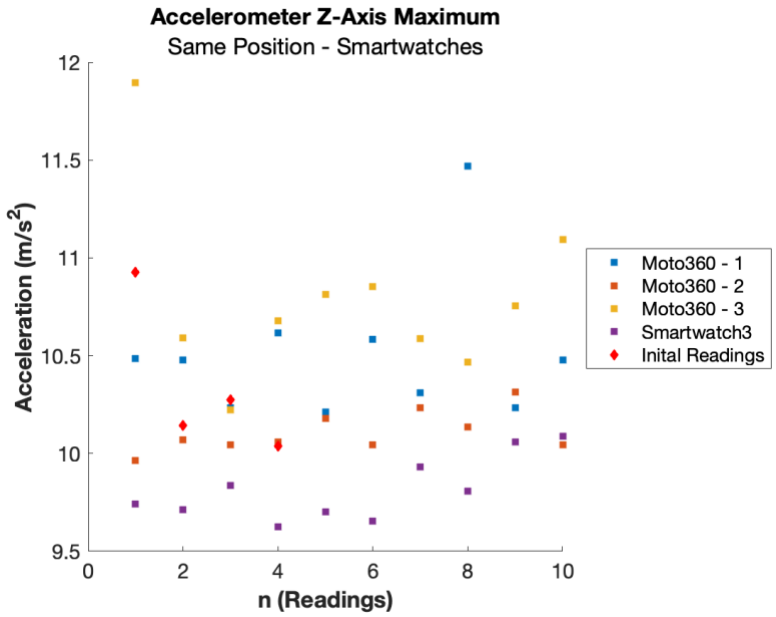


Figure 7.13 Smartwatches – Same Position: Z-Axis Maximum.

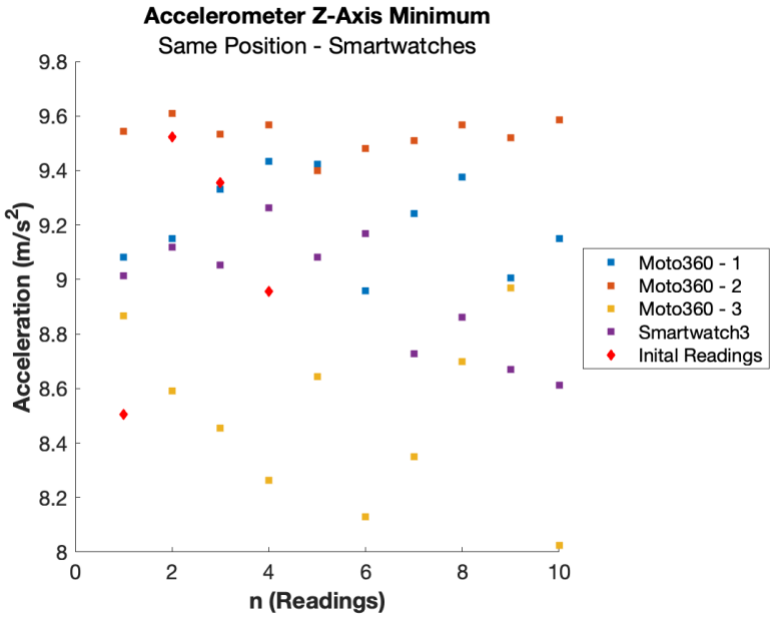


Figure 7.14 Smartwatches – Same Position: Z-Axis Minimum.

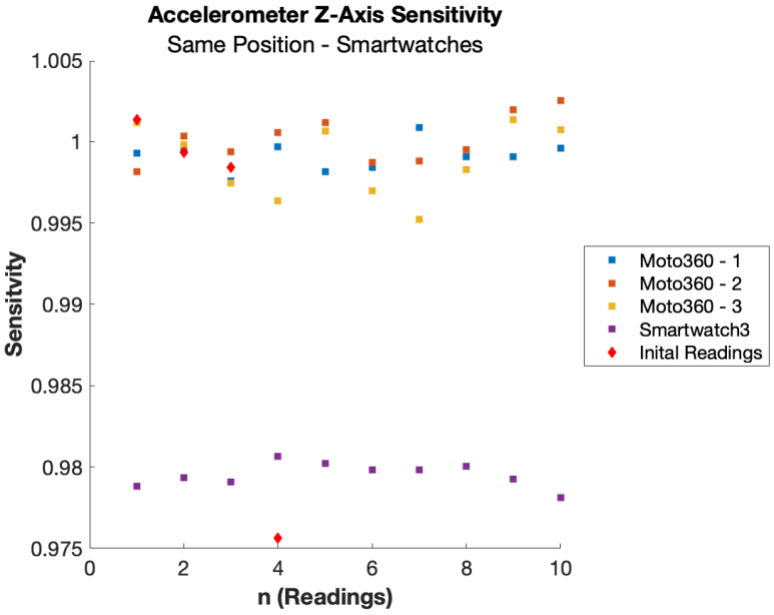


Figure 7.15 Smartwatches – Same Position: Z-Axis Sensitivity.

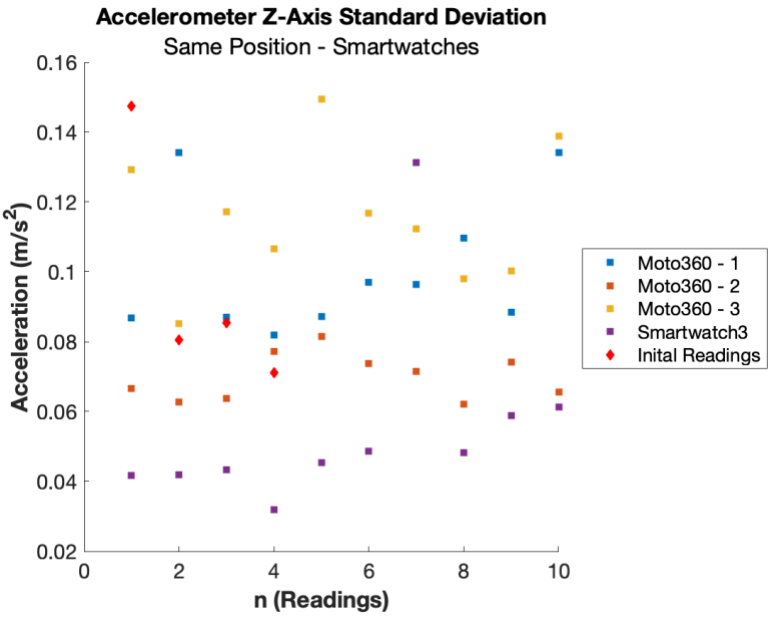


Figure 7.16 Smartwatches – Same Position: Z-Axis Standard Deviation.

7.2.3. Controlled Scenario – Results

	N = 140	Pixel 3	Nexus 6P – 1	Nexus 6P – 2	Nexus 6P – 3	Pixel 4a – 1	Pixel 4a – 2	Pixel 4a – 3	Pixel 4a – 4	Pixel 4a – 5	Pixel 4a – 6	Pixel 4a – 7	Pixel 4a – 8	Pixel 4a – 9	Pixel 4a – 10	
		Actual														
Predicted	Pixel 3	10	0	0	0	0	0	0	0	0	0	0	0	0	0	10
	Nexus 6P – 1	0	10	0	0	0	0	0	0	0	0	0	0	0	0	10
	Nexus 6P – 2	0	0	10	0	0	0	0	0	0	0	0	0	0	0	10
	Nexus 6P – 3	0	0	0	10	0	0	0	0	0	0	0	0	0	0	10
	Pixel 4a – 1	0	0	0	0	10	0	0	0	0	0	0	0	0	0	10
	Pixel 4a – 2	0	0	0	0	0	10	0	0	0	0	0	0	0	0	10
	Pixel 4a – 3	0	0	0	0	0	0	9	0	0	0	0	0	0	0	9
	Pixel 4a – 4	0	0	0	0	0	0	0	0	0	0	0	0	0	5	5
	Pixel 4a – 5	0	0	0	0	0	0	0	0	10	0	3	0	1	0	14
	Pixel 4a – 6	0	0	0	0	0	0	1	0	0	10	0	0	0	0	11
	Pixel 4a – 7	0	0	0	0	0	0	0	0	0	0	6	0	9	0	15
	Pixel 4a – 8	0	0	0	0	0	0	0	0	0	0	0	10	0	0	10
	Pixel 4a – 9	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1
	Pixel 4a – 10	0	0	0	0	0	0	0	10	0	0	0	0	0	5	15
		10	10	10	10	10	10	10	10	10	10	10	10	10	10	

Figure 7.17 Smartphones – Same Position: Confusion Matrix Results.

		Actual				
	N = 40	SmartWatch3	Moto360 – 1	Moto360 – 2	Moto360 – 3	
Predicted	SmartWatch3	10	0	0	0	10
	Moto360 – 1	0	2	5	2	9
	Moto360 – 2	0	7	0	3	10
	Moto360 – 3	0	1	5	5	11
		10	10	10	10	

Figure 7.18 Smartwatches – Same Position: Confusion Matrix Results.

Based on the smartphone data from Figure 7.17, demonstrated that the accuracy of SOFIA when tested on smartphone devices for the controlled scenario was 78%. There where 110 instances where a smartphone was correctly identified out of the 140 iterations. The misclassification rate of SOFIA when tested on smartphones was 21%, because there were 30 instances where the smartphone was incorrectly identified out of the 140 iterations.

The smartwatch data from Figure 7.18, demonstrated that the accuracy of SOFIA when tested on smartwatches for the controlled scenario was 42%. There were 17 instances where a smartwatch was correctly identified out of 40 iterations. The misclassification rate of SOFIA on smartwatches was 57%. because there were 23 instances where a smartwatch was incorrectly identified out of 40 iterations. Compared to the smartphones, the accuracy for smartwatches was reduced because the sensors on smartwatches were influenced by movement causing the sensor data to be less consistent.

7.3. Realistic Scenario – Varied Device Position on Table

For the different position setup, the mobile devices were flat on a table with the screen facing up, diagonal, sideways or upside-down. The experiment was performed for 10 iterations on each device in the same position. On each device, only a single fingerprint was created. Using the identification feature in SOFIA, the sensor data was collected, and a sensor fingerprint created on every identification iteration. The newly created sensor fingerprint is compared with the initial sensor fingerprints saved on the cloud. After the comparison completed, manual confirmation whether the identification was correct or incorrect on the prototype was entered for logging purposes.

7.3.1. Realistic Scenario – Smartphone Identification Data

The sensor data collected from smartphones during the realistic experiment demonstrated that changing the position of the devices still provided consistent results on certain devices. The Pixel 3 and three Nexus 6P devices remained consistent once again in most numerical attributes from the sensor fingerprint model. This can be observed as a horizontal plot of points for each respective device. However, the data collected from the 10 Pixel 4a smartphones showed there was overlap between the devices, even when changing positions which led to device to be identified incorrectly by SOFIA. There was an outlier in the scatter plot which was an instance when the sensor data was collected while the device was upside-down.

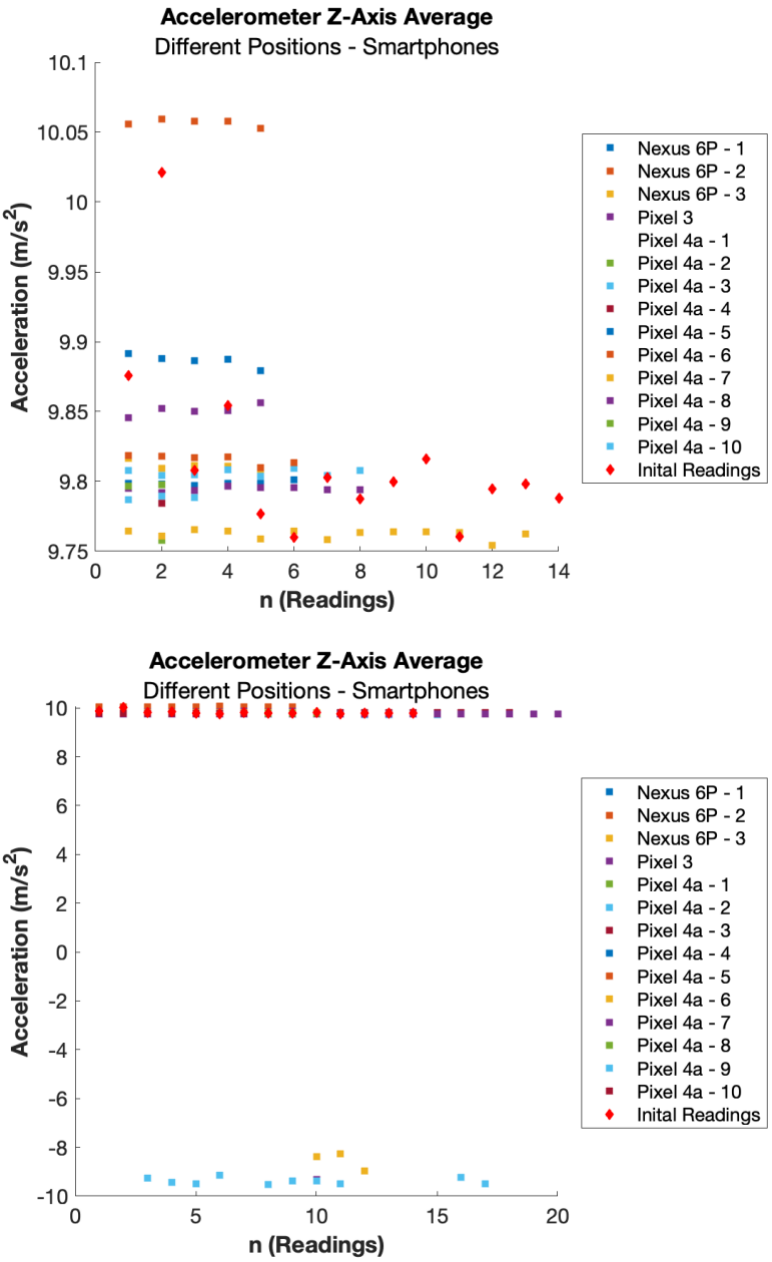


Figure 7.19 Smartphones – Different Positions: Z-Axis Average.

(Top) Accelerometer Z-Axis Average excluding iteration that places phone face down. (Bottom) Accelerometer Z-Axis Average including iteration that places phone face down.

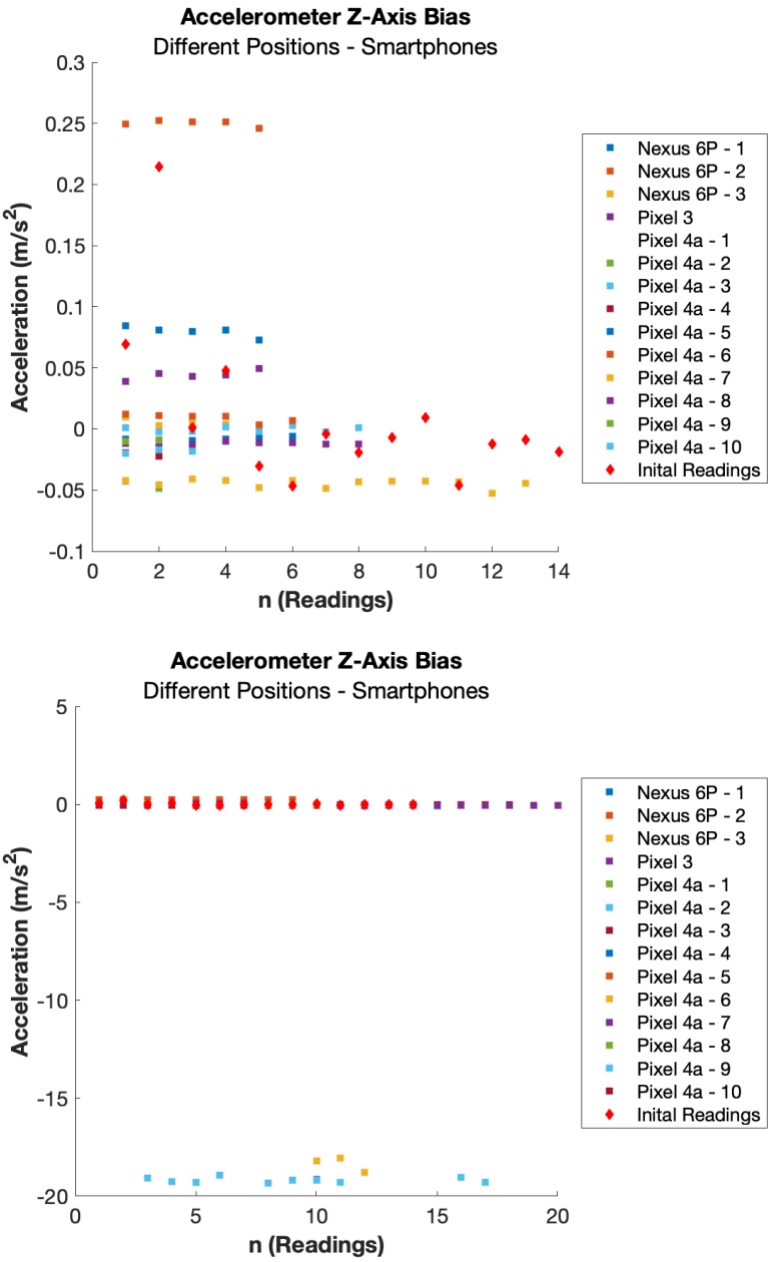


Figure 7.20 Smartphones – Different Positions: Z-Axis Bias.

(Top) Accelerometer Z-Axis Bias excluding iteration that places phone face down. (Bottom) Accelerometer Z-Axis Bias including iteration that places phone face down.

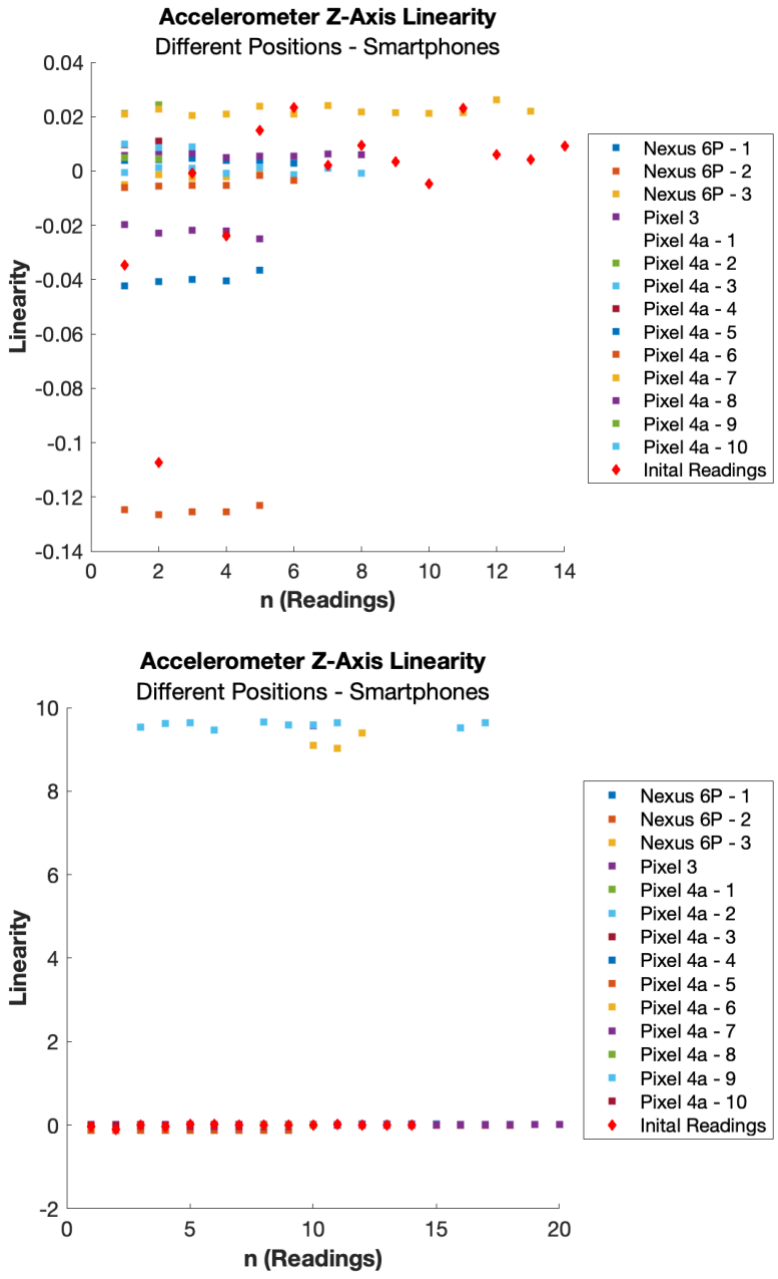


Figure 7.21 Smartphones – Different Positions: Z-Axis Linearity.

(Top) Accelerometer Z-Axis Linearity excluding iteration that places phone face down. (Bottom) Accelerometer Z-Axis Linearity including iteration that places phone face down.

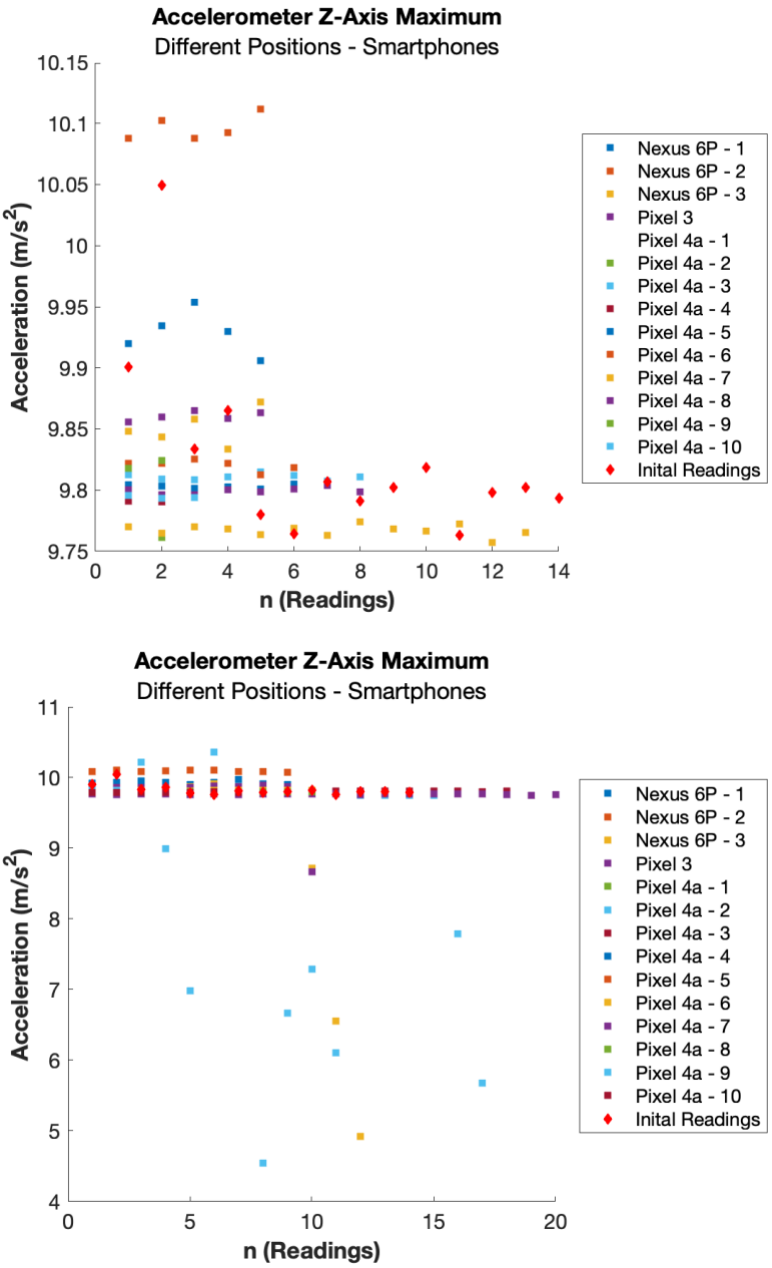


Figure 7.22 Smartphones – Different Positions: Z-Axis Maximum.

(Top) Accelerometer Z-Axis Maximum excluding iteration that places phone face down. (Bottom) Accelerometer Z-Axis Maximum including iteration that places phone face down.

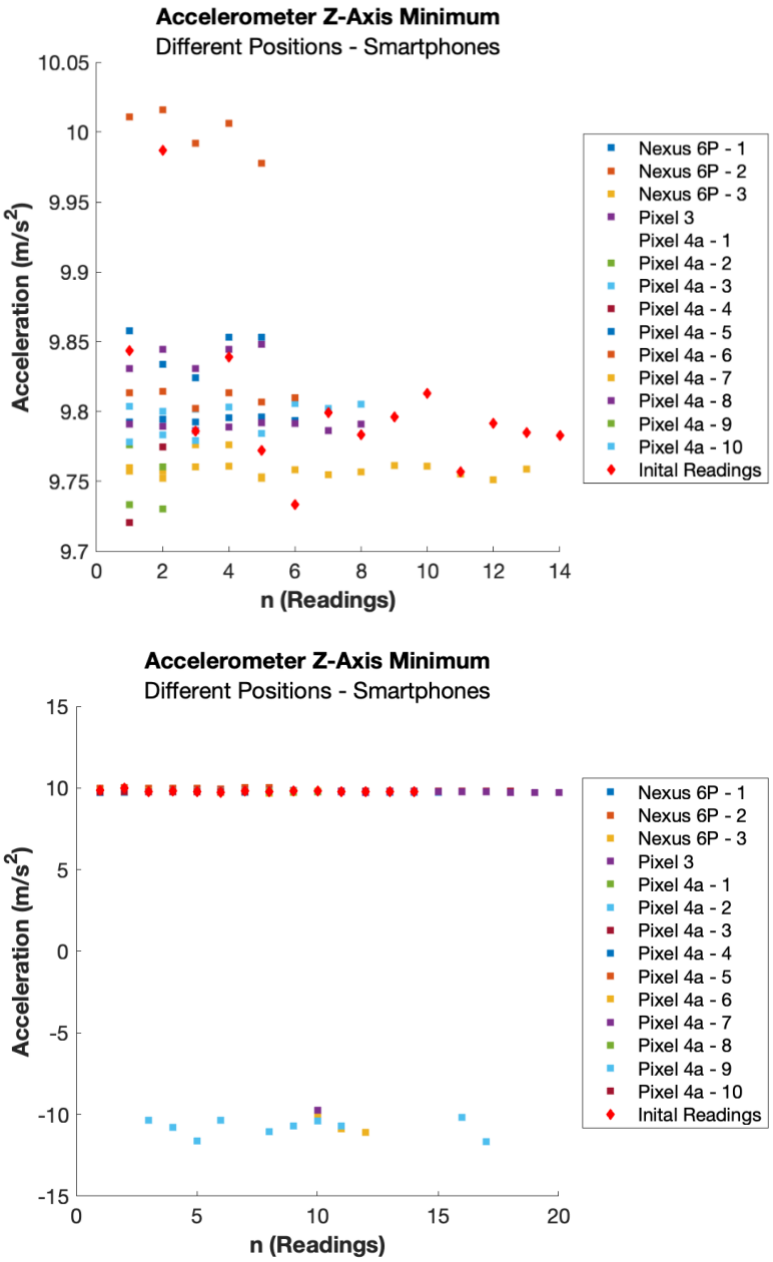


Figure 7.23 Smartphones – Different Positions: Z-Axis Minimum.

(Top) Accelerometer Z-Axis Minimum excluding iteration that places phone face down. (Bottom) Accelerometer Z-Axis Maximum including iteration that places phone face down.

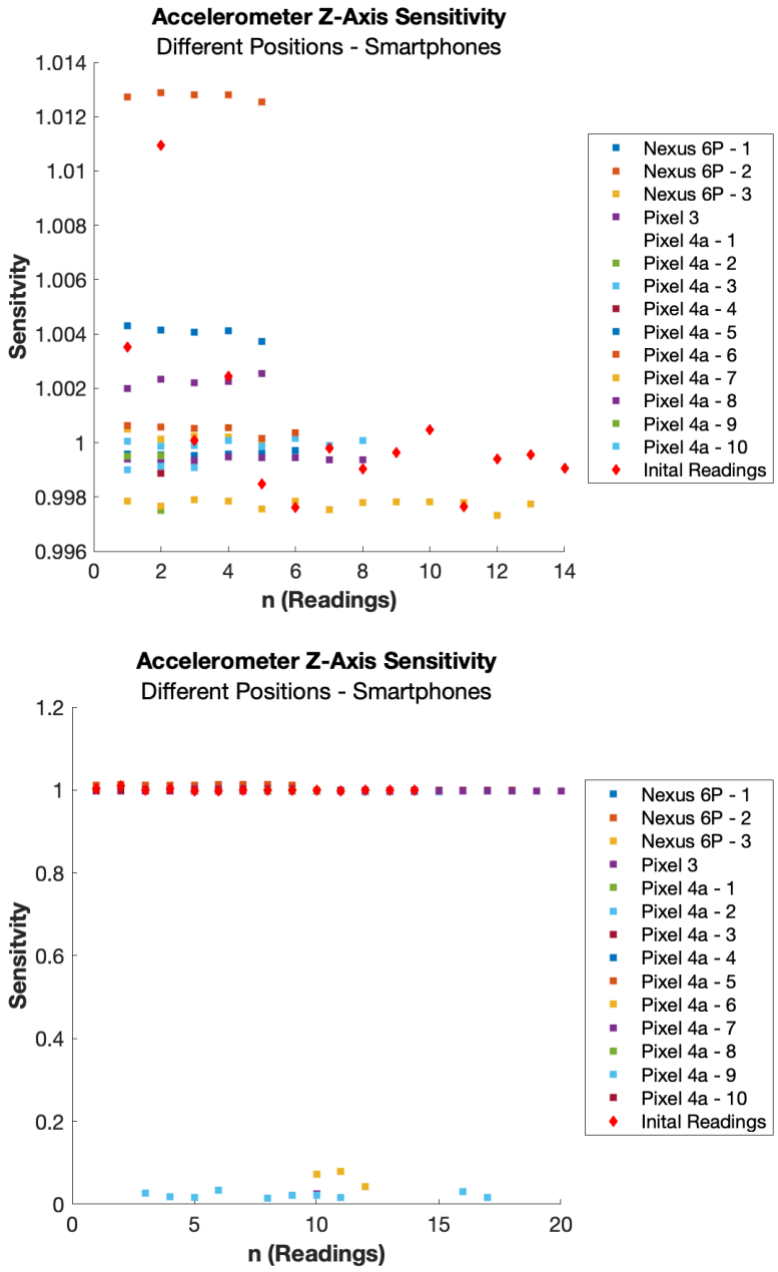


Figure 7.24 Smartphones – Different Positions: Z-Axis Sensitivity.

(Top) Accelerometer Z-Axis Sensitivity excluding iteration that places phone face down. (Bottom) Accelerometer Z-Axis Sensitivity including iteration that places phone face down.

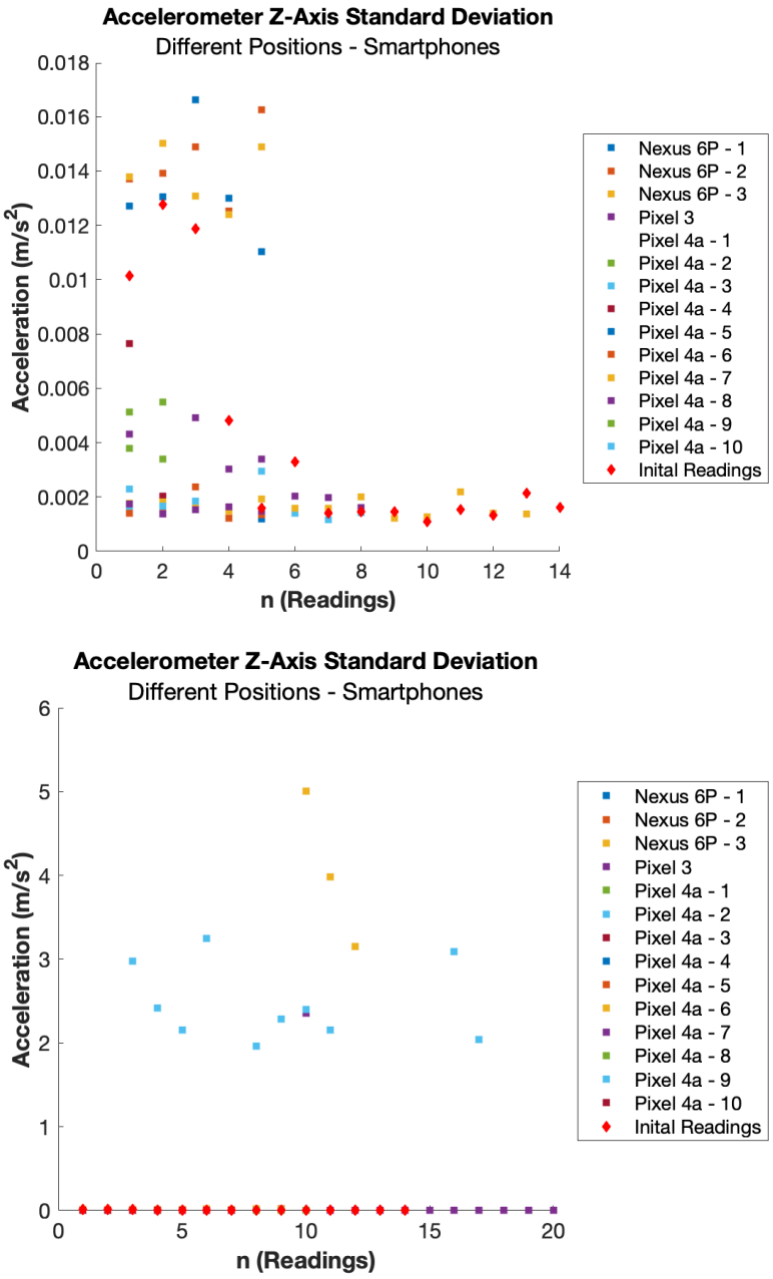


Figure 7.25 Smartphones – Different Positions: Z-Axis Standard Deviation.

(Top) Accelerometer Z-Axis Standard Deviation excluding iteration that places phone face down. (Bottom) Accelerometer Z-Axis Standard Deviation including iteration that places phone face down.

7.3.2. Realistic Scenario - Smartwatch Identification Data

The sensor data collected from smartwatches during the realistic experiment demonstrated that changing the position of the devices still provided consistent results on certain devices. The SmartWatch3 remained consistent in most attributes, this can be seen as a horizontal plot of points for the SmartWatch3. However, the data collected from the three Moto360 watches has shown there was overlap between the devices which led to device to be identified incorrectly by SOFIA. There was an outlier in the scatter plot which was an instance when the sensor data was collected while the device was upside-down.

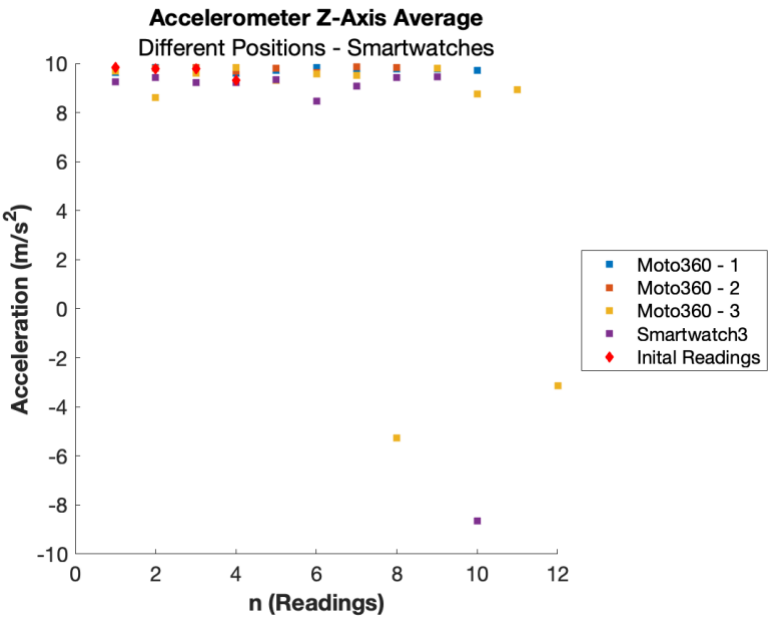


Figure 7.26 Smartwatches – Different Positions: Z-Axis Average.

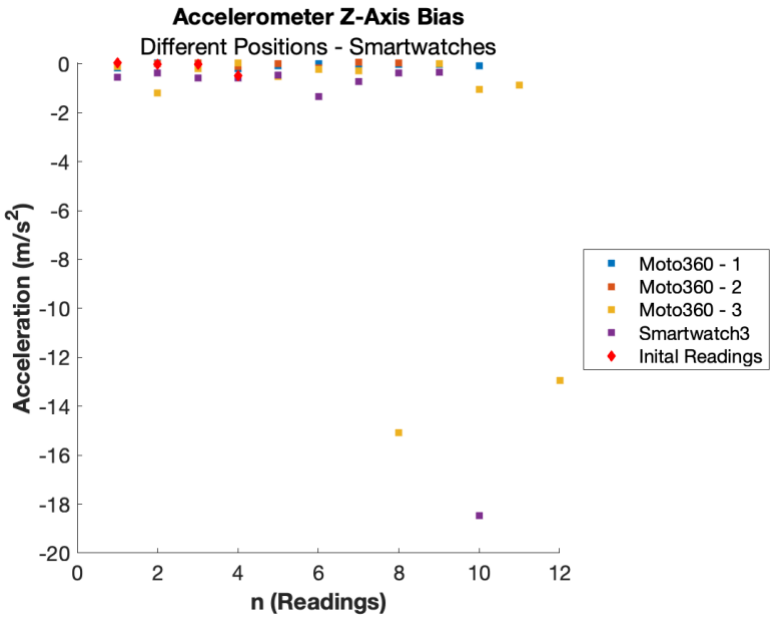


Figure 7.27 Smartwatches – Different Positions: Z-Axis Bias.

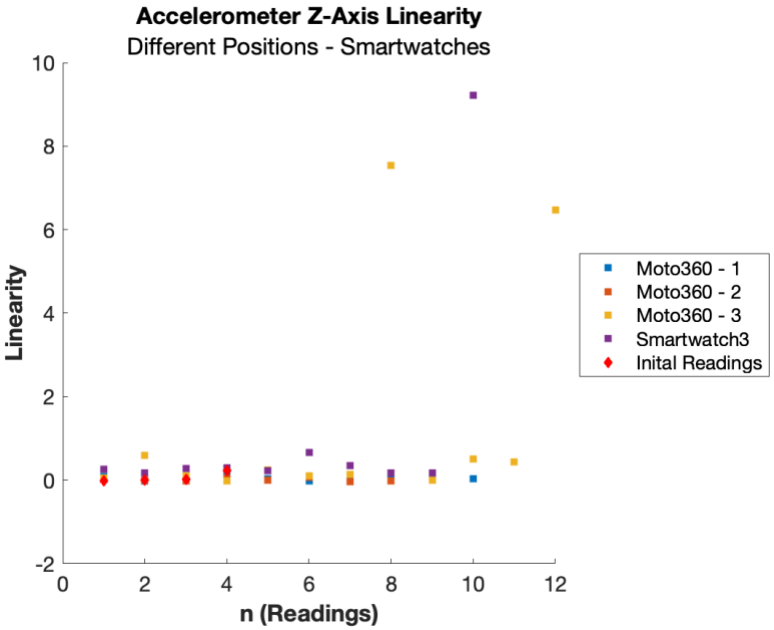


Figure 7.28 Smartwatches – Different Positions: Z-Axis Linearity.

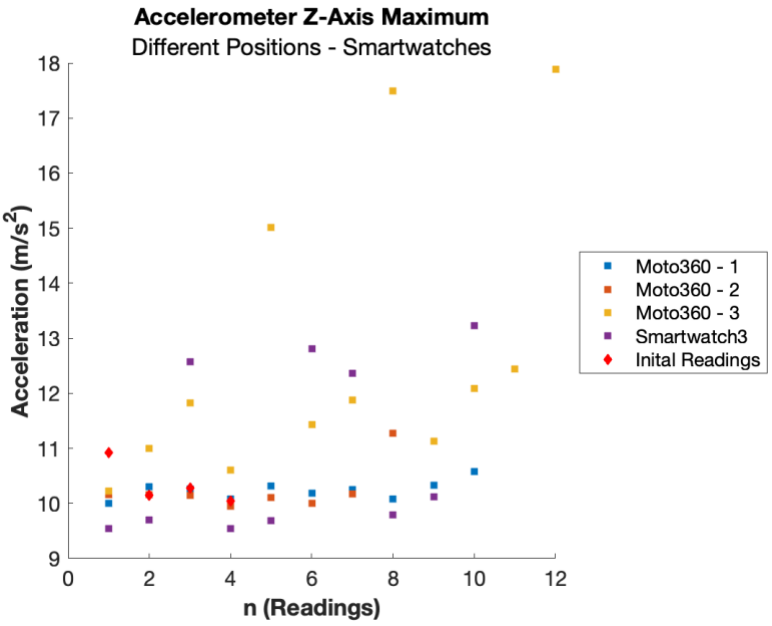


Figure 7.29 Smartwatches – Different Positions: Z-Axis Maximum.

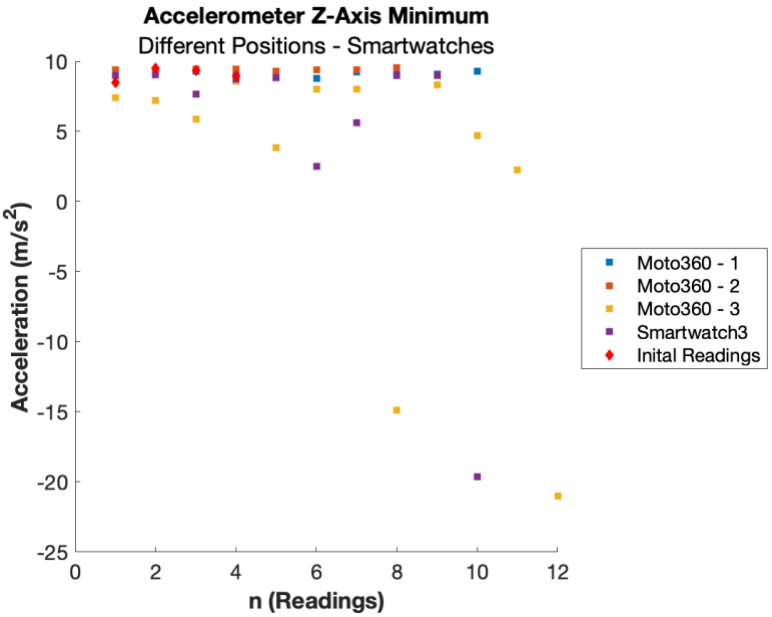


Figure 7.30 Smartwatches – Different Positions: Z-Axis Minimum.

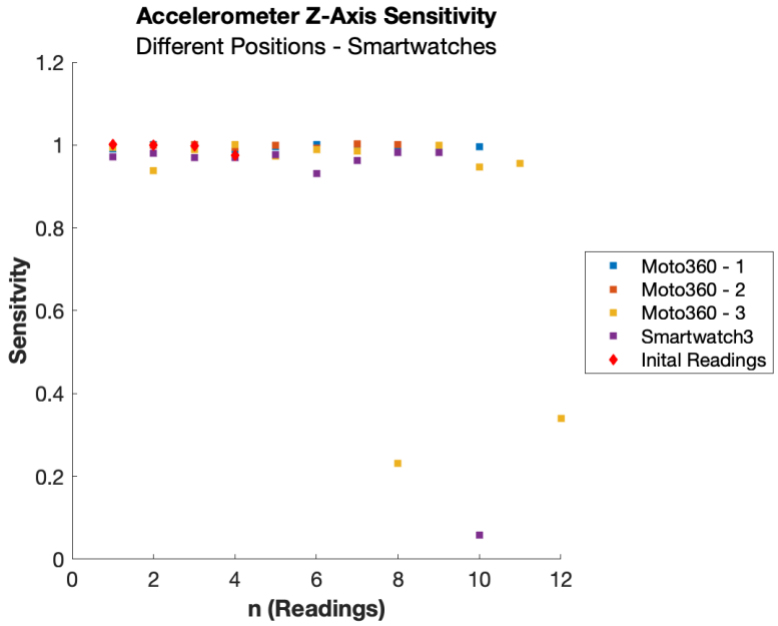


Figure 7.31 Smartwatches – Different Positions: Z-Axis Sensitivity.

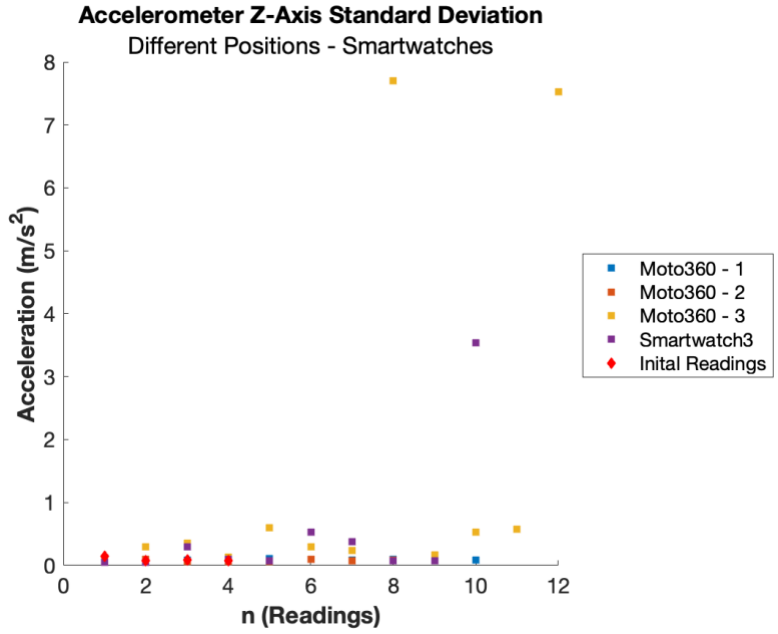


Figure 7.32 Smartwatches – Different Positions: Z-Axis Standard Deviation.

7.3.3. Realistic Scenario – Results

Predicted \ Actual (N = 140)	Pixel 3	Nexus 6P – 1	Nexus 6P – 2	Nexus 6P – 3	Pixel 4a – 1	Pixel 4a – 2	Pixel 4a – 3	Pixel 4a – 4	Pixel 4a – 5	Pixel 4a – 6	Pixel 4a – 7	Pixel 4a – 8	Pixel 4a – 9	Pixel 4a – 10	
Pixel 3	10	0	0	0	0	0	0	0	0	0	0	0	0	0	10
Nexus 6P – 1	0	9	0	0	0	0	0	0	0	0	0	0	0	0	9
Nexus 6P – 2	0	1	10	1	0	0	0	0	0	0	0	0	0	0	12
Nexus 6P – 3	0	0	0	9	0	0	0	0	0	0	0	0	0	0	9
Pixel 4a – 1	0	0	0	0	0	0	0	0	0	0	0	0	0	4	4
Pixel 4a – 2	0	0	0	0	1	6	1	1	1	1	1	3	2	1	18
Pixel 4a – 3	0	0	0	0	0	0	8	0	8	0	0	0	1	0	17
Pixel 4a – 4	0	0	0	0	0	0	0	0	0	0	0	0	0	2	2
Pixel 4a – 5	0	0	0	0	0	0	0	0	1	0	3	0	4	0	8
Pixel 4a – 6	0	0	0	0	0	0	1	0	0	9	0	0	0	0	10
Pixel 4a – 7	0	0	0	0	0	0	0	5	0	0	3	0	2	0	10
Pixel 4a – 8	0	0	0	0	9	4	0	0	0	0	0	7	0	0	20
Pixel 4a – 9	0	0	0	0	0	0	0	0	0	0	3	0	1	0	4
Pixel 4a – 10	0	0	0	0	0	0	0	4	0	0	0	0	0	3	7
	10	10	10	10	10	10	10	10	10	10	10	10	10	10	

Figure 7.33 Smartphones – Different Positions: Confusion Matrix Results.

Predicted \ Actual (N = 40)	SmartWatch3	Moto360 – 1	Moto360 – 2	Moto360 – 3	
SmartWatch3	10	0	0	0	10
Moto360 – 1	0	2	4	4	10
Moto360 – 2	0	4	2	2	8
Moto360 – 3	0	4	4	4	12
	10	10	10	10	

Figure 7.34 Smartwatches – Different Positions: Confusion Matrix Results.

Based on the smartphone data from Figure 7.33, demonstrated that the accuracy of SOFIA when tested on smartphone devices for the realistic scenario was 54%. There where 76 instances where a smartphone was correctly identified out of the 140 iterations. The misclassification rate of SOFIA when tested on smartphones was 45%, because there were 64 instances where a smartphone was incorrectly identified out of the 140 iterations. Compared to the controlled position experiment, the accuracy for smartphones dropped from 78% to 54%. This is due to an increase in sensor data overlap between devices which was more present when the device position was changed.

Moving on to the smartwatch data from Figure 7.34, demonstrated that the accuracy of SOFIA when tested on smartwatches for the realistic scenario was 45%. There were 18 instances when a smartwatch was correctly identified out of 40 iterations. The misclassification rate of SOFIA on smartwatches was 55% because there were 22 instances where a smartwatch was incorrectly identified out of 40 iterations. Compared to the controlled position experiment, the accuracy for smartwatches increased from 42% to 45%.

7.4. Countermeasures

Although sensor fingerprinting can be a reliable way to track mobile devices it is not perfect. As shown in the results there are condition that must be meet, such as the position of the device. If the owner of the device places their device in a different position that is not expected, then the identification accuracy will be reduced. Another method to countermeasure would be reducing the precision of the sensor. If the data is too scattered, then the identification is more likely to incorrectly identify the mobile device.

8. Conclusion

Based on the data collected from the SOFIA, accelerometer data alone can be used to device fingerprint mobile devices. The experimentation results demonstrated that it is possible to identify a mobile device, smart phones and smart watches, by collecting readings even only when the mobile device is facing upward.

To counter sensor fingerprinting on web browsers, one option is to disable JavaScript in the browser settings to prevent the data collection, however, it may not be practical. To counter sensor fingerprint on mobile applications, one option is to position your mobile device in arbitrary position to reduce the success of identification. Although countermeasures exist, the sensors on modern mobile devices will continue to provide more data which only increases the potential for sensor fingerprinting.

References

[1] Bojinov, Hristo, Michalevsky, Yan, Nakibly, Gabi, and Boneh, Dan. "Mobile Device Identification via Sensor Fingerprinting." (2014). Web.

[2] Chaoshun Zuo, Haohuang Wen, Zhiqiang Lin, and Yinqian Zhang. 2019. Automatic Fingerprinting of Vulnerable BLE IoT Devices with Static UUIDs from Mobile Apps. In Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security (CCS '19). Association for Computing Machinery, New York, NY, USA, 1469–1483. DOI:https://doi.org/10.1145/3319535.3354240

[3] Class UUID, Java Documentation, https://docs.oracle.com/javase/8/docs/api/java/util/UUID.html, website last visited on April 4, 2021

[4] Jonathan R. Mayer and John C. Mitchell. 2012. Third-Party Web Tracking: Policy and Technology. In Proceedings of the 2012 IEEE Symposium on Security and Privacy (SP '12). IEEE Computer Society, USA, 413–427. DOI:https://doi.org/10.1109/SP.2012.47

[5] LUKAS, J., FRIDRICH, J., AND GOLJAN, M. Digital camera identification from sensor pattern noise. Information Forensics and Security, IEEE Transactions on 1, 2 (2006), 205–214.

[6] Motion sensors, developer.android.com, https://developer.android.com/guide/topics/sensors/sensors_motion, website last visited on February 25, 2021

[7] Nikolay Matyunin, Nikolaos A. Anagnostopoulos, Spyros Boukoros, Markus Heinrich, André Schaller, Maksim Kolinichenko, and Stefan Katzenbeisser. 2018. Tracking Private Browsing Sessions using CPU-based Covert Channels. In Proceedings of the 11th ACM Conference on Security & Privacy in Wireless and Mobile Networks (WiSec '18). Association for Computing Machinery, New York, NY, USA, 63–74. DOI:https://doi.org/10.1145/3212480.3212489

[8] Pierre Laperdrix, Nataliia Bielova, Benoit Baudry, and Gildas Avoine. 2020. Browser Fingerprinting: A Survey. ACM Trans. Web 14, 2, Article 8 (April 2020), 33 pages. DOI:https://doi.org/10.1145/3386040

[9] T. Kohno, A. Broido and K. C. Claffy, "Remote physical device fingerprinting," in IEEE Transactions on Dependable and Secure Computing, vol. 2, no. 2, pp. 93-108, April-June 2005, DOI: 10.1109/TDSC.2005.26.

[10] Vikas Mishra, Pierre Laperdrix, Antoine Vastel, Walter Rudametkin, Romain Rouvoy, and Martin Lopatka. 2020. Don't Count Me Out: On the Relevance of IP Address in the Tracking Ecosystem. In Proceedings of The Web Conference 2020 (WWW '20). Association for Computing Machinery, New York, NY, USA, 808–815. DOI:https://doi.org/10.1145/3366423.3380161

[11] W. Wu, J. Wu, Y. Wang, Z. Ling, and M. Yang. 2016. Efficient fingerprinting-based android device identification with zero-permission identifiers. IEEE Access 4 (2016), 8073–8083. DOI:https://doi.org/10.1109/ACCESS.2016.2626395