# ABSTRACT

Title of thesis: **ENERGY HARVESTING MICROGENERATORS FOR BODY SENSOR NETWORKS**

Mehdi Dadfarnia, Master of Science, 2014

Thesis directed by: Professor John S. Baras
Department of Electrical and Computer Engineering
and ISR

Body sensor networks have the potential to become an asset for personalizing healthcare delivery to patients in need. A key limitation for a successful implementation of body sensor networks comes from the lack of a continuous, reliable power source for the body-mounted sensors. The aim of this thesis is to model and optimize a micro-energy harvesting generator that prolongs the operational lifetime of body sensors and make them more appealing, especially for personalized healthcare purposes. It explores a model that is suitable for harvesting mechanical power generated from human body motions. Adaptive optimization algorithms are used to maximize the amount of power harvested from this model. Practicality considerations discuss the feasibility of optimization and overall effectiveness of implementing the energy harvester model with respect to body sensor power requirements and its operational lifetime.

# ENERGY HARVESTING MICROGENERATORS FOR BODY SENSOR NETWORKS

by

Mehdi Dadfarnia

Thesis submitted to the Faculty of the Graduate School of the
University of Maryland, College Park in partial fulfillment
of the requirements for the degree of
Master of Science
2014

Advisory Committee:
Professor John S. Baras, Chair/Advisor
Associate Professor Mark Austin
Research Assistant Professor Huan Xu

The work of Mehdi Dadfarnia was supported by
the National Institute of Standards and Technology (NIST)
through grant 70NANB11H148 and through an internship.

# Acknowledgments

I would like to express my sincere gratitude to Dr. Kamran Sayrafian, Professor John Baras, Professor Mark Austin, and Professor Paul Mitcheson, who have all advised and guided me throughout this research effort.

I would like to thank my family, especially my parents, for their care and love throughout my entire life.

I would also like to give special thanks to IS, SB, HJ, EK, and FG for their friendship and support throughout this past year.

# Table of Contents

# List of Figures

# List of Tables

# List of Acronyms

| | |
|---|---|
| BSN | Body Sensor Network |
| CDRG | Coulomb-Damped Resonant Generator |
| CFPG | Coulomb-Force Parametric Generator |
| CPS | Cyber-Physical System |
| MSD | Mass-Spring-Damper |
| Q-V | Charge-Voltage (relationship) |
| VDRG | Velocity-Damped Resonant Generator |

# Chapter 1: A Systems-Level Background on Body Sensor Networks

Before delving into details about body sensors and energy harvesting architectures that could assist them to make their usage practical, this chapter takes a step back to introduce the body sensor networks on a system-level scale. Benefits of body sensor networks are discussed, as well as their potential impact to the future of healthcare systems and current limitations for reaching this potential.

## *1.1: A Shift in Healthcare Delivery*

The future of healthcare is moving towards a shift in the patient-healthcare provider relationship, with an emphasis towards safer, more efficient, higher quality personalized patient care [2,4]. This is essential to meet the needs of deteriorating cardiac health and soaring healthcare costs [1]. However, due to recent technological advances, personalized healthcare delivery has only recently been seen as a viable realization. Specifically, these technological advances can be attributed to improved wireless technology, miniaturized sensors, and greater processing power [3].

In [4], 6 areas for improving current healthcare practices are outlined with the following properties:

- Patient-centeredness: providing compassion, empathy, and responsiveness to needs, values, and expressed preferences of the individual patient.
- Effective: providing services based on scientific knowledge to all who could benefit, and refraining from providing services to those not likely to benefit.
- Safe: avoiding injuries to patients from care that is intended to help them.

- Timely: reducing waiting-time for the patient to see the healthcare provider and minimizing potentially harmful delays for both those who receive and provide healthcare.
- Efficient: avoiding wastefulness, including waste of equipment, supplies, ideas, cost, and energy.
- Equitable: providing care that does not vary in quality because of personal characteristics such as gender, ethnicity, geographic location, and socioeconomic status.

The same report [4] cites the need for a systems approach to improve the complex healthcare delivery system. The report outlines 10 rules for the future redesign of (personalized) healthcare systems with the aim to improve healthcare delivery implementations with respect to the 6 areas outlined above. The 10 rules for the redesign of healthcare delivery are as follows [4]:

- Care is based on continuous healing relationships: Patients should receive care whenever they need it, and not limited to face-to-face visits. This implies that the health care system must be responsive at all times, and access to care should be provided over the Internet, by telephone, and by other means in addition to in-person visits.
- Care is customized according to patient needs and values: The system should be designed to meet the most common types of needs, but should have the capability to respond to individual patient choices and preferences.
- The patient is the source of control: Patients should be given the necessary information and opportunity to exercise the degree of control they choose over

health care decisions that affect them. The system should be able to accommodate differences in patient preferences and encourage shared decision-making.

- Knowledge is shared and information flows freely: Patients should have unfettered access to their medical information and to clinical knowledge. Clinicians and patients should communicate effectively and share information.
- Decision-making is evidence-based: Patients should receive care based on the best available scientific knowledge. Care should not vary illogically from clinician to clinician or from place to place.
- Safety is a system property: Patients should be safe from injury caused by the care system. Reducing risk and ensuring safety require greater attention to systems that help prevent and mitigate errors.
- Transparency is necessary: The system should make available to patients and their families information that enables them to make informed decisions when selecting a health plan, hospital, or clinical practice, or when choosing among alternative treatments. This should include information describing the system's performance on safety, evidence-based practice, and patient satisfaction.
- Needs are anticipated: The system should anticipate patient needs, rather than simply react to events.
- Waste is continuously decreased: The system should not waste resources or patient time.
- Cooperation among clinicians is a priority: Clinicians and institutions should actively collaborate and communicate to ensure an appropriate exchange of information and coordination of care.

## *1.2: Personalized Healthcare and Body Sensor Networks*

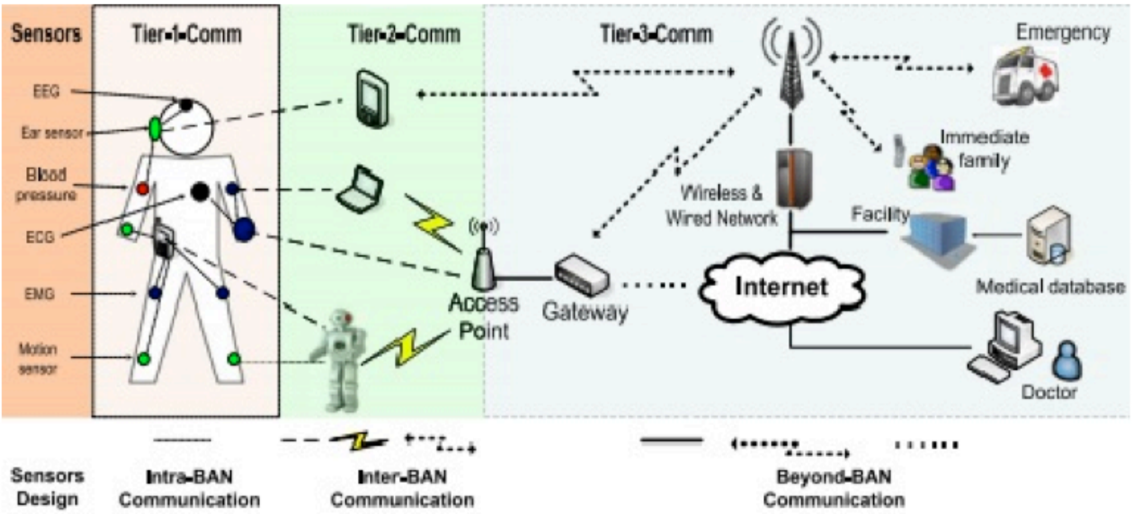


Figure 1.1: Illustration of a typical Body Sensor Network deployed for mobile health [9]

Up-and-coming technologies are explored to look for solutions that can help implement the redesign of the improved healthcare delivery system. Body Sensor Networks (BSN) is a technology that can be integrated with mobile healthcare delivery (see Figure 1.1), in order to ensure quick, accurate, safe, transparent, and personalized healthcare for the patient.

In the context of using body sensor networks for improving healthcare delivery, it is insightful to define the structure of a body sensor network that is integrated with a mobile healthcare delivery system. This way, a clearer picture is given prior to discussing the benefits of utilizing this technology for improved mobile healthcare delivery. The following are the three structural subsystems that may make up the architecture and devices needed for this application of body sensor networks:

- Sensor Nodes: These are the nodes applied on or inside the human body in order to collect the physiological data needed to make a medical decision about the patient. They should be able to communicate to the Hub, and also communicate between other body sensor nodes. Each sensor node is capable of incorporating a stack of sensors that can read information or take in data from the patient.

- Hub: This device collects all the data coming from the sensor nodes, and relays them to the server. It also acts as an interface to the patient to provide them with information regarding their health status, the system status, or communications with healthcare providers (such as physicians or nurses).
- Server: This subsystem stores and analyzes all data coming from different patients (each with their own hub device). The server database can be accessed with the patient hub, a healthcare provider/physician device, or potentially through the internet with a secure interface.

In addition to the structural make-up of a BSN, the bounds in which the technology interacts with can further elaborate on the context of using BSN's for healthcare monitoring. The system boundary that encompasses these 3 subsystems can be defined in 3 different scopes: a solution space, a design space, and a problem space [7]. The solution space, or the components that are directly designed by stakeholders, is the body sensor network integrated with mobile healthcare delivery. That is the Sensor Nodes, Hub, and Server. The solution space is embedded within the design space. The design space includes the patient body, healthcare provider (physicians, nurses, etc) medical decisions, and maintenance staff in addition to the solution space. The problem space is even broader, and also includes all the components in the design space. The problem space may also include legal processes and institutions, system hackers and intruders, and also the environmental conditions that affect communications and hardware structure.

With the given structural and system boundary context, body sensor networks have the potential to improve healthcare delivery systems according to the aforementioned 10 rules for healthcare redesign (which are in turn aiming to achieve the 6 areas of improvements identified for healthcare systems). A discussion is presented below for laying out the six benefits and outcomes of implementing body sensor networks for healthcare delivery. The relationships of these six outcomes with improving healthcare delivery, by referring to the aforementioned 10 rules for healthcare delivery redesign, are also discussed.

1. Educate patients about their disease with a personal profile [5].
    a) Relation to redesigning an improved healthcare delivery system: *Knowledge is shared and information flows freely.* By educating patients about their disease and allowing them access to their personal health profiles, the patients gain access to knowledge and information about their health.
2. Ensure communication of patient's health status between patient and physician, especially feedback and communications for medical decision-making on behalf of the healthcare provider to the patient [5, 6].
    a. Relation to redesigning an improved healthcare delivery system: *Care is based on continuous healing relationships; Care is customized according to patient needs and values; Decision-making is evidence-based; Waste (and time) is continuously decreased*. This system-level goal insures that

the patient is in constant, continuous communication with the physician or medical healthcare provider through the BSN, integrated with the healthcare delivery system. Since the healthcare provider has access to continuous records of the patient's health, they can assess a more customized remedy for the patient rather than analyzing their health based off a "snapshot" of time during a face-to-face visit. Not only does this larger array of information allow the healthcare provider to make a more customize-able conclusion for the patient's health, but their diagnosis and remedy will be more evidence-based. Fortunately, this larger array of information does not cut out from the patient's time. They should be able to do their daily activities as the BSN obtains and retains information from them. This is in contrast to check-ups and face-to-face visits where the patient and healthcare provider have to schedule a certain amount of time a day to meet each other.

3. Provide patients with tools/treatments to manage their disease [5].

    a. Relation to redesigning an improved healthcare delivery system: *The patient is the source of control; Needs are anticipated.* By giving the patient access to tools that can help manage their disease, it becomes possible to give them a source of control, or rather the perception of a source of control. The patient will be able to receive real-time updates about their health, and then be able to react accordingly. It heightens their sense of responsibility on their own anticipated health needs.

4. Ensure accessible communication of patient's health status to patient device [5].

    a. Relation to redesigning an improved healthcare delivery system: *The patient is the source of control; Transparency is necessary.* Body Sensor Networks enable the patient with access to a device that will allow them to see their health data records and diagnoses. Having convenient access to such data gives the patient a sense of control over their own health status, and provides transparency to their physiological information.

5. Efficient, comprehensive, and transparent information management for BSN data, including electronic medical records [4, 6].
    a. Relation to redesigning an improved healthcare delivery system: *Knowledge is shared and information flows freely; Transparency is necessary*. The goal is to have efficient and transparent data management of the BSN for healthcare delivery purposes. With the patient records accessible to the patient at all times, their physiological data and electronic health records is shared with them and also provides more transparency to diagnoses done by physicians and healthcare providers.
6. Empower patient privacy with informed consent, legal protection, and proper regulations [6].
    a. Relation to redesigning an improved healthcare delivery system: *Safety is a system property.* Privacy is a major concern for using BSN, as is security. For a BSN to be widely available and widely used by patients, it must first build a trust with them in order to ensure that all their health records and data are secure and private. Without this sense of comfort, the BSN will either come across as unsafe to use, prone to identity-theft. By

ensuring legal rights and protection of BSN users (both patients and physicians), it will become both more suitable and much safer for widespread use.

### ***1.3: Stakeholders, Actors, and Use Cases for Healthcare Delivery with BSN***

An insight on the benefits of using BSN for healthcare delivery is gained with exploring the details in the BSN system boundary. This includes the interactions between the BSN, and the stakeholders and actors within its system boundaries. The stakeholders identified for an implementation of body sensor networks for healthcare delivery will drive the body sensor network development process. There are possibly three categories of stakeholders: Users, System Architects, and System Designers.

Users include patients and healthcare providers (physicians, nurses, and such). These are the primary customers who will use the body sensor network system for the purpose of healthcare provisions. Requirements are derived from these users for the System Designers to develop the system. System Designers include systems scientists and engineers who design a platform for the body sensor network, depending on what requirements they see the Users need. System Architects include engineering and Information Technology firms. These stakeholders implement the systems that are designed by System Designers. They construct the structure and architecture of the body sensor network for the Users. It should be noted that there is no single ideal design for all BSN implemented in a healthcare delivery system, and as a result a platform utilized by the System Architects can help perform trade-off on parameters to give a customized, balanced solution to a specific medical condition [8].

Furthermore, six possible actors have been identified in using the body sensor network. These include the following:

- Maintenance
- Legal System
- Patient
- Healthcare Provider
- BSN Provider (Manufacturer)
- Intruder

The maintenance actors include the technology specialists who are in charge of making sure all the communications are secure. These include technicians necessary to set-up the system and fix any component failure in case any part of body sensor network becomes faulty. Also, the legal system comes into play to make sure that patients are given their security and privacy. The healthcare providers include physicians or nurses who are assigned to patients in order to make sure their health status are okay. They also provide feedback that the patient may need or request. BSN providers include any manufacturers or distributors of Servers, Hubs, and Sensor Nodes. The intruder plays a role in trying to breach the security of the body sensor network.  These actors and their relationship to possible body sensor network use cases are depicted in Figure 1.2.

Six major use cases are identified through looking at relevant literature [1, 3-4, 6] and the six outcomes of implementing body sensor networks as a means of healthcare delivery (described in section 1.2). They involve putting together the system, maintaining component usability, providing patients with health monitoring and data protection, allowing storage of health records, and allowing communication between the patient and physician. These use cases are as follows:

1. System Setup
2. Maintain Components
3. Provide User-Data Protection
4. Monitor Health
5. Store Accessible Health Records
6. Allow Patient-Healthcare Provider Communication

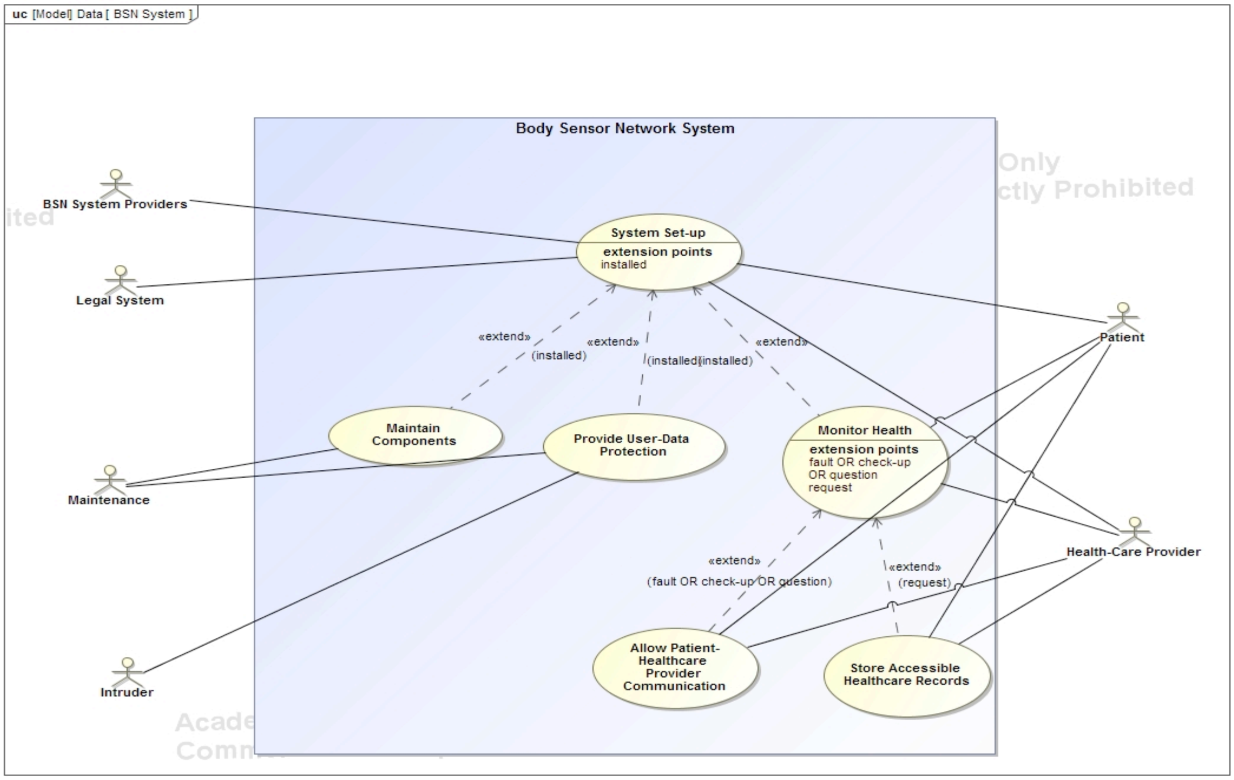


Figure 1.2: Use Case Diagram for Healthcare Delivery with BSN

The use case diagram in Figure 1.2 illustrates the associations of each of the system's use cases with its primary actors. Secondary actors are also signified in the use case descriptions (Appendix A).

The choice of these use cases comes from the intended behaviors and functionalities of the BSN-integrated healthcare delivery system. After the system is setup (Use Case 1), the patient’s health is monitored (Use Case 4), where their health records are stored in an accessible manner (Use Case 5). Simultaneously, components in the BSN are being maintained (Use Case 2) and security is enabled for data protection (Use Case 3). If needed, the physician and patient can also come into communication with each other (Use Case 6). The use case descriptions can be found in Appendix A.

It should be noted that the purpose of the use cases is to look for requirements and see if there are any limitations or challenges to meet system goals (such as ensure seamless provision of healthcare to the patient). For body sensor networks, this includes looking for requirements and challenges needed to make them a driving force for the personalization of healthcare. Use cases are broken down into use case descriptions, use case goals, behavioral diagrams, and traceability measures in order to ensure that the use case effectively finds requirements, limitations, or challenges that need to be refined in the system.

Use case descriptions are made for each use case scenario. They allow stepping through and exploring each scenario, looking for goal-driven behaviors among the system actors. The use case descriptions include the post-conditions and pre-conditions, goals of the use case, primary actors, derived requirements, a primary flow of events, and alternative flow of events. Pre-conditions entail the conditions that must hold for the use case to begin. Post-conditions take into account the conditions that hold once the flow of events has been completed [10].

The goals of each use case lay out the objectives that the BSN, integrated with the healthcare delivery system, is expected to meet for the primary system actors in each use case. The derived requirements are higher-level functionality requirements. They are derived from behaviors or requirements that are meant to accomplish goals for a use case.

Activity diagrams, sequence diagrams, and state machine diagrams are used to assist the development of the primary flows, as these diagram types do aid in the generation of behavioral and structural requirements and structure interface requirements, respectively [10]. A major benefit gained from using these diagrams stems from making it easier to catch logical flaws in the primary flow's statements, and thus helping to refurnish the primary flow.

### ***1.4: Conclusion of Systems Analysis of Body Sensor Networks & Necessity of Prolonging Their Operational Lifetime***

The focus of this thesis isn't to give a detailed system-wide outlook on body sensor networks, but to focus on a particular aspect of body sensor networks that may enable these systems to become an asset to personalizing and improving healthcare. For this reason, traceability measures are omitted and use case descriptions are laid out in Appendix A. Instead, the focus is on the resulting challenges and limitations of body sensor networks that were realized at the conclusion of the use case descriptions and system-to-component requirements traceability.

One of the outcomes that result from exploring use case descriptions is becoming aware of the dire needs and limitations of the BSN. In this regard, use case descriptions show that it is important to maintain a reliable, continuous stream of energy in order to seamlessly power the body sensor networks. In order to make the body sensor network operable, low consumption power and long battery time is key for use [11]. The rest of the thesis shows a novel way to prolong the operational lifetime of a body-mounted sensor node by utilizing a micro-energy harvesting power generator. Physical geometric and design parameters of this micro-generator device are presented, along with the relationship of the device's architecture with the ambient environment of the device. Conclusions are then made about the energy harvesting devices and their effectiveness for prolonging the operational lifetime of a sensor node.

# Chapter 2: Energy Harvesting Mechanisms from the Human Body

The main goal of the remainder of this thesis is to focus on finding a way to prolong the operational lifetime of a body sensor. As a consequence, this would potentially enable the successful commercial application of such devices. It would minimize the cost and convenience penalty associated with sensor replacements, recharging batteries, or even system failures due to a dying power source. Once a suitable method that would be able to prolong the operational lifetime of a body sensor is found, the maximum harvested power from this method is investigated through optimizing the device architecture. The end goal is to use energy-harvesting methods, also known as energy-scavenging, to meet a substantial portion of the power consumption levels needed by body sensor applications.

## *2.1: Harvesting Energy from Ambient Environment of Sensors*

Without attempting to alter the battery size on the wearable sensors, the aim is to prolong the operational lifetime of sensor networks by harvesting energy from the ambient environment of body sensors. This entails the process of capturing and storing energy from the environment surrounding the human body. The sources of energy available to power the sensor nodes include: far-field electromagnetic radiation, near-field coupling, light (solar cells), radio-frequency (RF), thermal gradients, and human kinetic motion. There are advantages and disadvantages with respect to each of these energy harvesting methods.

Far-field electromagnetic radiation [12] and near-field coupling [13] can be effective power sources, however the major problem with these sources is that they require building new infrastructures as they are methods that deliver power to the device. The infrastructure issue and its associated costs are the key reason why it is advantageous to look at the other four sources of energy.

Using a light source with solar cells is a more commercially established energy harvesting solution [3], and due to this may be more desirable for use as an energy source. The major problem with relying on light as an energy source is that reliance on a light source to be available to a body-mounted sensor is very unreliable as the availability of a light source to the solar cells would be unpredictable. As for utilizing RF radiation as a source of energy to be harvested for the body sensor, studies [15] have shown that there is a lack of availability of energy to be harvested with current extraction methods. Harvesting energy from human body thermal gradients, on the other hand, have been proven to be successful with some watches [16], but are not a feasible option for harvesting energy on a miniature-scaled size that reflect the size of the microgenerators [17].

Harvesting energy from the human body, on the other hand, seems to be one of the most convenient and attractive solutions for wearable wireless sensors in healthcare applications [18]. However, energy harvesting from the human body still has its own challenges due to the random nature of human kinetic motion. Contemplating kinetically energy harvesting microgenerators as a feasible solution to prolong body sensor node operational lifetime requires the following considerations: the typical power consumption of the device, usage pattern, and device size.  These considerations may limit the power density that a micro-generator is capable of harvesting.

Power consumption levels of a body-mounted sensor depend on the functionalities present on the sensor node. A sensor node will typically use power for the following 3 functions [19]: the sensor itself, signal processing circuitry, and the wireless data link. The signal data rates for a sensor node would be used for collecting heart rates, blood pressure, temperature, and other bodily characteristics. Monitoring health does not require a high clock rate. A rate between 1 and 10 samples per minute may be enough to collect necessary information to understand the human body's status. Considering that each sample needs no more than 16 bits to store its value, data rates between 16 and 80 bits per minute sound like a feasible option for relaying the human body signals. Without going into the details in [19], it is possible that these very low data rates will not burden the sensor, signal processing circuitry, and wireless data links to require total power levels of more than 1μW.

Usage patterns depend on the human body motions that affect the sensor node. The human body produces motions that are variable and unpredictable, and an efficient body-mounted, kinetic micro-generator needs an architecture that adheres to the nature of this type of motion.

The size of the energy harvester influences the kinetic energy harvester's transducer mechanism. Depending on the transducer mechanism, the efficiency of converting mechanical work to power changes with the microgenerator size. Since the sensor node is itself miniaturized, the architecture of the energy harvester must be on the millimeter-scale. The literature [17, 19-21] shows that transducer mechanisms for such small-scaled kinetic energy harvesting architectures are efficient for implementation and thus feasible to construct.

## *2.2: Architecture of Kinetic Energy Harvesters Effective for Harvesting Energy from Human Body Motions*

Different architectures for kinetic energy harvesters exist, and the goal of this section is to narrow down to architectures that would be most efficient to use on wearable or body-mounted sensor nodes. The overall generic structure of an energy harvester, depicted in the figure below [22], provides a more organized view to study the energy-harvester characteristics and design parameters ideal for human kinetic motions. The generic electromechanical block diagram of an energy-harvesting micro-generator consists of a mass-spring-damper (MSD) system, a transducer, and an interfacing power-processing circuit.

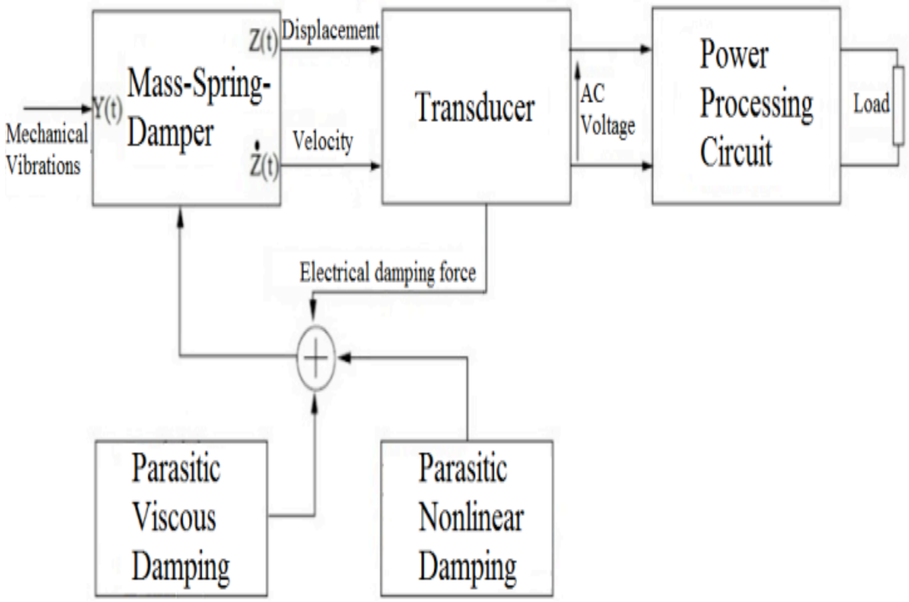


Figure 2.1 Generic Electromechanical Block Diagram of an Inertial Micro-generator [22]

The type of transducer model affects the type of (electrical) damping force that is incorporated into the MSD system. Although there will be mentions of interfacing the mechanical generated power from the MSD model to the power processing circuitry, the remainder of this section intends to focus on the MSD model and its transducer-dependent damping force. Both viscous and nonlinear parasitic damping forces could also be incorporated into the MSD and transducer design. An example of where these nonlinear affects arise from is the collisions that the proof mass has as it springs from one end of the MSD structure to the other. For simplicity, considering these effects are neglected in this thesis as they have minimal effect on the overall dynamics of the MSD & transducer systems. In fact, assuming that no mechanical power yields from collisions between the mass and MSD architectural frame in turn leads to an underestimated amount of generated power [27].

The design of a kinetic microgenerator architecture suitable for extracting power from human body motion depends on its MSD model, the spring, the damping factor, and the transducer mechanism (affecting the operation of the damping force). As for the MSD model, two main mechanical models for kinetic micro-generators are known as direct-force and inertial.

The MSD model for the direct-force method can be seen in Figure 2.2, from [17]. It makes use of a direct application of force on the MSD model to drive the proof mass, *m,* with a force $f_{dr}(t)$ strong enough to displace the mass with a distance $z(t)$. Though displaced, the proof mass stays in between the upper and lower bounds of its displacement area, denoted with +/- $Z_l$. The spring constant $k$ affects the mobility of the proof mass against the opposing force of the transducer's damper, and thus produces the mechanical power from this MSD model.

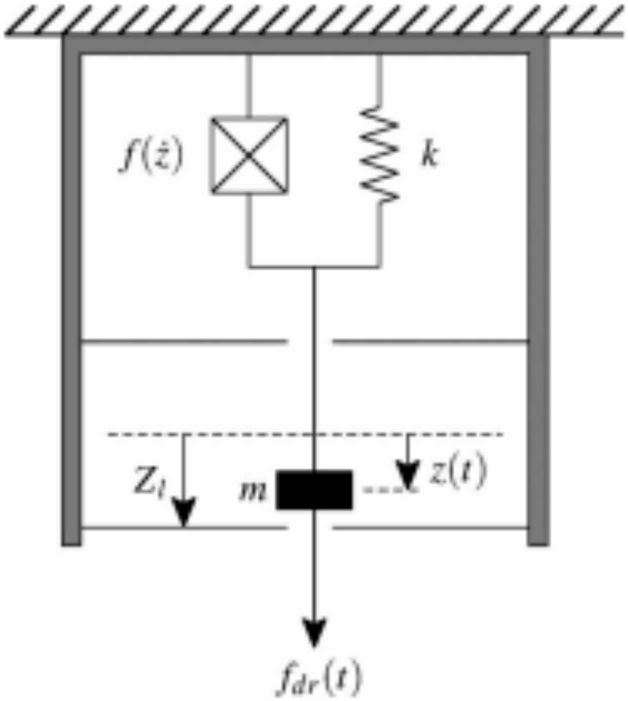


Figure 2.2: Generic model of a direct-force micro-generator [17]

The MSD model for the inertial method can be seen in Figure 2.3, from [17]. The inertial forces from the acceleration and deceleration of human kinetic motion causes the proof mass *m* to bounce up and down, with respect to the spring constant *k*, from one end-stop +/- $Z_l$ to another. These end-stops, +/- $Z_l$, represent the upper and lower bounds of the path of the proof mass within the MSD architectural framework. The displacement of the proof mass from its rest position relative to the frame is denoted by *z(t)*. The absolute motion of the frame is *y(t)* and that of the proof mass is *x(t) = y(t) + z(t)*. Energy is converted when work is done against the transducer's damping force *f(z')*, opposing the motion based on the direction of proof mass velocity, *z'(t)* [23].

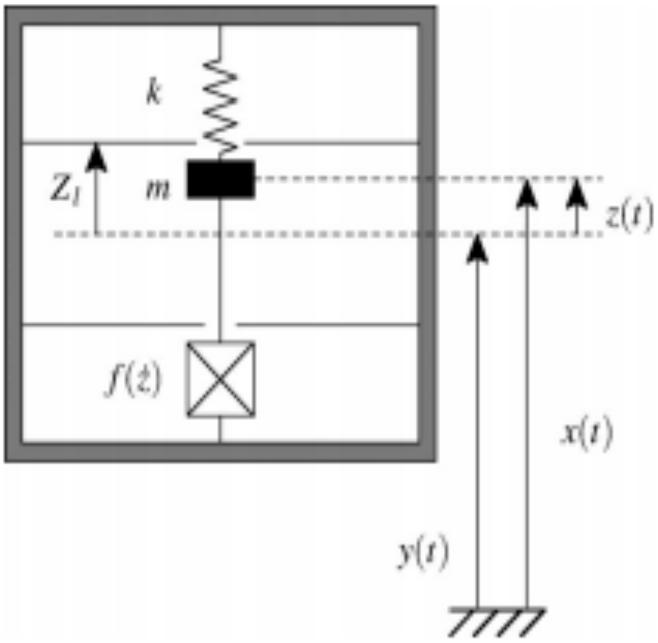


Figure 2.3: Generic model of an inertial micro-generator [17]

Unlike micro-generators that utilize a direct application of force, their inertial counterparts require only one point of attachment to the moving proof mass structure. This allows for a greater mounting flexibility and also a greater degree of miniaturization that is more ideal for wearable, body-mounted sensors [17]. The direct-force MSD model, though, does have its own unique applications with human body motions. For example, they are ideal for being placed at the heel of a shoe as they can make use of the mechanical contact between the foot and ground to apply force on its damper.

## ***2.3: Effective Transduction Mechanisms for Harvesting Energy from Human Body Motion***

In addition to the kinetic microgenerator's MSD model, its transduction method is important for determining the amount of generated mechanical power from the model.

The transduction method determines the characteristic and operating principles of the damping force in the MSD model. Typical transduction methods for energy harvesting micro-generators include: electromagnetic, electrostatic, and piezoelectric transductions [17]. The damping force characteristic of these 3 main transduction methods are as follows [24]:

- Electromagnetic transduction: the damping force can be altered by the resistance of a sensor load connected to the electromagnetic coil.
- Electrostatic transduction: the damping force can be set by the electric field between two capacitor electrodes.
- Piezoelectric transduction: the damping force can be altered by the tuning impedance between the terminals of the piezoelectric material.

The transduction mechanism depends on the source of motion from which energy is being harvested as well as the size of the energy-harvesting generator. Since the micro-generators that would be useful for wearable applications are miniaturized, electromagnetic transduction methods become inefficient at harvesting energy from milli- or micro-scaled energy harvester sizes [17]. On the other hand, piezoelectric materials are more suitable for MSD models that make use of a direct application of force on the device rather than inertial forces (use of piezoelectric transduction with direct force of heel strikes is explored in [25]).

Electrostatic transduction does not have these aforementioned issues, and instead it has advantages with respect to optimizing its damping force. Optimizing the damping force for electrostatic devices is more straightforward than the other transduction methods. But furthermore, the electrostatic damping force can be dynamically optimized which is important for operating under highly variable sources of motions such as a human body [24].

Specifically, electrostatic transduction makes use of mechanically-variable capacitors. The capacitive plates are formed between each side of the proof mass and the corresponding end-stop that it moves into or away from. The plates are separated by a source of motion (human body motion), and opposite charges across the capacitor plates attract each other to form the moving variable-capacitor [17]. Work is done and energy is generated when the capacitor plates (proof mass relative to the fixed end-stops) pull apart with respect to each other. The mechanical work done to separate the plates converts into electric power when interfaced with power-processing circuitry.

Switched electronic circuitry connections between the transducer and circuitry involve a reconfiguration of the microgenerator through operation of switches, at different parts of the power generation cycle (generally known as pre-charge, generate, and dis-charge stages of the cycle [26]). With switched electrostatic transducers, the variable capacitors between the proof mass and the end-stops may be realized with two different motions: sliding motion or a perpendicular motion between the capacitive plates. Three possible electronic transduction architectures [20, 22] that utilize sliding and perpendicular motions between capacitive plates is depicted in Figure 2.4.

- In-plane overlap type: the proof mass moves vertically with respect to the end-stops, where the capacitive plates are formed between the fingers of the end-stops and fingers of the proof mass. These capacitive plates slide parallel with respect to each other.

- In-plane gap closing: the proof mass moves horizontally with respect to the end-stops, where the capacitive plates are formed between the fingers of the end-stops and fingers of the proof mass. These capacitive plates move perpendicularly with respect to teach other.
- Out-of-plane gap closing: A proof mass with a capacitive side moves to and from (perpendicularly) a fixed end-stop.

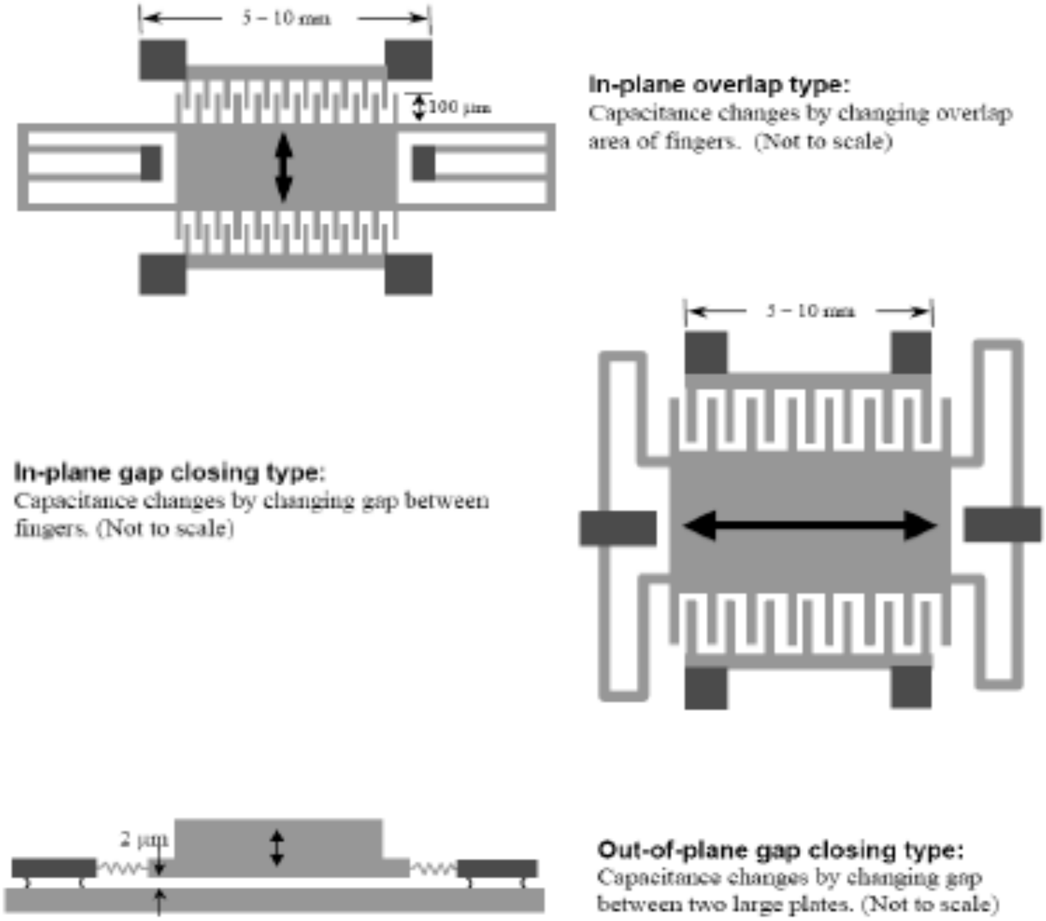

5 – 10 mm
100 μm
In-plane overlap type:
Capacitance changes by changing overlap area of fingers. (Not to scale)
5 – 10 mm
In-plane gap closing type:
Capacitance changes by changing gap between fingers. (Not to scale)
2 μm
Out-of-plane gap closing type:
Capacitance changes by changing gap between two large plates. (Not to scale)

Figure 2.4: Three architectures for implementing electrostatic micro-generators [22]

With the perpendicular motion, there is steady overlap between the two capacitive plates. The capacitive plates have constant charge, but are not connected to any voltage source. Because the same amount of charges are held as separation increases (during perpendicular motion), the energy density and electric field strength decrease while the volume of the electric field and the (potential) energy in the electric field increases. As separation further increases, additional potential energy is stored in the increased volume of the electric field (voltage increases faster than capacitance decreases and thus potential energy increases) [22]. Once the energy is generated, the capacitor is discharged and the charge on the capacitors is restored to restart the cycle. The Q-V graph (based on the relationship *Charge=Capacitance*Voltage*) in Figure 2.5 shows the charge and voltage relationships for a perpendicular motion (i.e. constant charge) variable capacitance, as it goes through the following 3 stages to generate power [26]:

- First, the device is pre-charged to a low voltage in the first part of the cycle by making a connection to a voltage source.
- Then, the plates are disconnected from the source and separated under constant charge during the generation part of the cycle.
- Finally, the capacitor is discharged and the capacitance is increased, ready for the cycle to restart.

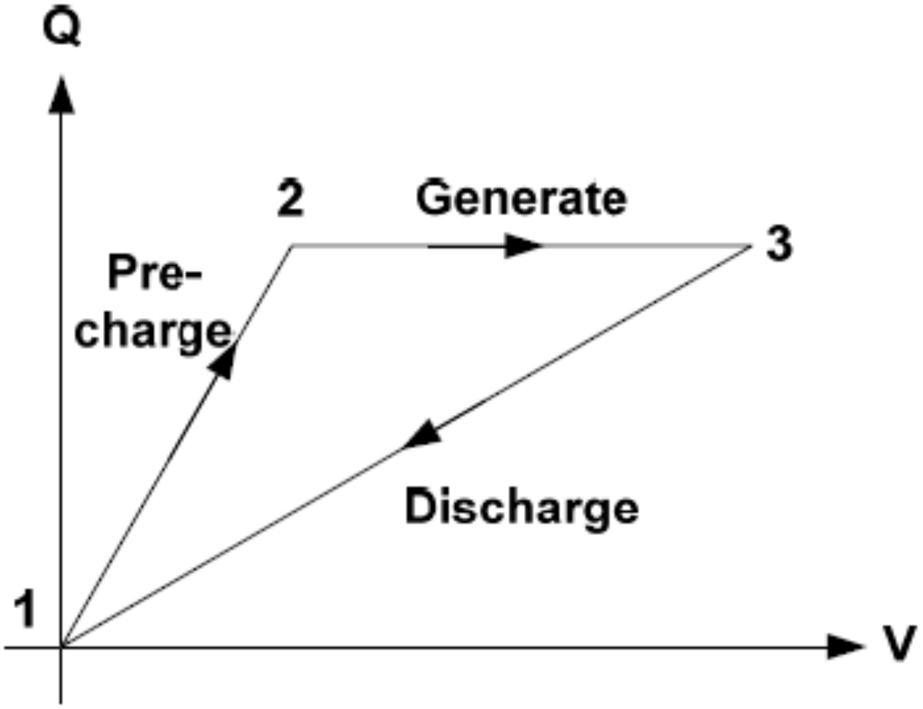


Figure 2.5: Q-V graph for a switched, constant charge electrostatic micro-generator [26]

With the sliding motion on the electrostatic device's capacitive plates, there is steady perpendicular gap between the two capacitive plates. Typically, the capacitive plates have constant voltage on them as they are connected to a fixed voltage source. The voltage and plate separation is constant, therefore the electric field is constant, which in turn means the energy density is constant. Since the volume of the field decreases, the current is forced to flow into the source. Since the capacitance decreases (fewer charges) and voltage is constant, the potential energy is being transferred to the voltage source circuitry as the charges are flowing there. Switches disconnect the capacitor from the voltage before the capacitance is increased at constant charge, ready for the cycle to restart [22]. The Q-V graph in Figure 2.6 shows the charge and voltage relationships for a sliding motion (i.e. constant voltage) variable capacitance, as it goes through the following 3 stages to generate power [26]:

- First, the capacitor is pre-charged to a set voltage whilst at high capacitance.
- Then, whilst connected to a voltage source, the capacitance is reduced, forcing charge back into the voltage source (generation).
- Switches then disconnect the capacitor from the voltage source before the capacitance is increased at constant charge, ready to restart the cycle.

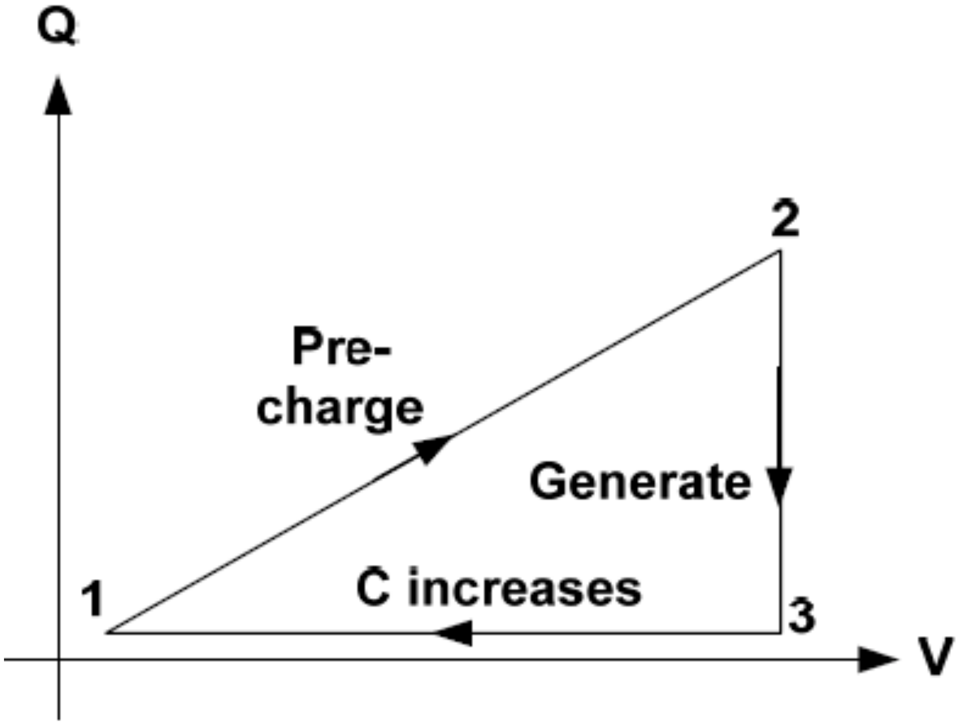


Figure 2.6: Q-V graph for a switched, constant-voltage electrostatic micro-generator [26]

It should be noted that the areas of the triangles formed in the Q-V graphs in Figures 2.5 and 2.6 represent the energy that is generated with each switched cycle.

It should be noted that the capacitive plates in either case (sliding or perpendicular motion) can assume double-sided or single-sided operation. Single-sided operation is when energy is only being extraction in one direction of the proof mass motion (and thus is not being extracted on the proof mass's return stroke), whereas double-sided operation extracts energy from both directions of motion [23]. In the remainder of this paper, the models and simulation assume double-sided operation.

Analyses show that the maximum effectiveness of these harvesters [31] are dependent on whether the capacitive plates are operating by perpendicularly moving to and from each other or by sliding past each other. It is apparent that capacitive plates that are moving past each other with a sliding motion (i.e. operating with a fixed, constant voltage) operate with an acceptable effectiveness over a much wider envelope than perpendicularly moving plates (i.e. operating with constant charge). This makes the choice of sliding motion/constant voltage a preferred implementation for switched electrostatic transducer.

Although switched electrostatic transducer designs have so far seemed to be the most suitable for applications involving energy harvesting from human body motions, it should be noted that they do have their limitations. These electrostatic transducers require a pre-charge in order to start their energy generation cycles. This means that an active pre-charge circuitry system or electret device must supplement the switched electrostatic transducer. This comes with extra control circuitry complexity [17].

Yet, the miniaturized, millimeter-scaled device size required of the human body-mounted micro-generator leads to the conclusion that a switched, electrostatic transducer is the most suitable to harvest energy.  Furthermore, the usage patterns of an energy harvester mounted to a wearable device also leads to the conclusion that an inertial MSD framework is most efficient to extract energy.

However, the unpredictable usage patterns of a wearable device or body-mounted sensor also has implications on the spring constant *k* of the MSD model. An energy harvester making use of a spring has an internal resonant frequency and generates power by characterizing the vibrations that are inflicted on the device. Vibration energy harvesters are employed in resonant systems as they generate maximum power when the device's characteristic, resonant frequencies are tuned to match the ambient vibration frequency [21].

Therefore, energy harvesters are not efficient when used to generate power from non-resonant acceleration/deceleration forces inflicted on the device, i.e. human kinetic motion.  This non-resonant, unpredictable human kinetic motion makes tuning the resonant frequency of the spring or spring-like structure to the frequency of the human body motion very difficult. As a result, an effective micro-generator that harvests energy from human body motion should avoid making use of a spring or spring-like structure and adhere to effectively extracting energy from non-resonant sources with other architectural structures [27].

### ***2.4: A Non-resonant, Electrostatic, Kinetic, Inertial Micro-Energy Harvester***

A non-linear, non-resonant, snap-action energy harvesting micro-generator architecture is introduced in [28]. This model, known as the Coulomb-Force Parametric-Generator (CFPG), makes use of a MSD model where no spring exists. With no spring, the model no longer has an intrinsic resonant frequency. Instead, this architecture makes the proof mass snap away from an end-stop towards the opposite side of the frame only if the (externally-induced) acceleration of the frame is greater than the electrostatic, Coulombic damping (or holding) force that holds on the proof mass to an end-stop.  The Coulombic damping force must be less than the product of the mass and external acceleration for the proof mass to break away from the frame. This implies that the maximum energy can be obtained if the proof mass only moves during the peak of external acceleration exerted on the CFPG device [29, 30].

The CFPG MSD (or MD) model utilizes the inertial operating principle and the electrostatic Coulombic damping force. The transducer (damping force) can be implemented as the force between parallel capacitor plates moving perpendicularly to each other with a constant charge or between parallel sliding plates fixed with a constant voltage [30]. This implies that the CFPG dampers make use of switched, electrostatic transduction. In [23], a study compares power generated by a CFPG with 2 other resonant, linear architectures, Velocity-damped resonant generators (VDRG) and Coulomb-damped resonant generators (CDRG). Subject to a harmonic source motion with frequency $w$ and external source motion amplitude $Y_o$, the figure below [23] illustrates that the CFPG is superior to the other 2 architectures when the source frequency is relatively low (as in the infrequent human body motions) and the source amplitude is relatively high. This makes CFPG devices more ideal for small devices operating in larger ranges of motions.

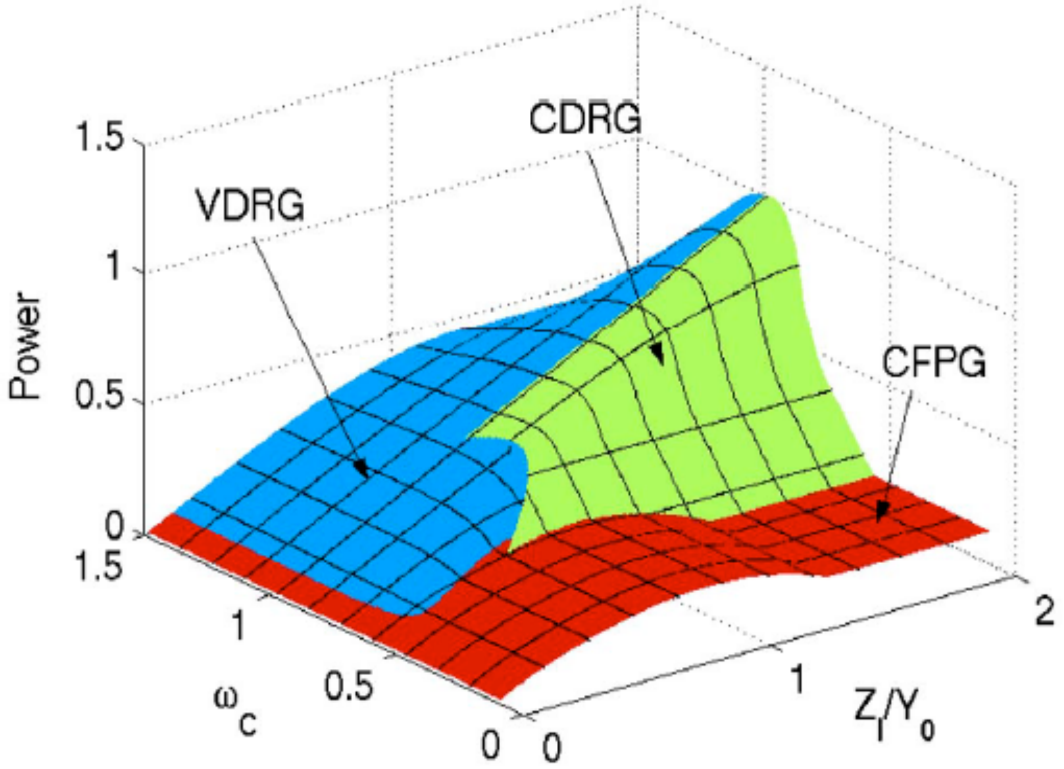


Figure 2.7: Generator architecture power harvested comparison with respect to input acceleration frequency and ratio between distance of proof mass full flight and input acceleration motion amplitude [23]

As a result, it can be concluded that the CFPG model's use of an inertial, non-resonant, non-linear, spring-less MSD model in conjunction with a switched, electrostatic transducer (or damper) makes it suitable for generating power from the human body. For the remainder of this thesis, the CFPG architectural framework is used to develop a Matlab/Simulink model that can study the temporal behavior of the generated mechanical power, have a more accurate estimation of the amount of power generated from various human movements, and optimize the design parameters of the energy harvesting micro-generator.

Details about the Coulomb-Force Parametric-Generator architecture and its energy harvesting capabilities compared to other types of micro-generators can be found in [21-13, 28]. Fabrication and testing of these device structures are discussed in [19, 20, 22, 30]. Interfacing the micro-generator's MSD model and transducer models with the power processing circuitry is discussed in [19, 22, 24, 31].

Studies to simulate and optimize the architectural parameters of energy harvesting micro-generators and prolong the operational lifetime of sensor nodes using these optimized architectures are discussed in [19-20, 27, 32-34].

In [27], the authors point out that the CFPG device only has one parameter to optimize, the electrostatic Coulombic damping force (or holding force). They point out that the maximum power achieved for the CFPG can be found by sweeping this parameter across a range of values. The Coulomb force is essentially decreased to a point where the damping force just allows displacement of the proof mass from the MSD frame.

In [33], the authors describe methods for estimating harvested energy from acceleration traces. Although they use a resonating inertial harvester model, they attempt to optimize their harvester parameters. Relevant to the discussion of harvesting energy from kinetic human body motions, they use their model to examine a dataset of acceleration traces from various common motions and activities, as well as long-term human mobility.

In [32, 54], the authors provide statistical analysis of measurements taken from 40 test subjects over a period of 8 hours during the day, and use this information, along with the operational architecture of a CFPG device, to determine the distribution of harvestable energy. The aim of this study was to observe whether kinetic energy generated by typical human forearm motion could be a source of energy to prolong the lifetime of wearable medical sensors.

The study in [18] aims to utilize the acceleration traces from [32] to build a simulation-based CFPG model to further study this device. An enhanced Simulink model of the CFPG device was shown to examine the temporal behavior of the generated power. Such a dynamic model provides a more accurate estimate of the amount of power generated from various human motions. It also allows for further optimization of the micro-generator's design parameters and the characterization of the input acceleration. The contribution in this thesis expands on the methods and results introduced in [18].

Chapter 3 discusses the details regarding the mathematical modeling and Simulink implementation of the CFPG device. Chapter 4 describes an optimization problem that relates the input acceleration, the electrostatic Coulombic damping force, and output generated power. Power gains between the optimized and non-optimized models are compared. Finally, chapter 5 concludes the thesis and remarks on future work that remains for understanding the effectiveness of energy harvesting to benefit the use of wearable medical devices and body-mounted sensor nodes.

# Chapter 3: CFPG Modeling and Simulation

## *3.1: A Mathematical Model for the CFPG Device*

A mathematical model is necessary to accurately simulate the temporal behavior of the CFPG device's MSD model as well as the amount of mechanical power the model is capable of generating. As mentioned in the previous chapter, parasitic damping and proof mass collision effects of the MSD model dynamics are not taken into account, despite reason to believe that they could result in more power generated by the energy harvester. The following non-linear differential equation has been specified as a model to capture the dynamics of the MSD system in a CFPG micro-generator [28]:

$$my''(t) = -mz''(t) - F \ast sign(z'(t)) \qquad (1)$$

In this equation, *m* represents the proof mass, *y(t)* represents the motion of the generator frame with respect to the inertial frame (*y''t)* is the second derivate of *y(t)* and indicates the input acceleration), *z''(t)* is the proof mass acceleration, *F* represents the electrostatic Coulombic damping (or holding) force, and *sign(z'(t))* represents the sign, or equivalently the direction, of the proof mass velocity (i.e. *z'(t)*). The sign function in this equation makes sure that the Coulombic damping force takes on the opposite sign of the proof mass velocity. This indicates that work is done when the transducer's Coulombic damping force opposes the motion based on the direction of proof mass velocity, *z'(t)*, as the damping force attempts to hold the proof mass to an end-stop. It should be noted that the proof mass position, *z(t)*, is limited between the MSD frame's end-stops, at distances $+Z_l$ and $–Z_l$.

Values from previous studies on the CFPG structure [32, 35] determine the length and thickness of the MSD model. The length is set at 10 mm, with a cross-sectional area of 100 $mm^2$. The thickness of the MSD module depends on an *alpha* coefficient. This coefficient is set to have a value of 0.1, and the thickness itself is equal to the product, *alpha**length. The proof mass takes roughly half the volume of the MSD module, with the other half allowing for the movement of the proof mass between the module's end-stops [17]. As a result, given the chosen length, the distance between the two end-stops in the MSD frame is 5 mm. This means that the two end-stop limits are $+Z_l$ = 2.5 mm and -$Z_l$= -2.5 mm. Using the same studies for consistency [17, 32, 35], the mass density is set at $19.3*10^3$ kg/$m^3$, giving the model a total proof mass of 0.965 g.

From the discussion in the previous chapter, it is known that no power is generated while the proof mass is stuck to an end-stop, and that power is only generated when the proof mass makes a full-flight from one end-stop to the other. When this occurs, the dynamic model represented by Equation 1 generates power. Equation 2 can compute mechanical power, as follows:

$$P(t) = F * z'(t). \tag{2}$$

*P(t)* represents the instantaneous generated mechanical power from the MSD module, *F* represents the same electrostatic Coulombic damping (or holding) force as in Equation 1, and *z'(t)* represents the velocity of the proof mass.

### ***3.2: Simulink Representation of CFPG Mathematical Model***

Simulating the dynamics of Equations 1 and 2 into Simulink requires handling a few more considerations about the physical CFPG dynamics. Although Equation 1 mathematically captures the phenomena occurring in the CFPG's MSD system, a few constraints on the physics behind the proof mass motion and resulting power generation complicates the direct implementation of this equation.

Direct implementation of Equation 1 does not guarantee that power is only generated when the proof mass makes a complete flight between the end-stops of the MSD frame. If the proof mass makes an incomplete flight, that is the proof mass goes back to the end-stop that it had last been released from, the sign of the Coulombic electrostatic damping force component in Equation 1 should not change. Otherwise, a sign change would mean that real power is still being generated for the incomplete flight. Instead, at the point where the proof mass is changing directions to head back to the end-stop that it had become released from, the simulations require to generate negative, reactive power [28]. This requirement cancels out the positive, real power that had been generated when the proof mass had departed its end-stop up until the point it started turning back. The end result of an incomplete flight is a net power of zero. The *sign(z'(t))* component in Equation 1 should only change when the proof mass reaches an end-stop. It should not be able to switch signs in the case of an incomplete proof mass flight.

Replacing the *sign(z'(t))* function in Equation 1 with the relay function in Equation 3 makes sure the model represents the proper sign changes as the Coulombic electrostatic damping force opposes the proof mass motion. This function is a typical Preisach model of hysteresis based on the position of the proof mass.

$$\mathrm{relay}(z(t)) = \begin{cases} 1, & if\ z(t) = -Z_i \\ -1, & \text{if } z(t) = Z_i \\ k, & \text{if } -Z_i < z(t) < Z_i \end{cases}, \qquad (3)$$

In Equation 3, k=1 if the proof mass position *z(t)* was last outside the range $-Z_l < z(t) < Z_l$ when $z(t) = -Z_l$. Likewise, k=-1 if the proof mass position *z(t)* was last outside the range $-Z_l < z(t) < Z_l$ when $z(t) = Z_l$. Figure 3.1 shows a graphical representation of Equation 3.

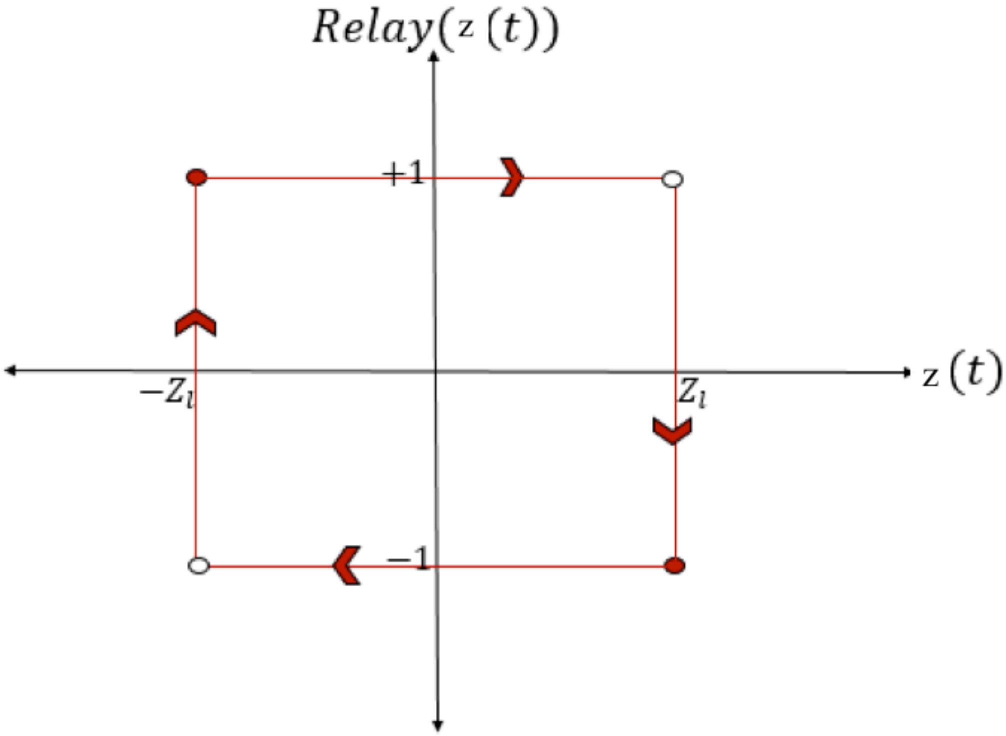
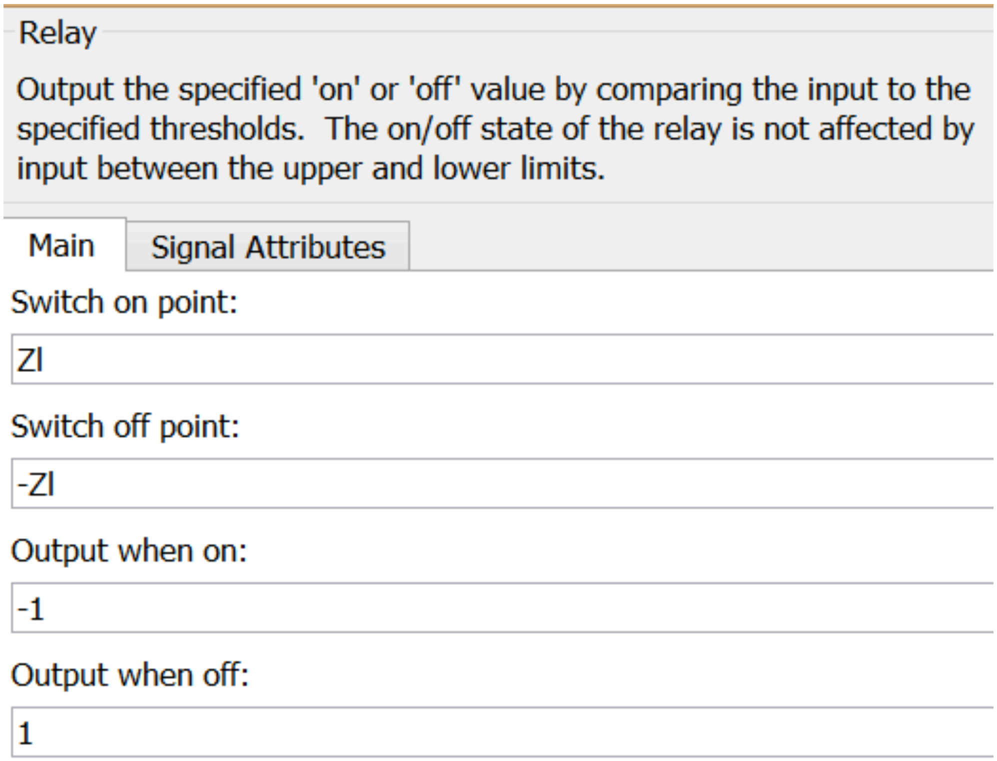


Figure 3.1: Relay Function

Note that, unlike the sign function in Equation 1, the relay function depends on the position of the proof mass rather than depending on the proof mass velocity. However, the output sign of this relay function still depends on the proof mass velocity direction. In Equation 3, it is the position of the proof mass that indicates the proof mass's direction of its velocity. Once the proof mass reaches an end-stop, it may only depart to one direction, towards the other end-stop. Thus the proof mass direction of velocity is clearn when it is about to depart from its end-stop position.

To understand exactly how Equation 3 meets the requirement for the sign of the Coulombic damping force to only change when the proof mass meets end-stops, consider orientations chosen for the negative and positive directions for the proof mass velocity. Moving downwards away from the upper end-stop $+Z_l$ is considered the negative direction and moving upwards away from the lower end-stop $-Z_l$ is considered the positive direction. Whenever the proof mass position $z(t)$ reaches the upper end-stop, $+Z_l$, the relay function will output a -1. Replacing the sign function in Equation 1 with the relay in Equation 3 causes the sign of the Coulombic damping force in Equation 1 to be +1. This is because the mass will only depart this upper end-stop to go downwards, a negative velocity direction, and the Coulombic damping force works against the direction of proof mass motion. The sign of the Coulombic damping force will remain the same until the mass reaches the lower end-stop, $-Z_l$. Once the proof mass position $z(t)$ reaches $-Z_l$, the relay function will output a +1. Replacing the sign function in Equation 1 with the relay in Equation 3 causes the sign of the Coulombic damping force in Equation 1 to be -1. This is because the proof mass will depart this lower end-stop only to go upwards, a positive direction of velocity. This results in the Coulombic damping force to work against the direction of proof mass motion. This cycle continues as the proof mass moves between the two end-stops. Details of the implemented Simulink “Relay” block function is depicted in Figure 3.2.



Relay

Output the specified 'on' or 'off' value by comparing the input to the specified thresholds. The on/off state of the relay is not affected by input between the upper and lower limits.

Switch on point: ZI

Switch off point: -ZI

Output when on: -1

Output when off: 1

**Figure 3.2: Relay Function Simulation Specifications**

Equation 1 is modified with the relay function in Equation 3. The resulting Equation 4 is used for simulating the CFPG's MSD system.

$$my''(t) = -mz''(t) - F * relay(z(t)) \qquad (4)$$

The Simulink implementation of Equation 4 is depicted in Figure 3.3. The input of this diagram is the external acceleration *y''(t)* exerted on the CFPG device. Each input runs through this model for a range of Coulombic damping force values. For each damping force value, a temporal instantaneous power output is generated as well as plots for the proof mass position, velocity, and acceleration. It should be noted that the integrator's limits specify the end-stop boundary limits allowed by the device geometry. Observations of the average power generated for each damping force value can be made, as well as observing the damping force value that yields the highest power output for a given external acceleration input. The Matlab code that runs the Simulink diagram in Figure 3.3 is provided in Appendix B.

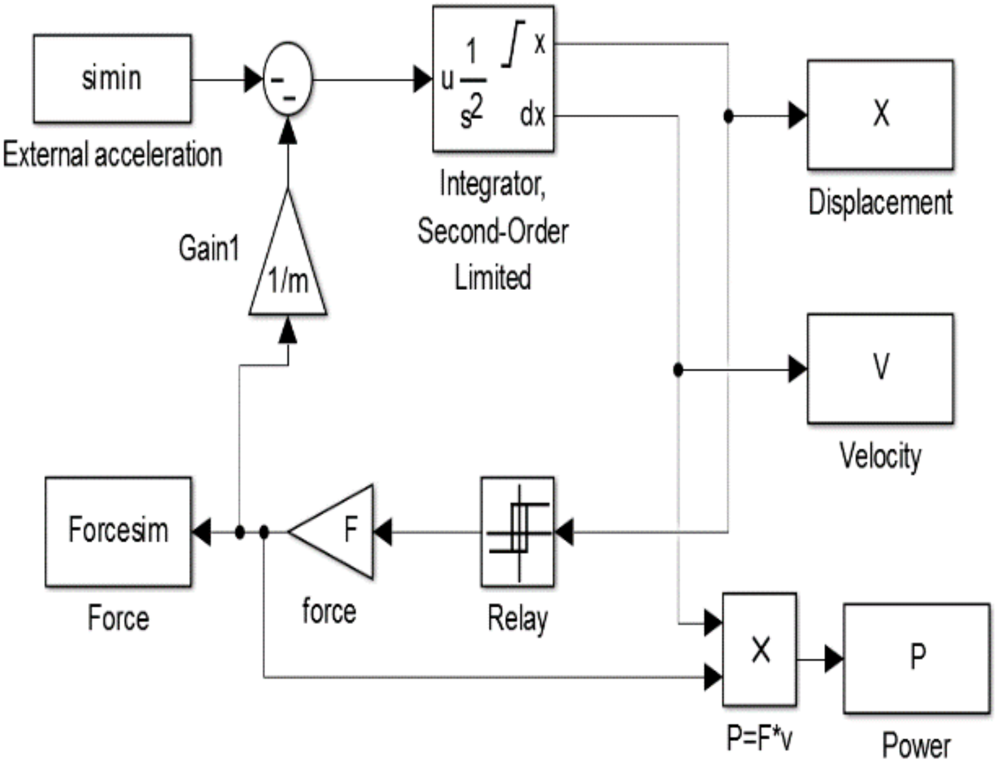


Figure 3.3: Simulink block-diagram implementation of CFPG model

Several test scenarios are used to verify the modified differential equation and the Simulink model. Section 3.3 discusses these verification test scenarios in greater detail.

### *3.3: Functional Operation of CFPG Model*

The CFPG model should only generate power when the proof mass makes a complete flight between the MSD frame’s end-stops. To test the model for this, the temporal behavior of generated mechanical powers and proof mass positions are considered for the following cases:

- Proof mass makes complete flight between two end-stops.
- Proof mass makes incomplete flight, and it returns back to end-stop it had just been released from.

These two cases are investigated by using artificially-generated sinusoidal acceleration inputs. Using an artificially-generated sinusoidal acceleration input allows control over the amplitude and frequency of the sinusoidal inputs. This in turn allows tweaking the signal's magnitude and frequency and having control over when the input signal allows the proof mass to snap away from an end-stop. An example artificially-generated sinusoidal acceleration input is depicted in Figure 3.4. Inputting sinusoidal acceleration signals into the CFPG model in Figure 3.3 gives insight onto the power generated with respect to different input acceleration magnitudes and frequencies.

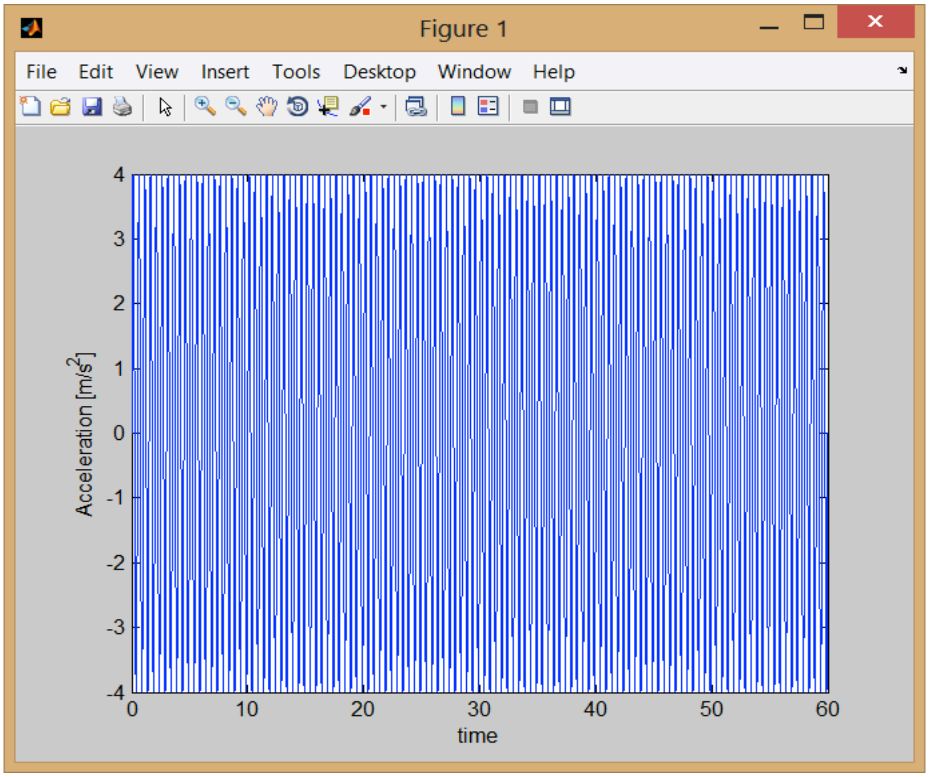


Figure 3.4: Input Sinusoidal Acceleration

The sinusoidal acceleration input and the Coulombic electrostatic damping force are tweaked so that the proof mass completes a full flight from one end-stop to the other. This requires the sinusoidal acceleration input to be just strong enough to break away from the Coulombic damping force and snap away in the direction of the opposite end-stop. Positive, real instantaneous power is generated as the proof mass is travelling between one end-stop to the other. By the time the proof mass reaches the opposite end-stop, average positive energy is generated over the time interval that the proof mass was travelling between the two end-stops. This phenomenon is portrayed in Figure 3.5, where the generated mechanical power and proof mass position traces are shown when the sinusoidal acceleration input has an amplitude of 4 $m/s^2$, frequency of 6 Hz, and the Coulombic electrostatic damping force is set constant at F=0.9 mN (milli-Newtons). This matches the expectation for the correct functional operation of the MSD in a CFPG device.

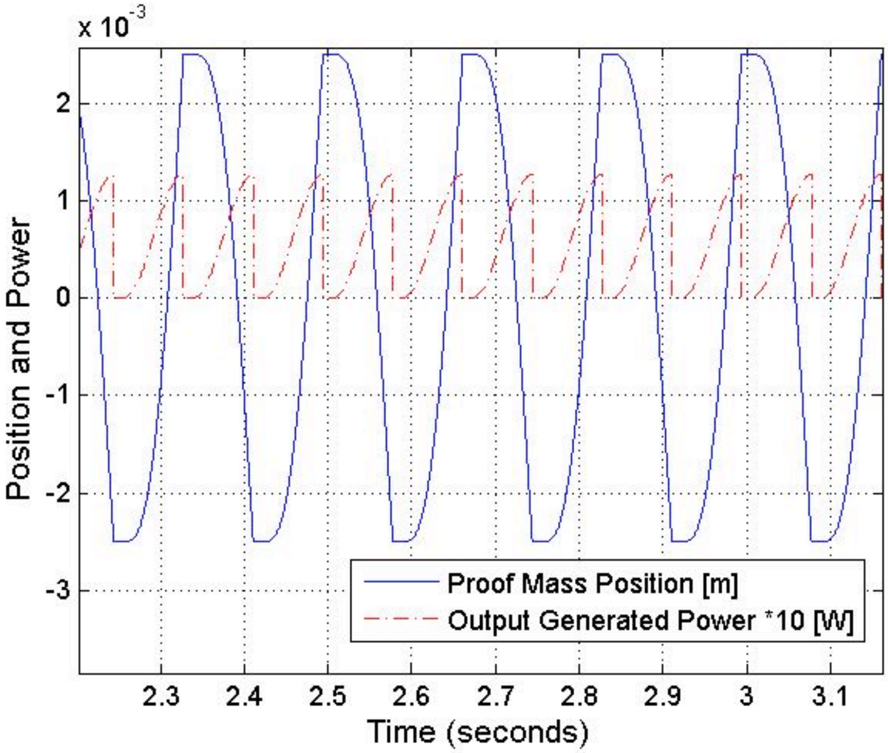


**Figure 3.5: Proof mass full flight between device end-stops and resulting generated instantaneous power**

It is possible to either tweak the sinusoidal acceleration input or the Coulombic damping (or holding) force so that the proof mass does not complete a full end-to-end flight across the MSD frame's end-stops. Compared to the previous case where the proof mass makes a complete flight between the two end-stops, the Coulombic damping force is increased to oppose the sinusoidal acceleration input and prohibit the proof mass to make a complete flight across the MSD frame. As a result of this incomplete flight, it is expected that the instantaneous output power will have equal positive and negative components (the negative components being reactive power) through the duration of this incomplete flight. Consequently, no average power is generated across the time interval where the proof mass makes the incomplete flight. This phenomenon is depicted in Figure 3.6, where the generated mechanical power and proof mass position traces are shown when the sinusoidal acceleration input has an amplitude of 4 m/s$^2$, frequency of 6 Hz, and the Coulombic electrostatic damping force is set constant at F=2.7 mN (milli-Newtons). This confirms that a zero net average power will be generated for incomplete proof mass flights.

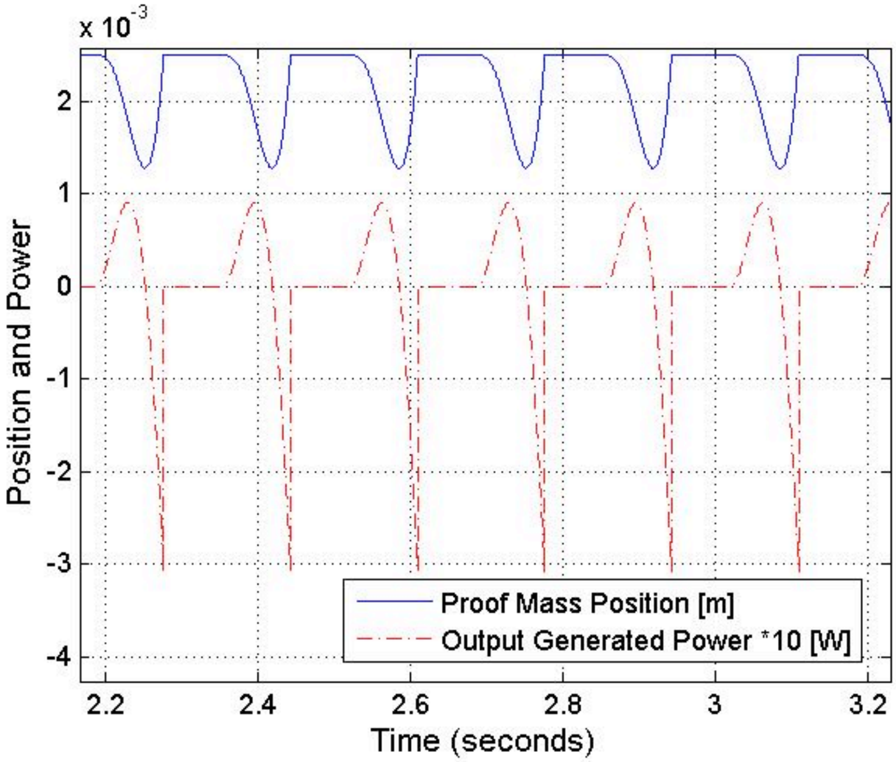


**Figure 3.6: Proof mass incomplete flight between device end-stops and resulting generated instantaneous power**

## ***3.4: Relationship Between Sinusoidal Acceleration Input and CFPG Electrostatic Damping Force***

In addition to making sure power is only generated when the proof mass makes a complete flight between the two end-stops and that no power is generated otherwise, a few more aspects of the simulations are tested to verify the validity of the CFPG model. With sinusoidal acceleration inputs, it is expected that increasing the CFPG's Coulombic damping force increases the amount of generated power while still maintaining a constant number of flights that the proof mass makes between the two end-stops [28]. However, increasing the Coulombic damping force too far causes the damping force to become too strong for the sinusoidal acceleration input to overcome. As a result, because the damping force is too strong to allow any proof mass movement, both the average generated power and the number of flights between the two end-stops drops to zero.

To verify the CFPG model's behavior, a series of simulations are performed to keep track of the relationships between the external sinusoidal acceleration, the Coulombic damping force, and the generated mechanical power. Using the same-length time interval of sinusoidal acceleration data, this input is characterized with an amplitude and frequency in order to input into the simulations model. This way, the following relationships are explored:

- Relationship between average power generated and sinusoidal acceleration inputs of varying amplitudes. The sinusoidal acceleration data's frequencies are kept constant, as well as the CFPG model's damping force.
- Relationship between average power generated and sinusoidal acceleration inputs of varying frequencies. The sinusoidal acceleration data's amplitudes are kept constant, as well as the CFPG model's damping force.
- Relationship between average power generated and the CFPG's Coulombic damping forces.
- Exploring the relationships that exist between Coulombic damping force values that attempt to maximize output generated powers with respect to a range of sinusoidal acceleration input amplitudes and frequencies.

A summary of the results obtained from exploring these relationships is discussed below.

First, an investigation is done to explore the impact of the sinusoidal acceleration input's amplitude on the output generated power when the sinusoidal input's frequency and the CFPG model's damping force are kept constant. For weak (i.e. low) sinusoidal amplitudes that are not greater than the damping force, the proof mass sticks to an end-stop and does not move. As a result, no mechanical power is generated. However, the model generates power once a sinusoidal acceleration input's amplitude is strong enough to allow the proof mass to make a full flight from one end-stop to the other.

The average power generated stays the same even as the input's amplitude increases. This is because the motion of the proof mass is bounded by the end-stops. The bounds limit the acceleration inputs from displacing the proof mass beyond the end-stops, instead adhering all the acceleration points to travel the same distance in the CFPG device. For example, increasing the amplitude of the input acceleration signal results in having the proof mass travel a fixed distance (between the end-stops) with a higher velocity but for a shorter time interval. Higher proof mass velocity means more instantaneous power has been generated (power is a product of velocity and damping force), but for a shorter time interval. Comparing power generated between a high input amplitude and a lower one, equal average power is harvested across the same-length time interval that encompasses the full flight of the proof mass between the end-stops. Furthermore, a constant sinusoidal acceleration input frequency maintains the same number of flights that the proof mass makes between the two end-stops. Power is generated every time a proof mass makes a flight between the two end-stops, and a constant number of flights corresponds to the same number of times that which power is generated within a length of time.

Figure 3.7 demonstrates this phenomenon when sinusoidal acceleration input's frequency is held constant at 3 Hz and the Coulombic damping force is constant at $F$=2mN. There exists a threshold for the input amplitude, above which the amount of the average generated power is constant, while below that threshold there is no output power. Similar behavior is observed for other combinations of constant sinusoidal input frequencies and damping force values.

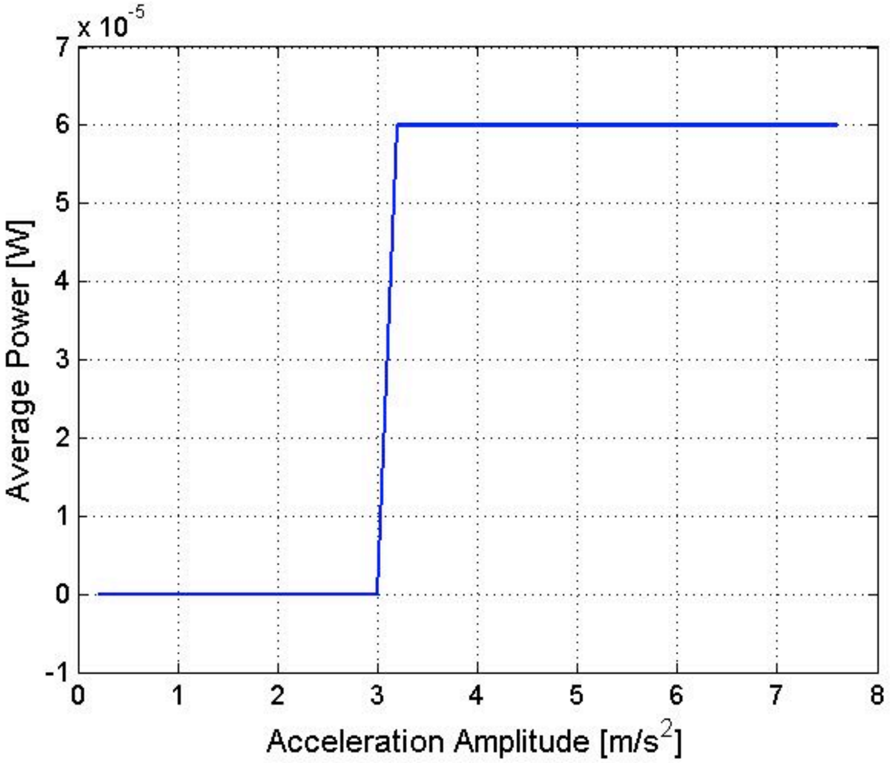


Figure 3.7: Average harvested power versus amplitude of sinusoidal acceleration input

The sinusoidal acceleration input's frequency impacts the average generated power. Keeping the sinusoidal input's amplitude and MSD module's Coulombic damping force constant, the average generated power monotonically increases with increasing the input's frequency. However, the harvested power drops to zero once the frequency reaches high threshold.  Higher frequencies translate to faster oscillation of the proof mass. This results in higher generated power. However, at some point, the frequency would be too high for the proof mass to make complete end-to-end flights and consequently the generated power drops to zero.

As the frequency of the sinusoidal input within a specified time interval increases, so do the number of flights between the CFPG frame's end-stops. With more flights between the end-stops, the CFPG device generates more power. At some point, the frequency becomes too high and the proof mass is unable to make complete flights from one end-stop to the other. The power output drops to zero, and remains zero for higher sinusoidal acceleration input frequencies as well.

Figure 3.8 demonstrates this phenomenon when the sinusoidal acceleration input's amplitude is held constant at 4 m/s$^2$ and the Coulombic damping force is constant at $F$=2mN. There exists a threshold where increasing the sinusoidal input's frequency does not further monotonically increase the amount of generated power, but instead the MSD model does not yield any more generated power. Similar behavior is observed for other combinations of constant sinusoidal input amplitudes and Coulombic damping force values.

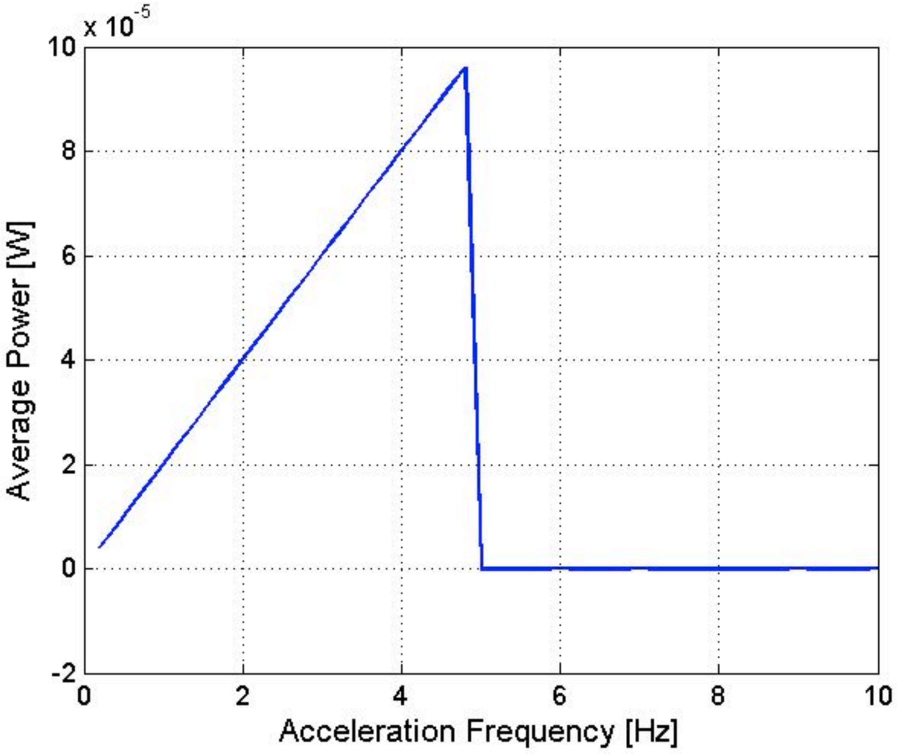


Figure 3.8: Average harvested power versus frequency of sinusoidal acceleration input

Next, the effect of the electrostatic Coulombic damping force on the average generated power is examined for when sinusoidal acceleration input parameters (amplitude and frequency) are kept constant. The average generated power sharply drops to zero after increasing the damping force beyond a certain threshold. In addition, the peak average power (which in Figure 3.9 around 80 $\mu W$) occurs at a damping force value just below this threshold.

Physically, this indicates that a strong electrostatic damping force ($F$) prevents the proof mass from moving, resulting in zero output power [28]. Further increasing the damping force values makes no difference, as it is already too strong of an opposing force to allow the proof mass any movement. Also, it should be noted that with no damping force, the proof mass is able to move between the frame's end-stops, however it does not generate power since there is no force doing work to oppose to the proof mass movements. Therefore, by judiciously choosing the value of F, one can affect the average generated power depending on the acceleration input.

Figure 3.9 demonstrates this phenomenon when the sinusoidal acceleration input maintains a frequency of 3 Hz and an amplitude of 4 m/s$^2$. This input is simulated for a range of various damping force values. There exists a threshold where increasing the MSD model's damping force value ($F$=2.7mN) that does not further monotonically increase the amount of generated power, and instead the MSD model does not yield any more generated power. Similar behavior is observed for simulating a range of damping force values across other combinations of constant input sinusoidal acceleration frequencies and amplitudes.

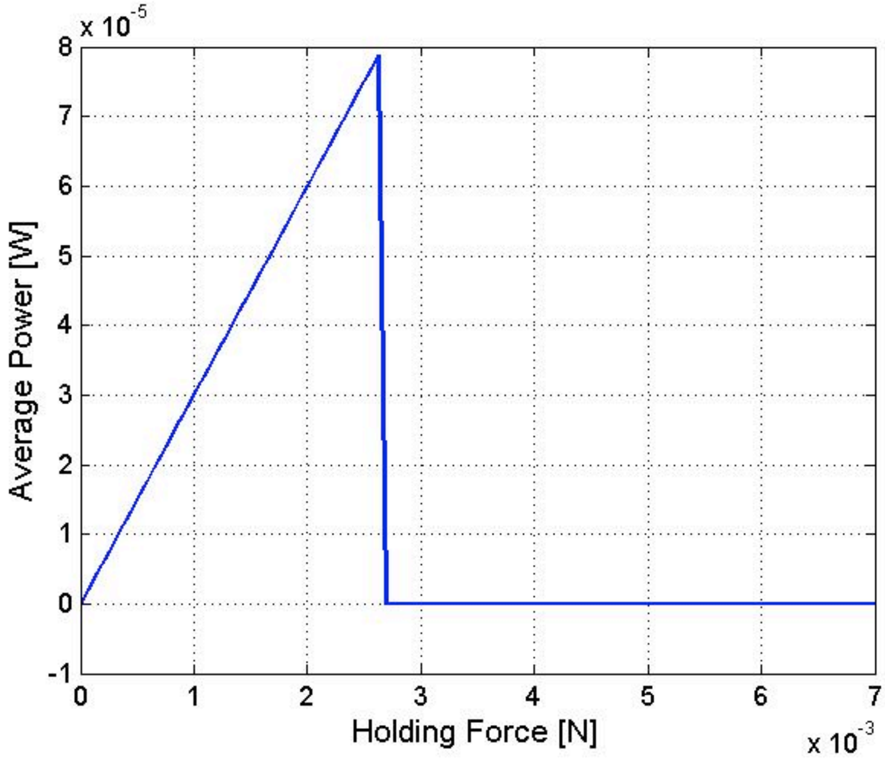


**Figure 3.9: Average harvested power versus damping (or holding) force tuned to sinusoidal acceleration input**

The relationship between the average generated power and the damping force, while the external sinusoidal acceleration input remains the same, points to the existence of an optimal value for the electrostatic force $F$ that which maximizes the generated mechanical power. As discussed previously, the Coulombic damping force must be less than the product of the mass and external acceleration for the proof mass to break away from the frame. That is, tuning the Coulombic damping force to be just less than this product means the proof mass snaps away from an end-stop and does maximum work against the damping force to make a full flight to the opposite end-stop. This yields the maximum amount of mechanical power possible with the given input. Simulations are set up to further explore the relationship between the optimal damping forces, their corresponding maximum generated power values, and sinusoidal acceleration excitations inputs.

An evaluation is done to find the maximum power output and optimal damping forces for distinct sinusoidal acceleration inputs, each characterized by a different combination of signal amplitude and frequency. A series of simulations ran every possible sinusoidal input frequency and amplitude combination for frequencies between 0.2 and 10 Hz (frequencies simulated in increments = 0.2Hz) and amplitudes between 0.2 and 6.4 m/s$^2$ (amplitudes simulated in increments = 0.2 m/s$^2$). Each series of simulations ran each acceleration input, characterized by a particular combination of the sinusoidal frequency and amplitude, for a range of values of the damping force between the values 0 and 50 mN (F simulated in increments = 62.5 $\mu N$). The optimal damping force values that give the maximum generated power output for each sinusoidal acceleration input is depicted in Figures 3.10 and 3.11.

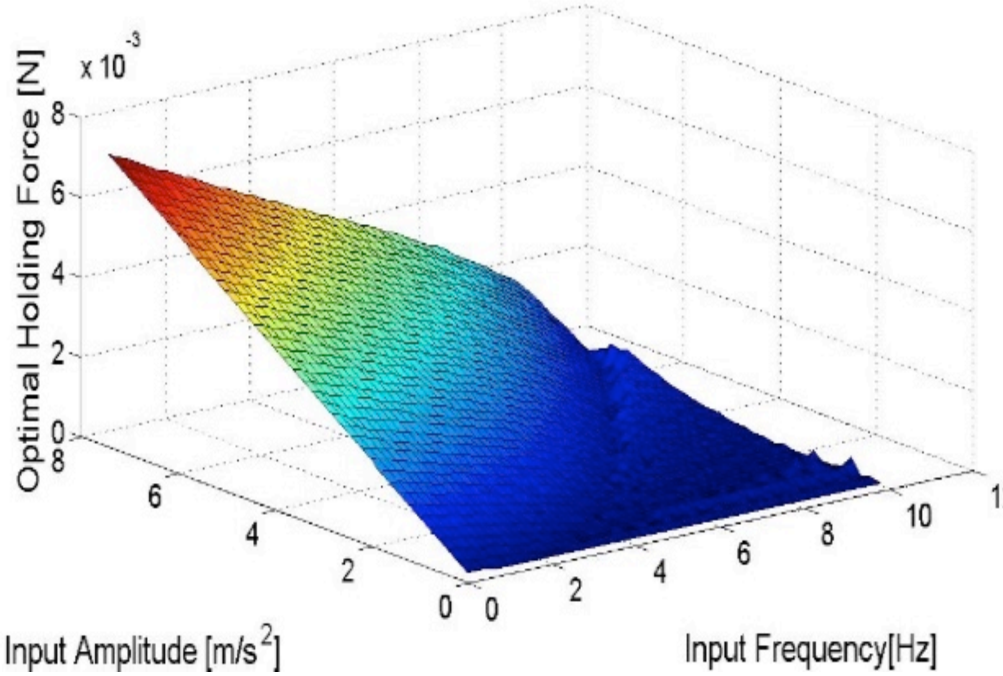


Figure 3.10: Optimal value of the electrostatic force for sinusoidal acceleration inputs with varying amplitudes and frequencies

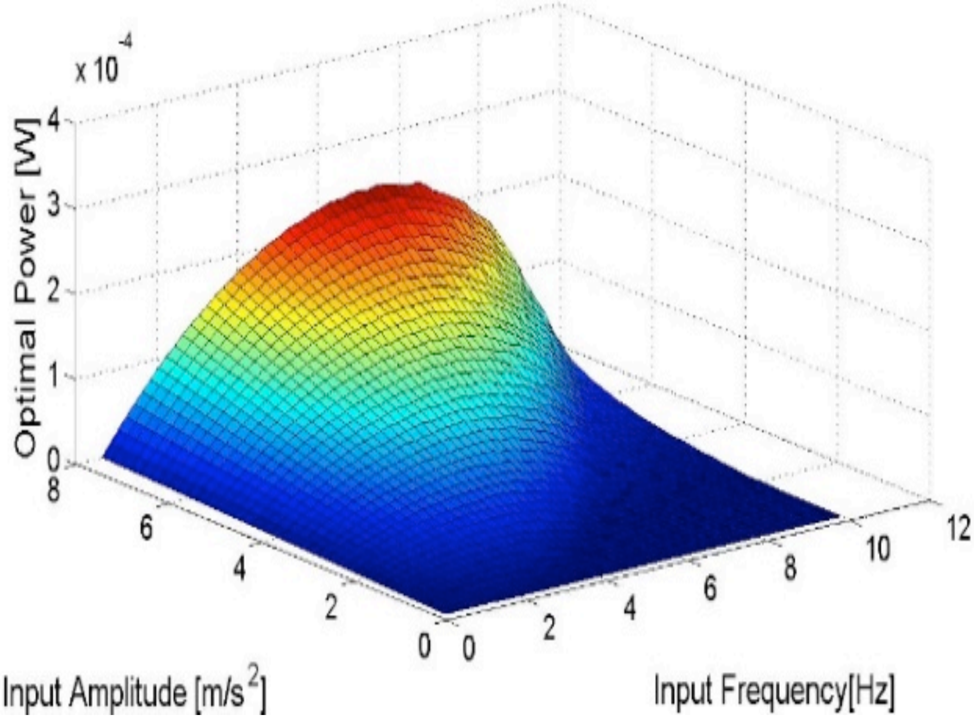


Figure 3.11: Maximized average harvested power for sinusoidal acceleration inputs with varying amplitudes and frequencies

Figures 3.10 and 3.11 indicate that stronger sinusoidal acceleration input amplitudes correspond to higher optimal damping forces and to increased resulting maximized generated power values. However, the generated power also depends on the acceleration input's frequency. If the signal's frequency is too high (i.e. over 7Hz), then the maximized power output is relatively small. Likewise, if the frequency is too low (i.e. less than 2Hz), the maximized power outputs are still low. The signals with the highest optimized generated power outputs have amplitudes that are as high as possible but with a frequency that isn't too high or too low (4-5 Hz). Examination of Figures 3.10 and 3.11 shows that up to 0.25 mW (and more, if even higher amplitude values were included in this study) of mechanical power can be harvested by the CFPG devices, which may be suitable for many low-power body sensor node applications [19].

These plots verify the behaviors of the CFPG device when experiencing external sinusoidal acceleration excitations as described in [28]. Furthermore, these plots indicate that the damping force can be a design parameter in the MSD component of a CFPG device and that, if carefully selected, the efficiency of the energy harvesting microgenerator device can be greatly improved. This brings to question if whether it is possible to tune the damping force values depending on the amount of human motion inflicted on the CFPG device.

# Chapter 4: Maximizing CFPG Harvested Power

In this chapter, a novel approach is described which aims to adaptively optimize the Coulombic damping force with time. This is the first time such adaptive optimization approach is being applied to maximizing the power output from a CFPG micro-energy harvester. Nonetheless, previous studies have shown methodologies used in order to optimize the power generated from energy harvesting microgenerators.

In [33], the authors optimize an inertial microgenerator's design parameters to maximize the power harvested from the device. Their optimization efforts require the resonant frequencies of their devices (e.g. the spring) to match the dominant frequency of the ambient environment that the devices are situated in. This is not ideal for harvesting energy from the human body since human body limb motions do not undergo vibratory motions that can be characterized by a consistent frequency. In [20], the authors pick a micro-energy harvester that utilizes Coulombic-force to harvest power and optimize it with respect to the geometrical features of the energy harvester's architecture. The main conclusion from this study was that the power converted from the energy harvester is unavoidably linked to the mass of the device. In [27], the authors aim to show optimization results for CFPG micro-energy harvesting generators when applied to human walking motion. Their optimization method involves sweeping the value of the Coulomb-force parameter across a range of values to find the Coulombic-force value that yields the most power for the device. Among the compared micro-generator architectures and the given constraints on the miniaturized size of the architectures, their results suggest that the CFPG architecture achieves the highest power density [27].

***4.1: Optimization Problem Formulation***

The purpose of this chapter is to adaptively optimize the transducing parameter (the Coulombic, electrostatic damping or holding force) of a CFPG energy harvesting microgenerator with respect to acceleration input from real human body motion. An important objective is to characterize the amount of harvested mechanical power with the Coulombic electrostatic damping force. The end goal is to show that by judiciously choosing the value of the Coulombic electrostatic damping force, *F*, with respect to time, one can maximize the average generated mechanical power.

The following optimization problem in Equation 5 has been formulated by utilizing Equations 2 and 4 as constraints:

$$Argmax_{F_i}\left[\left(\frac{1}{\Delta}\right) \times \sum_{t=t_i}^{\Delta-t_i} P(t)\right] \qquad (5)$$

such that

$$P(t) = F_i * z'(t)$$

and

$$my''(t) = -mz''(t) - F_i * relay(z(t))$$

The objective aims to maximize the average harvested power during the time interval *Δ* by choosing the optimal value of the design parameter, electrostatic force *F*. The choice of *Δ* translates to how quickly the value of the electrostatic force should be adapted to the input in order to maximize the output power. This interval of time can be treated as another parameter and this study considers interval sizes of 40, 5, 2, 1, 0.5, 0.2, and .125 seconds for adaptively tuning the damping force. This optimization formulation utilizes acceleration traces collected from real human body motions [18, 32] and runs this data through programs that implement Equation 5, with chosen optimization algorithms.

### ***4.2: Source of Input Acceleration Traces***

The human body motion acceleration data used in this thesis makes use of acceleration traces from a previous study  [32], where forearm and leg acceleration data were measured with an X6-1 USB tri-axial accelerometer [37]. The accelerometer was either placed on the calf or forearm of an individual. The acceleration traces were time-stamped and stored on-board, and the accelerometer had a sampling rate of 32 Hz with an amplitude range of +/- 12 g for the acceleration measurements. In [32], the accelerometer collects up to 8 hours of acceleration traces during the day from 40 different individuals, obtaining a total of 320 hours of data. The test subjects were between 20 and 55 years of age, with the same number of males and female, and various body types. Figure 4.1 depicts the accelerometer mounted on the forearm of one of the test subjects.

Although the accelerometer measured all 3 axes of data from the human limb motions, only the z-axis from the acceleration traces were taken into account. The reluctance to use the magnitude of all 3 axes means that the estimates for harvested energy are conservative. Future studies will aim to see the differences between using one axis of the acceleration trace inputs and using the magnitude of all 3 axes.

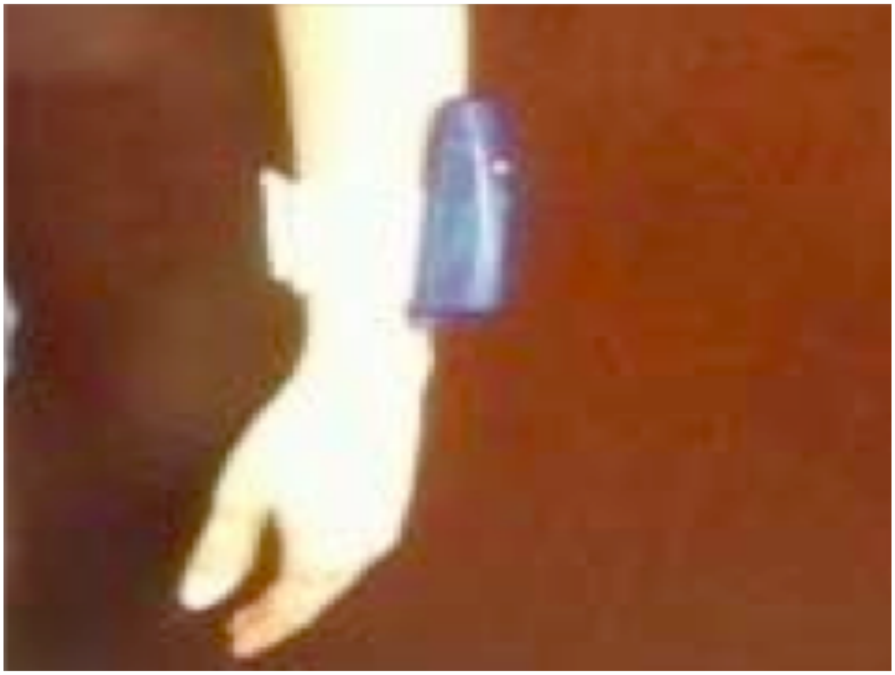


Figure 4.1: The accelerometer mounted on the forearm [32]

To implement the optimization algorithm, seven different sets of data were accessed by random from the database [32]. From each of these seven sets of data, 40-second samples were randomly selected in order to use as acceleration input traces for the optimization scheme. Seven of these 40-second samples came from data obtained by harnessing the accelerometer on an individual’s arm, and 2 of them came from harnessing the accelerometer on an individual’s leg.

A few difficulties and challenges come from the acceleration traces being used as inputs into the implementation of the optimization formulation, Equation 5. First, it is important to note that the acceleration input into the Simulink model is in the form of discrete data. The (time) steps taken by the Simulink solver are different from the ones specified in the accelerometer’s input time vector. As a result, running the accelerometer data through the Simulink model may output inaccurate results if the input accelerometer data is not aligned with steps taken by the Simulink solver [42]. The input accelerometer datapoints that are not aligned with the steps taken by the Simulink solver are skipped and not taken into account in the simulation output. To avoid this, the solution is to not provide the time vector of the accelerometer data when inputting these datapoints as discrete data into the simulation. This way, the sampled data will take on the time-steps from the fixed-step Simulink solver. However, this was followed up by another problem arising from the mechanism in which the accelerometer would record data.

Although the accelerometer operated at 32 Hz, it did not record data at 32 Hz unless it detected the presence of acceleration on the device. As a result, one-second samples from the acceleration database did not necessarily have 32 data-points. Inputting the data into the simulations without a time-stamp would lose the sense of time in which the accelerometer actually recorded data. The solution to this is to interpolate the accelerometer datapoints with a piecewise cubic Hermite interpolating polynomial [43] and upsample the data to 32 Hz. Setting the time-steps from the Simulink solver at 32 Hz, it became possible to input the interpolated acceleration data of the same frequency without the time vector, and know that the output data stream has linearly spaced points with a frequency of 32 Hz.

## ***4.3: Implementation of Optimization Formulation***

The goal is to implement the optimization formulation described in Section 4.1 (Equation 5) using computer simulations. Each of the seven 40-second acceleration traces described in Section 4.2 runs through the implementation of the optimization formulation in Equation 5. The program takes each of the 40-second acceleration traces and runs it through the optimization algorithm for each of the damping force-tuning interval sizes of

40, 5, 2, 1, 0.5, 0.2, and .125 seconds. The damping force is tuned for successive window frames of input acceleration, where the window frame lengths are specified by the interval of time.

A Pattern Search optimization algorithm [36] is implemented using Matlab's Global Optimization Toolbox [38] and the Simulink Design Optimization Toolbox [39]. With the Simulink Design Optimization Toolbox, it is possible to use a Simulink diagram as a custom objective function, in this case a non-linear differential equation implemented in Simulink (see diagram in Figure 3.3). The toolbox enables the user to specify design variables, design requirements, and constraint functions. Once the optimization problem has been formulated, the user is allowed to pick an optimization algorithm from Matlab's Optimization and Global Optimization toolboxes. Of the optimization algorithms available to implement from the Simulink Design Optimization Toolbox, the Pattern Search algorithm was chosen because it does not require a gradient to find an optimum point. This is important as the objective function (the Simulink block diagram) is a non-linear differential equation and the available gradient-based solvers are ineffective for such problems. Furthermore, a pattern search algorithm allows for running the optimization in a parallel setting and distributing the computational load across multiple Matlab workers, as it is shown that parallel computing for pattern-search algorithms may accelerate optimization up to 2.81 times [55]. The Matlab and Simulink implementation of this optimization algorithm can be found in Appendix B.

The implementation of Equation 5 starts with selecting an acceleration trace for the input, an initial value for the proof mass position and velocity, an initial value for the Coulombic damping force F, and a time interval size for how often $F$ is adaptively optimized. Once these are specified, windows of the acceleration data with a window-length defined by the chosen interval size are selected to run through the optimization algorithm, one window frame of the acceleration data at a time. For each successive window of acceleration input, the Simulink Optimization Toolbox is used to specify the Simulink block diagram of Equation 5 as the optimization's constraint function, set the design variable to be the parameter *F*, utilize a customized objective function that assigns the optimization algorithm to find the maximized average harvested power from the current acceleration window frame, and finally runs the pattern search algorithm to find the optimal damping force.

Limits on the design variable are also specified in the algorithm. From [30, 35], it is known that the maximum Coulombic damping force value that allows the proof mass to snap away from an end-stop and generate mechanical power is equal to the product of the proof mass and the peak acceleration magnitude. Observing the collected acceleration database, there exists a peak acceleration around 10 m/s$^2$ for the arm-mounted accelerometer traces and a peak acceleration is around 20 m/s$^2$ for the leg-mounted accelerometer traces. Since the proof mass is set at 0.965 g, the maximum values for the Coulombic electrostatic damping force are set at 10 mN and 20 mN for the arm and leg input acceleration traces, respectively.

Next, the optimization algorithm verifies the validity of the optimization formulation in Equation 5. With validity of the model, the Simulink Design Optimization toolbox optimizes the design with respect to the selected Pattern Search algorithm. Once a design parameter damping force *F* has been found to maximize the average power generated by the current interval window of acceleration, the optimization algorithm updates the initial proof mass position and velocity values. This update initializes the next successive acceleration window frame (with a window frame length specified by the chosen interval of time for that optimization run) that which runs through the optimization algorithm.

The pattern search algorithm finds a sequence of points that approach an optimal point without needing to calculate the gradient of the objective function or its constraints [41]. The pattern search algorithm iteratively searches a set of points, called a mesh, to find a design parameter point, a damping force value, that improves the objective value (harvested power). The algorithm finds the points in the current mesh by polling design parameter points that surround the current most optimal design parameter point with a specified mesh size. The objective values of these polled points are then computed. If a point yields an objective value greater than the current most optimal design parameter value, this new point is selected as the new most optimal design parameter value. In the next iteration of the algorithm, the mesh size expands to poll points within a wider range of the new optimal design parameter value. However, if no polled points yield any objective values greater than the current most optimal design parameter value, the current most optimal design parameter value does not change in the next iteration but the mesh size contracts and searches a range of design parameter values closer to the current most optimal point.

The criteria for stopping the pattern search could be a time limit, a maximum number of iterations, a tolerance on how small the mesh size contracts to, or a tolerance on the change in objective function in 2 consecutive successful iterations (where the mesh size is also less than this tolerance). For the purposes of maximizing power yielded by each interval window of acceleration input, the algorithm terminates its search for an optimal design value for the electrostatic damping force when the mesh size polling around an optimal damping force value is less than a specified mesh tolerance.

One of the challenges of this optimization was finding the optimization settings that accurately find a window frame's optimal electrostatic damping force. Many tweaks were necessary to find such settings, as it is very possible for the pattern search algorithm to skip or miss the optimal damping value that yields maximized power for a particular interval window of acceleration inputted into the system. Choosing maximum and minimum design parameter values have thus far been described but the other optimization settings and their definitions are laid out in Table 4.1.

**Table 4.1: Pattern Search Optimization Settings**

| Name of Optimization Setting | Description |
|---|---|
| F-maximum | Maximum value of F allowed in optimization |
| F-minimum | Minimum value of F allowed in optimization |
| Mesh Tolerance | Tolerance on the mesh size. If the mesh size is less than this value, the solver will stop |
| Initial Mesh Size | Initial mesh size for pattern algorithm |
| Mesh Contraction | Mesh contraction factor, used when iteration is unsuccessful |
| Mesh Expansion | Mesh expansion factor, expands mesh when iteration is successful |
| Maximum Iterations | Maximum number of iterations before termination for expanding/contracting the mesh size on current window of optimization |
| Cache Tolerance | When cache is set to 'on', patternsearch keeps a history of the mesh points it polls and does not poll points close to them again at subsequent iterations<br>The Cache Tolerance specifies how close the current mesh point must be to a point in the cachesize history for patternsearch to avoid polling it |
| Cache Size | Size of cache history |

The mesh size tolerance has been set so that the closest polled points to the optimal electrostatic damping force are within acceptable accuracy. The optimal electrostatic damping force values are on the scale of a few milliNewtons, and the mesh size tolerance has been set at $10^{-5}$ so that the optimal electrostatic damping force that the algorithm finds is within a few microNewtons of the actual optimal electrostatic damping force. The initial mesh size for the pattern algorithm is set to be equal to the maximum value of the electrostatic damping force allowed by the optimization algorithm. The reasoning for this is so that the mesh size covers the entire range of acceptable electrostatic damping force values. The mesh contraction factor is set at 0.99 and the mesh expansion factor is set at 1.03. With a high contraction factor and low expansion factor, the algorithm polls many design parameter values without having to needlessly recalculate similar values if being run with a high expansion factor. For the maximum iterations, an unrealistically high value of 5000 was chosen so that the algorithm doesn't terminate due to excessive terminations rather than terminate due to surpassing the mesh size tolerance. The cache tolerance and cache size specify how close the current mesh point must be to a point in the cache history in order for the pattern search optimization algorithm to avoid polling it. A cache tolerance of $10^{-5}$ helps speed up the optimization algorithm by avoiding to poll mesh points that are close to previous points that have been polled.

Previous difficulties showed that with certain settings, the electrostatic damping force would stop updating from window to window or the proof mass would be stuck to an end-stop for a prolonged amount of time. There were also instances where the optimization algorithm would not converge due to the mesh size tolerance, but instead terminate after reaching the maximum iterations allowed. The settings mentioned above helped overcome all these challenges and difficulties. Lastly, the main sanity check to see if whether the optimization settings mentioned above are valid involved using sinusoidal acceleration traces as inputs into the optimization algorithm. Comparing the maximized output power of the optimization algorithm with the maximized power for the sinusoidal acceleration input study in the previous chapter confirmed that both methods yield the same amount of maximized power. This indicates that the optimization algorithm that implements Equation 5 is valid and can be used on human body motion acceleration trace inputs.

### ***4.4: Optimization Results***

As mentioned, the goal is to run each of the seven 40-second acceleration traces described in Section 4.2 runs through the implementation of the optimization formulation in Equation 5 for all cases where the Coulombic damping force *F* is adaptively optimized at various interval sizes of 40, 5, 2, 1, 0.5, 0.2, and .125 seconds. The aim of this section is to lay out and compare the optimized harvested power from these simulations against a non-optimized scenario. In other words, this section describes the optimization gains yielded against when having held the electrostatic damping force *F* to a constant value.

It is natural to believe that the maximized harvested power (after optimization) will increase with smaller interval sizes that which the value of the electrostatic force is adaptively tuned. However, optimizing the damping force with small-enough window sizes should not yield any more gains. This is because typically the input, being human body motions, changes very little between successive small intervals of acceleration data frames. One thing to note is that, in order for the simulations to reflect prototype CFPG architectures, one final physical constraint is implemented on the simulation model in that the value of the electrostatic damping force only changes when the proof mass is held steady at an end-stop.

Two of the seven 40-second acceleration traces come from natural leg motion and the rest come from natural arm motions. The acceleration traces used as input to the Matlab/Simulink simulation model are laid out in the plots below.

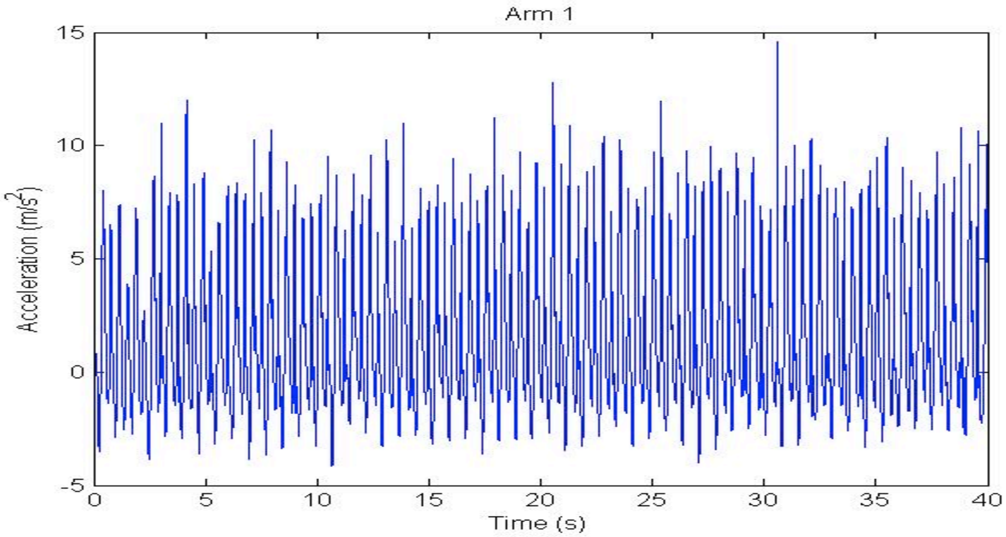
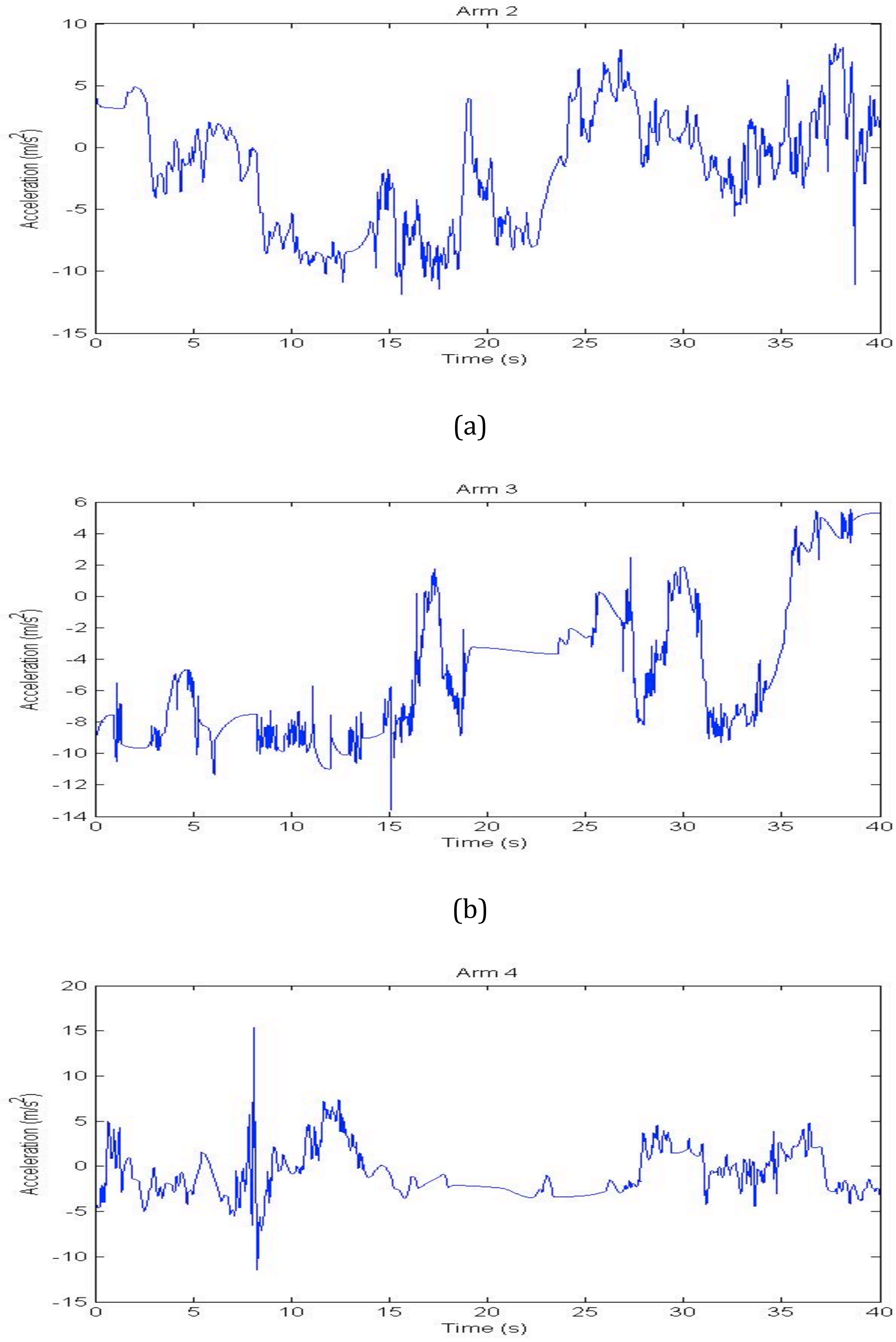


Figure 4.2: 40-second acceleration trace from source Arm 1

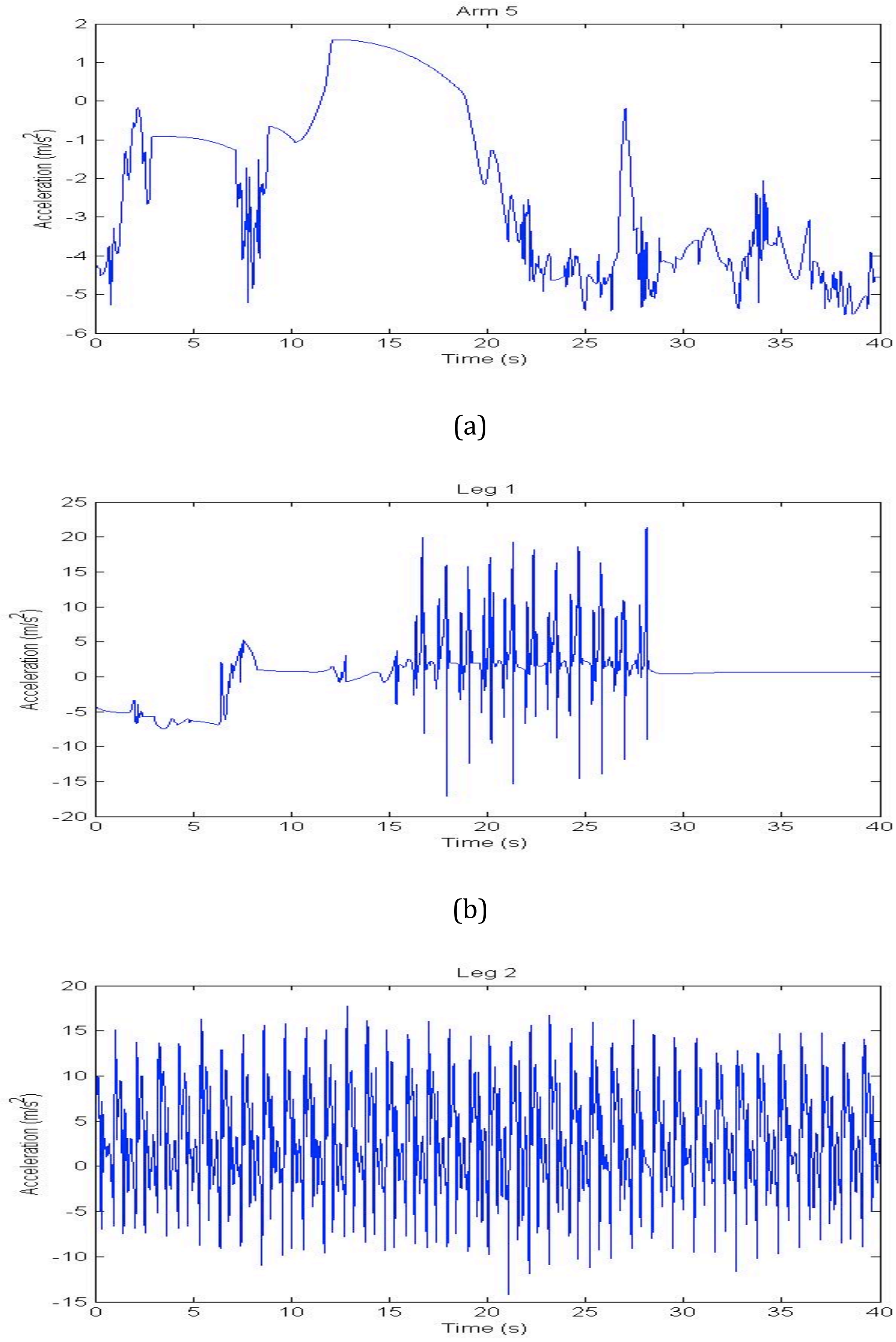


(a)

(b)

(c)

**Figure 4.3: 40-second acceleration traces from different sources, denoted by: (a) Arm 2, (b) Arm 3, (c) Arm 4**



(a)

(b)

(c)

**Figure 4.4: 40-second acceleration traces from different sources, denoted by: (a) Arm 5, (b) Leg 1, and (c) Leg 2**

These acceleration traces were used as the optimization model's inputs and their maximized power outputs were obtained. The maximized harvested powers and gains with respect to different interval sizes that which F is adaptively optimized with respect to the input acceleration traces are laid out in the Tables below. The gains were obtained by looking at the ratio of the optimized maximized power value to the power yielded when setting the CFPG device to have a constant electrostatic damping force value of F=0.81mN. A value of $F$ = 0.81 mN is chosen as the constant Coulombic electrostatic damping force since this particular value has also been used in literature [27] to simulate and optimize CFPG architectures. Matlab Code that implements this optimization can be found in Appendix B.

**Table 4.2: Energy harvested from different input acceleration datasets with respect to different damping force-tuning intervals**

| Optimization Interval $\Delta$ (Sec) | 40 s | 5 s | 1 s | ½ s | ¼ s | 1/8 s | Constant F=0.81 mN |
|---|---|---|---|---|---|---|---|
| Leg 1 Optimal Average Harvested Power ($\mu$W) | 3.1143 | 4.8636 | 5.1711 | 5.6167 | **6.0348** | 5.1110 | 2.1284 |
| Leg 2 Optimal Average Harvested Power ($\mu$W) | 33.623 | 31.277 | **41.302** | 39.938 | 35.420 | 33.013 | 14.148 |
| Hand 1 Optimal Average Harvested Power ($\mu$W) | 18.4073 | 18.555 | 18.849 | 22.790 | **26.148** | 23.427 | 11.865 |
| Hand 2 Optimal Average Harvested Power ($\mu$W) | 3.4331 | 5.3671 | **5.6622** | 5.1126 | 3.8192 | 2.9474 | 2.1417 |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| Hand 3 Optimal Average Harvested Power ($\mu$W) | 0.65180 | **1.9410** | 1.4581 | 0.84301 | 0.62011 | 0.32440 | 0.30238 |
| Hand 4 Optimal Average Harvested Power ($\mu$W) | 4.5444 | **5.0670** | 4.5905 | 4.4505 | 3.7032 | 2.3178 | 1.4222 |
| Hand 5 Optimal Average Harvested Power ($\mu$W) | 0.3873 | 0.4113 | **0.7115** | 0.1229 | 0.1576 | 0.1029 | 0.1090 |

Table 4.3: Power gains, ratio of energy harvested from different input acceleration datasets at different damping force-tuning intervals to energy harvested with a constant damping force F=0.81mN

| Optimization Interval $\Delta$ (Sec) | 40 s | 5 s | 1 s | ½ s | ¼ s | 1/8 s |
|---|---|---|---|---|---|---|
| Leg 1 Ratio of Optimization/Non-Optimization in Power Gain | 1.46 | 2.29 | 2.43 | 2.64 | **2.84** | 2.40 |
| Leg 2 Ratio of Optimization/Non-Optimization in Power Gain | 2.38 | 2.21 | **2.92** | 2.82 | 2.50 | 2.33 |
| Hand 1 Ratio of Optimization/Non-Optimization in Power Gain | 1.55 | 1.56 | 1.59 | 1.92 | **2.20** | 1.97 |
| Hand 2 Ratio of Optimization/Non-Optimization in Power Gain | 1.60 | 2.51 | **2.64** | 2.39 | 1.78 | 1.38 |
| Hand 3 Ratio of Optimization/Non-Optimization in Power Gain | 2.16 | **6.41** | 4.82 | 2.78 | 2.05 | 1.07 |
| Hand 4 Ratio of Optimization/Non-Optimization in Power Gain | 3.19 | **3.56** | 3.23 | 3.13 | 2.60 | 1.63 |
| Hand 5 Ratio of Optimization/Non-Optimization in Power Gain | 3.55 | 3.77 | **6.51** | 1.13 | 1.44 | No gain |

The results show that gains of between 2 to 6 times can be achieved when using the optimization algorithm to find maximized power harvested from CFPG by adaptively tuning the value of the electrostatic damping force instead of maintaining a constant value for the damping force. There are two instances where tuning the damping force at 5-second intervals of time yields the maximized power, 3 instances with 1-second intervals of time, and 2 instances with quarter-second intervals of time. It is obvious that when the interval is too big, it encompasses too much activity to harvest as much power as possible from the ambient environment. On the other hand, when the interval size is too small, the optimization algorithm does not yield more power because the input, being human body motions, changes very little between successive small intervals of acceleration data frames.

One possible hypothesis is that the amount or type of activity in the acceleration traces affects the interval sizes of tuning the Coulombic electrostatic damping force that yields the maximized average power harvested from the trace. One possible property of the acceleration traces to look at is the dominant frequency of motion and corresponding amplitude of the dominant frequency. This is determined by looking at the maximum spectral component of the Fourier transform of the acceleration trace [33]. This property points to the acceleration amplitude that most effectively generates power from the energy harvester [32]. Specifically, after taking the Fourier transform of the input acceleration data, the product of the acceleration amplitude and the corresponding frequencies is taken to look for the dominant frequency for which the product of the acceleration amplitude and corresponding frequency is maximized. The table below shows the dominant frequency and amplitude of that dominant frequency for each acceleration trace.

Table 4.4: Acceleration input characteristics

| Acceleration Trace | Dominant Frequency of Motion (Hz) | Acceleration amplitude of Dominant Frequency ($m/s^2$) |
|---|---|---|
| Leg 1 | 15.1 | 0.1752 |
| Leg 2 | 3.82 | 1.71 |
| Arm 1 | 2.67 | 3.14 |
| Arm 2 | 2.01 | 0.33 |
| Arm 3 | 10.95 | 0.06 |
| Arm 4 | 3.33 | 0.24 |
| Arm 5 | 3.61 | 0.07 |

Generally, a higher acceleration amplitude of the dominant frequency corresponds to more power yielded from the optimized energy harvester, with also smaller intervals of time where the damping force parameter is adaptively tuned. However, these measures are not conclusive. An expansive study needs to be done to explore the relationship between the acceleration input and the interval sizes of tuning the damping force that which yields the maximized harvested average power output. This expansive study should also utilize other statistical properties and parameters, such as average absolute deviation of the acceleration, to characterize the amount or type of activity in the acceleration input that affects the interval sizes.

These results point to the potential impact of using the optimal value of the electrostatic force $F$ in the harvested power. As observed, a significant gain (at least up to 2 times higher) is achieved for the sample data compared to the non-optimized scenario.

### ***4.5: Maximizing Harvested Power by Tuning Damping Force Parameter with Delay***

It is not realistic to implement real-time optimization as in the previous section as there is only a fraction of a second available to tune the electrostatic damping force parameter to obtain maximum harvested average power output. The previous section showed that it is possible to optimize the architecture of the micro-generator in order to maximize harvested power. To realistically implement this in hardware, it is instead possible to tune the value of the electrostatic damping force of a future input acceleration window frame, specified by an interval of time. It is natural to believe that the human body motion does not substantially change over short intervals of time, and so optimal damping force of one frame of acceleration will be similar to the optimal damping force of the subsequent interval's frame of acceleration. Therefore, this section explores a hypothetical model that tunes the damping force at a future, subsequent window frame of input acceleration with the current window's optimal damping force. The gains in the harvested power are also compared to holding the damping force at a constant value.

Tuning the damping force at a future, subsequent window frame of input acceleration with the current window's optimal damping force possibly entails characterizing the current input acceleration window frame. A statistical characterization of the current acceleration window frame may provide insight to the frame's optimal electrostatic damping force value. Instead of performing optimization in real-time, a database of previously-computed optimal damping force values corresponding to different acceleration input statistical characterizations may be used. The database would allow doing a simple search to find an optimal electrostatic force value for the next window frame based on the characterization of the current acceleration window frame. In other words, a model is built to adaptively tune the CFPG's electrostatic damping force value for the next subsequent window frame, specified by the same interval of time of the current window frame, given a characterization of the current acceleration frame. As such, the optimal electrostatic force is implemented with a delay of a window frame (specified by an interval of time). This model, named the delayed-damping force model, is depicted in Figure 4.5.

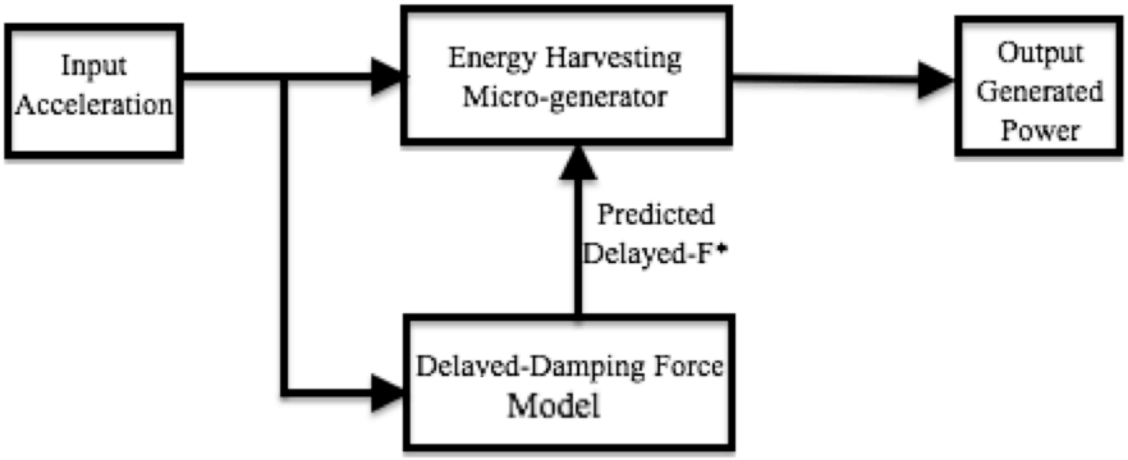

Input Acceleration
Energy Harvesting Micro-generator
Output Generated Power
Predicted Delayed-F*
Delaved-Damping Force Model

Figure 4.5: Delayed-damping force model block diagram

The effectiveness of the model's power generation determines the practicality of using such a model. The power harvested from this model is compared to the energy harvested when using the optimization formulation in the previous section and also compared to the case when the CFPG device uses a constant value for the electrostatic damping (for which a value of $F$=0.81 mN has been chosen). From the previous section's seven 40-second acceleration traces that have been used to adaptively optimize the value of the damping force over various time interval sizes, it is possible to see if whether the delayed-damping force model can be effective. The CFPG model is again simulated with the same seven acceleration traces but with optimal damping force values that are being implemented on the acceleration window frames (of a specified interval length) that succeed the window frames for which the optimal damping force had been obtained. The Matlab code that implements this test for the delayed-*F* model can be found in Appendix B.

The power yielded from the optimization implementation in the previous section and the implementation with the delayed-*F* model are compared in Table 4.5, as well as the power yielded when the CFPG model is being simulated with a constant electrostatic force of *F*=0.81 mN. The gains of these two implementations against the CFPG model simulations with the constant electrostatic force are also laid out.

**Table 4.5: Energy harvested from delayed-dampoing force model, and comparison to real-time adaptive optimization and constant-damping force implementation**

| **Dataset and optimal parameter-tuning interval size** | Maximized power yields for Optimization Implementation of Equation 5 ($\mu$W) | Maximized power yields for delayed-damping force predictor model ($\mu$W) | Power yields with constant electrostatic damping force value *F*=0.81mN (no adaptive tuning) ($\mu$W) |
|---|---|---|---|
| **Leg 1**, optimized by adaptively tuning F at intervals of **1/4s** | 6.0348 | 5.8606 | 2.1284 |
| **Leg 2,** optimized by adaptively tuning F at intervals of **1 s** | 41.302 | 27.596 | 14.148 |
| **Hand 1**, optimized by adaptively tuning F at intervals of **¼ s** | 26.148 | 16.555 | 11.865 |

| | | | |
|---|---|---|---|
| **Hand 2**, optimized by adaptively tuning F at intervals of **1 s** | 5.6622 | 4.8073 | 2.1417 |
| **Hand 3**, optimized by adaptively tuning F at intervals of **5 s** | 1.9410 | 0.08585 | 0.3024 |
| **Hand 4,** optimized by adaptively tuning F at intervals of **5 s** | 5.0670 | 2.4502 | 1.4222 |
| **Hand 5**, optimized by adaptively tuning F at intervals of **1 s** | 0.7115 | 0.35960 | 0.1090 |

Table 4.6: Ratio of Optimization/Non-optimization in power gain yields for delayed-damping force model against constant-damping force implementation

| **Dataset and optimal parameter-tuning interval size** | Ratio of Optimization/Non-Optimization in Power Gain for Optimization Implementation of Equation 5 | Ratio of Optimization/Non-Optimization in Power Gain yields for delayed-damping force model |
|---|---|---|
| **Leg 1**, optimized by adaptively tuning F at intervals of **1/4s** | **2.84** | 2.75 |
| **Leg 2,** optimized by adaptively tuning F at intervals of **1 s** | **2.92** | 1.95 |
| **Hand 1**, optimized by adaptively tuning F at intervals of **¼ s** | **2.20** | 1.40 |
| **Hand 2**, optimized by adaptively tuning F at intervals of **1 s** | **2.64** | 2.24 |
| **Hand 3**, optimized by adaptively tuning F at intervals of **5 s** | **6.41** | No gain |
| **Hand 4,** optimized by adaptively tuning F at intervals of **5 s** | **3.56** | 1.7228 |
| **Hand 5**, optimized by adaptively tuning F at intervals of **1 s** | **6.51** | 3.30 |

In all but one case, the model where the optimal damping force value was applied with one frame of delay showed significant gains over the case where the CFPG's damping value remains constant at 0.81mN. Also, as expected, the model with the delayed-damping force values showed less harvested power gain than the real-time optimization from Section 4.4. However, there may be a possible trade-off between computational time and power gain. Optimizing the damping force in real time produces the most power gain but requires a higher computation time to actually be implemented in real-time. The case where the damping force is held at a constant value (F=0.81mN in the experiments above) yields the least harvested power, but requires little computation time in regards to tuning the damping force value. The delayed-damping force model requires more exploration to see how it would be implemented on the CFPG, but from the initial results it is intuitive to believe that this model requires less computational time than real-time optimization, yet gives higher power gains over the case where the damping force is held at a constant value.

More research into the delayed-damping will give a better idea on the feasibility of implementing such a model for CFPG micro-energy harvesting generators. The basic issues for implementing such a model involve the representativeness of the data, statistical characterization of the data, validity of the model, rate of adaptation, validity of the prediction, and implementation aspects such as numerical robustness, computational stability, and real-timeliness [47].

It should be noted that in the literature, there have been several energy harvesting adaptive algorithms that assumed that the energy harvesting process is Markov or has independent identically distributed properties [44-46]. However, these assumptions do not necessarily hold for all motion energy traces and Markov processes may result in very different performance trends depending on the input [33].

## Chapter 5: Conclusion

This paper first presents the potential for body sensor networks in regards to making personalized healthcare a commercial reality. Through exploring use cases of body sensor networks, a key limitation for a large-scale implementation of body sensor networks is that they heavily rely on batteries that need frequent recharge. This reliance is a deterrent for using these sensors. Therefore, the aim of this paper is to find a method or technology that prolongs the operational lifetime of body sensors, and as a result makes them more appealing to use for personalized healthcare purposes.

An energy-harvesting technology suitable for body-mounted sensors is the Coulomb Force Parametric Generator (CFPG) kinetic microgenerator architecture. This architecture consists of a Mass-Spring-Damper (MSD) system, a transducer and power-processing circuitry. This thesis focuses on the MSD and transducer models. A modified model of the non-linear differential equation is presented for the MSD component. This model can be used to measure the amount of mechanical power generated. The conversion to electrical power is done through an electrostatic transducer module. Optimizing the architectural and design parameters of the harvester device based on the characteristics of the input acceleration increases the amount of the generated mechanical power. Different adaptive optimization models were also presented, and coming to any conclusive remarks about the power generated necessitates going back and comparing the power that could potentially be harvested with the power requirements of the body sensors.

The CFPG model simulation and optimization results show that the energy harvested by this device can reach up to 10's of micro-Watts in mechanical power. As for power consumption requirements of body-mounted sensors, there is a possibility to require no more than 1 $\mu$W for a body-mounted sensor node, including power required by the signal processing circuitry, wireless communication, and the sensor itself [19]. However, certain low-power applications may require as much as 100 $\mu$W [48]. Regardless, the energy harvested from the CFPG model simulations is sufficient and significant enough to make it a worthwhile endeavor for prolonging the operational lifetime of a body-mounted sensor node. Yet there are still limitations and challenges that need to be overcome.

To facilitate the adoption of energy-harvesting microgenerators, both progress is being made with newer technologies of body-powered applications having lowered their power requirements [19], as well as newer micro-generator trends showing significant improvement with the amount of energy they are capable of harvesting [49]. However, an absolute limitation on the power requirements is the size of the energy harvesters themselves [23]. For device sizes of interest in this paper, with a maximum volume of 1 cm$^3$, it is unlikely for a harvester to generate more than 1 mW from body motions [48].

Other limitations include adding an energy storage mechanism to the energy harvesting device since body motions that power energy harvesting vary substantially with time, and that these variations are unlikely to correspond with the varying demands of the body-mounted sensor node [19]. To elaborate, over 95% of total energy from the human body is collected only from 4-7% of a day [33]. In addition, models for predicting future consumption and power availability may aid in managing the variability of power harvested from the human body [50].

Effectiveness of coupling the energy harvesting transducers with power-processing interface circuit is another challenge of the energy harvesters. The transducers generate mechanical power, but power-processing circuitry is required to convert that into electric power that can be used by the sensor node. However, substantial power may be lost in this interface for power conversion, affecting the amount of power that can be delivered to the sensor node from the energy harvesters described in this paper [31]. In addition to the practical challenges and limitations with the energy harvester, more accurate models of the energy harvester will also aid in the optimization and simulation of these devices [32, 34].

It should be noted that this project is a Cyber-Physical System and highlights how joint design of the cyber and physical components can improve system efficiency. By adaptively tuning the electrostatic force $F$ for various human body motions, one can expect an improved efficiency in harvesting kinetic energy for wearable sensors. Physically, this can be accomplished by tuning the electric field between the CFPG's capacitive electrodes [28]. Mathematical formulation of the adaptive optimization problem can be solved by adding appropriate computational algorithms in the micro-generator architecture [18].

The plan for future works includes an exploration of the limitations and challenges discussed above, as well as improving the optimization algorithms and adaptive tuning models in Chapter 4 for higher accuracy and lower computational complexity. As mentioned, the ultimate goal is to develop an accurate model of the CFPG micro energy-harvester device and optimize the amount of generated power for various human body motions. This in turn prolongs the operational lifetime of body sensors. On a higher systems-level view, this research in turns aims to aid body sensor network systems in becoming a feasible asset for making personalized healthcare delivery a realization [18]. As a result, it is believed that integration of micro energy harvesting technology with wearable sensors is a promising approach in prolonging the operational lifetime of wearable medical sensors.

# **Appendix A: Use Case Descriptions of Medical Body Sensor Networks**

Use Case 1: System Setup

*Goals 1*

1. Provides the user with data protection, maintenance, and monitoring health.
2. Allows patient to adjust the setup themselves [51].

*Primary Actors*

Healthcare Provider, Patient, BSN Provider,

*Secondary Actors*

Maintenance, Legal System

*Pre-Conditions*

Healthcare Provider diagnoses or indicates that Patient needs a body sensor network to monitor their health.

*Primary Flow*

1. Patient is educated about their privacy rights from (Legal System). Patient gives informed consent.
    a. If informed consent is not given, then the patient cannot receive the BSN.
2. Healthcare Provider determines exactly what signals and sensors he wants to look at.
3. Senor nodes and hub received from BSN Provider
4. Qualified Healthcare Provider or technicians from Maintenance crew set-up the sensor nodes and hub on the Patient's body. BSN Provider configures communications to Server and access to records for Patient and Healthcare Provider.
5. Patient is educated about how the nodes/hub work (positioning, hub interface, etc).

*Alternate Flow*

Not Available

*Activity Diagram*

An activity diagram was chosen to depict the flow of activities required to perform this use case.

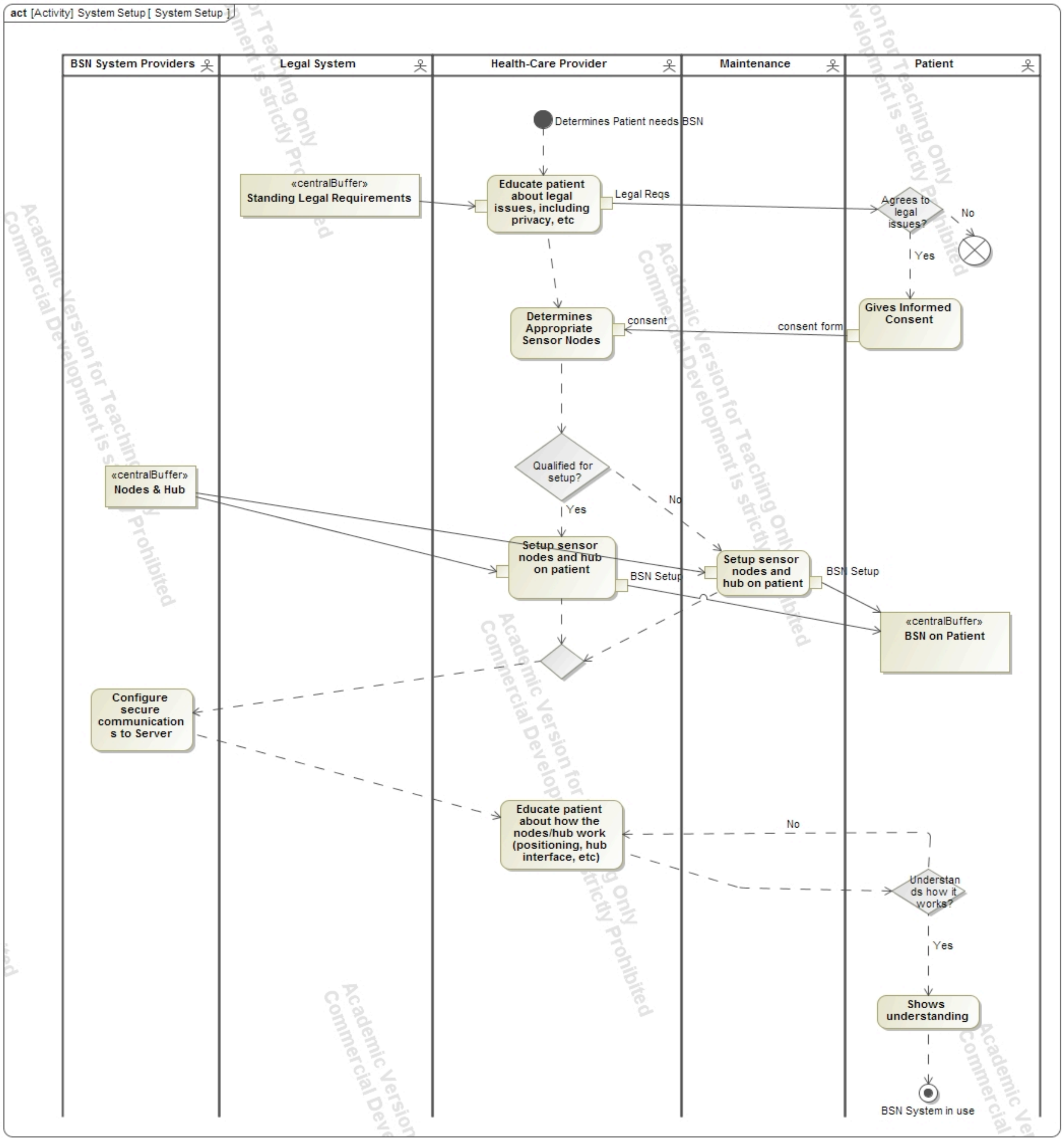

act [Activity] System Setup [ System Setup ]
BSN System Providers
Legal System
Health-Care Provider
Maintenance
Patient
Determines Patient needs BSN
«centralBuffer»
Standing Legal Requirements
Educate patient about legal issues, including privacy, etc
Legal Reqs
Agrees to legal issues?
No
Yes
Gives Informed Consent
consent form
consent
Determines Appropriate Sensor Nodes
Qualified for setup?
No
Yes
«centralBuffer»
Nodes & Hub
Setup sensor nodes and hub on patient
BSN Setup
Setup sensor nodes and hub on patient
BSN Setup
«centralBuffer»
BSN on Patient
Configure secure communications to Server
Educate patient about how the nodes/hub work (positioning, hub interface, etc)
No
Understands how it works?
Yes
Shows understanding
BSN System in use

Figure A.1: Use Case 1 Activity Diagram

*Post-Conditions*

Patient has BSN setup properly. They know how to use it and adjust it.
Use Cases 2-6.

*Derived Requirements*

1. Legal framework allows legal privacy protection, informed consent, and protection of minorities.
2. Sensor Nodes and Hub do not interfere with natural Patient postures.

3. Patients are capable of adjusting Nodes if necessary without disrupting the Nodes sensing capabilities or communication.
4. Sensor Nodes can be added or removed with ease.

<u>Use Case 2:</u> Maintain Components

*Goals 2*

1. Maintain Hub components
   1.1. Alert patient, healthcare provider, and maintenance crew in case of defective Hub (autonomic sensing) [19].
2. Maintain Server components
   2.1. Alert patient, healthcare provider, and maintenance crew in case of defective Server (maintenance and autonomic sensing) [19].
3. Maintain BSN components
   3.1. Alert patient, healthcare provider, and maintenance crew in case of defective BSN [19].
4. Make sure all system components are reliable.

*Primary Actors*

Maintenance

*Secondary Actors*

Patient, Environmental factors, Healthcare Provider

*Pre-Conditions*

Use Case 1

*Primary Flow*

1. Autonomous sensing techniques running through the BSN detect a problem at a communication link or component node. Faults include a lack of component function or lack of connectivity.
2. Problem is diagnosed using belief network techniques that run through the system. A lack of component function can be caused by a destroyed component, inconsistent sensing, a depleted power source, or a lack of communication.
   a. Note: to look at Sensor Node failures, the multi-sensor complementary data fusion needs to be taken into consideration.
3. Self-organization and sensor routing algorithms are used to engage in managing its performance in the presence of sensor failures.
4. Notification is given to relevant maintenance staff (e.g. technology specialists contacted for connectivity problems) for the diagnosed problem.
5. Relevant maintenance specialists fix the problem.
6. Self-organization and routing accommodates new fixes.

*Alternate Flow*

No faults are found; continue with Use Case 4

*State Machine Diagram*

A state-machine diagram was chosen for this use case to indicate the different states that the body sensor network system goes through when maintaining its nodes and links and detecting faults.

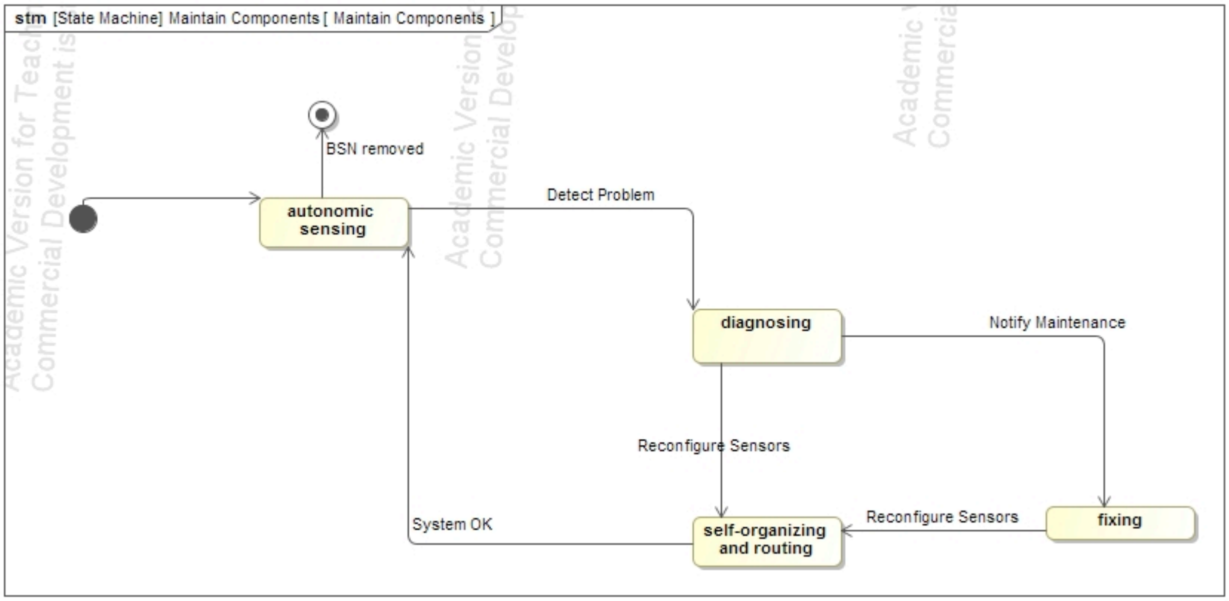


Figure A.2: Use Case 2 State Machine Diagram

*Post-Conditions*

System components and communication is reliable/normal.

*Derived Requirements*

1. Fault-detection includes all nodes (Hub, Sensor Nodes, and Server) and all the communication links between the nodes [19, 52].
2. Fault diagnosis is sophisticated and accurate [19].
3. Multi-sensor data fusion process brings together mutually correlated information (such as from ECG and haemodynamic signals for cardiac monitoring) [19] in order to enhance reliability in the event of sensor failure.
4. Sensors nodes and hub are resistant to different environmental conditions.
5. Self-organization and routing algorithms find the most efficient way to deal with energy consumption and communication paths [19].
6. All BSN components are given sufficient power.
7. Maintenance staff are specialized in solving different faults and failures.
8. User interface of patient's hub device or healthcare provider's device (or any device with access to internet) is practical for receiving information regarding fault detection

Use Case 3 Provide User-Data Protection

*Goals 3*

1. Ensure secure Server
   1.1. Alert maintenance in case of Server security breach (some algorithm) [19] which is basically anyone (hackers, other patients) getting access to other patients.
2. Ensure secure Hub
   2.1. Alert maintenance in case of Hub security breach (some algorithm) [19].
3. Ensure secure BSN
   3.1. Alert maintenance in case of BSN security breach (some algorithm) [19].
4. Keep intact legal protection and patient privacy [6].

*Primary Actors*

Intruder, Maintenance

*Secondary Actors*

Patient, Healthcare Provider

*Pre-Conditions*

Use Case 1

*Primary Flow*

1. Scenario 1: An intruder is sending malicious packets to the server,
   a. The server blocks it with very secure cryptographic network protocol
   b. Server signals to Maintenance crew about malicious attack. Also, signals to patient and healthcare provider.
   c. Maintenance specialists intervene to fix any issues.
   d. Server notifies patient and healthcare provider about fix.
2. Scenario 2: An intruder is sending malicious packets to the sensor nodes/hub by being within a few meters of sensor nodes
   a. The node/hub blocks packets with secured network protocol.
   b. Hub signals to maintenance, patient, and healthcare provider through the server about malicious attack.
   c. Maintenance specialists intervene to fix any issues.
   d. Server notifies patient and healthcare provider(s) about fix.
3. Scenario 3: An intruder is sending malicious packets to the sensor nodes/hub, and gets past the network protocols.
   a. If data is extracted, intruder is unable to identify hard-to-understand sensor data.
   b. Node with malicious packets signals other nodes that it has been hacked into.
   c. Other sensor nodes and hub isolate the node with malicious packets,

d. Hub signals to maintenance, patient, and healthcare provider through the server about malicious attack.
e. Maintenance specialists intervene to fix any issues.
f. Server notifies patient and healthcare provider about fix.

*Alternate Flow*
User data is protected, so continue with Use Case 4.

*Activity Diagram*
An activity diagram was used to go through the logical flow of actions in this use case.

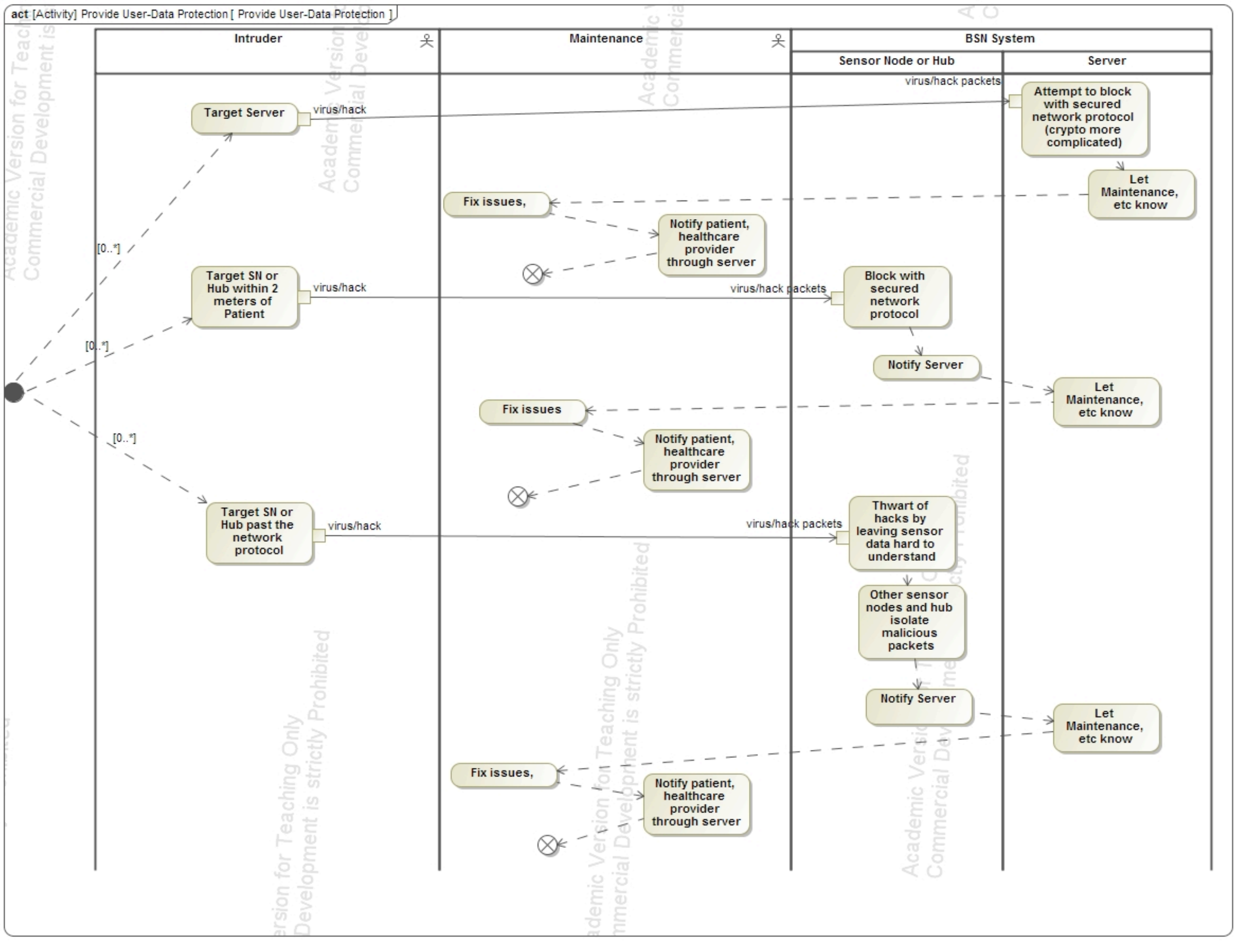

act [Activity] Provide User-Data Protection [ Provide User-Data Protection ]
Intruder
Maintenance
BSN System
Sensor Node or Hub
Server
virus/hack packets
Attempt to block with secured network protocol (crypto more complicated)
Target Server
virus/hack
Let Maintenance, etc know
Fix issues,
Notify patient, healthcare provider through server
[0..*]
Target SN or Hub within 2 meters of Patient
virus/hack
virus/hack packets
Block with secured network protocol
[0..*]
Notify Server
Let Maintenance, etc know
Fix issues
Notify patient, healthcare provider through server
[0..*]
Target SN or Hub past the network protocol
virus/hack
virus/hack packets
Thwart of hacks by leaving sensor data hard to understand
Other sensor nodes and hub isolate malicious packets
Notify Server
Let Maintenance, etc know
Fix issues,
Notify patient, healthcare provider through server

Figure A.3: Use Case 3 Activity Diagram

*Post-Conditions*
Security threats have been avoided and any security/privacy breaches have been thwarted.

*Derived Requirements*

*1.* Since the server has higher computational resources than the sensor nodes or hub, it should be capable of handling stronger cryptographic network protocol keys.

*1.1.* Interface between the hub and the server isn't difficult to use.

2. Signals sent to maintenance about malicious attacks should have different levels of malicious attack severity.
3. Network protocol keys for sensor nodes and hub should be computationally resourceful.
4. User interface of patient's hub device or healthcare provider's device (or any device with access to internet) or maintenance devices is practical for receiving information regarding malicious activity and fixes.
5. Use of anonymity and transmitting raw sensor data will thwart off intruders because it will take significant amount of effort and time to understand the context of the raw data [19].
6. Signal from infected node should let the others know that it has been intruded.

Use Case 4: Monitor Health

*Goals 4*

1. Provide patient with feedback.
   1.1. Provide patient and healthcare provider with health status [40].
   1.2. Provide standard of care for the general health condition or disease of the patient.
2. Analyze patient health at all times.
   2.1. Provide reliable analysis [53].
   2.2. Have enough power for analysis [9].

*Primary Actors*

Healthcare Provider, Patient

*Secondary Actors*

Not Available

*Pre-Conditions*

Use Case 1

*Primary Flow*

1. Raw physiological data, data important for contextual awareness [19], and any data required for multi-sensor data fusion [19], is captured by body sensor nodes.
2. Sensor nodes send their collected data to the hub.
3. Hub sends the collected data to the medical records server.
4. The medical records server collects all the data, and uses detection and decision-making algorithm to process it.
5. The server stores the data for each patient.
6. Based off of results of processed data, the server sends notification to patient hub and healthcare provider device.
7. Healthcare provider and patient view the patient's health status, and may consider communicating with each other (Use Case 6).

*Alternate Flow*

- If, at step 7 of the primary flow, the patient or healthcare provider considers communicating each other for feedback, information, check-up requests, etc, then perform the primary flow actions from Use Case 6.
- For specifics of storing accessible data (step 5 of this primary flow), look at flow of actions in Use Case 5.
- If security breached, perform actions in Use Case 3.
- If component fault detected, perform actions in Use Case 2.

*Activity Diagram*

An activity diagram was chosen as it helps represent the flow of activities that the actors perform on the data objects.

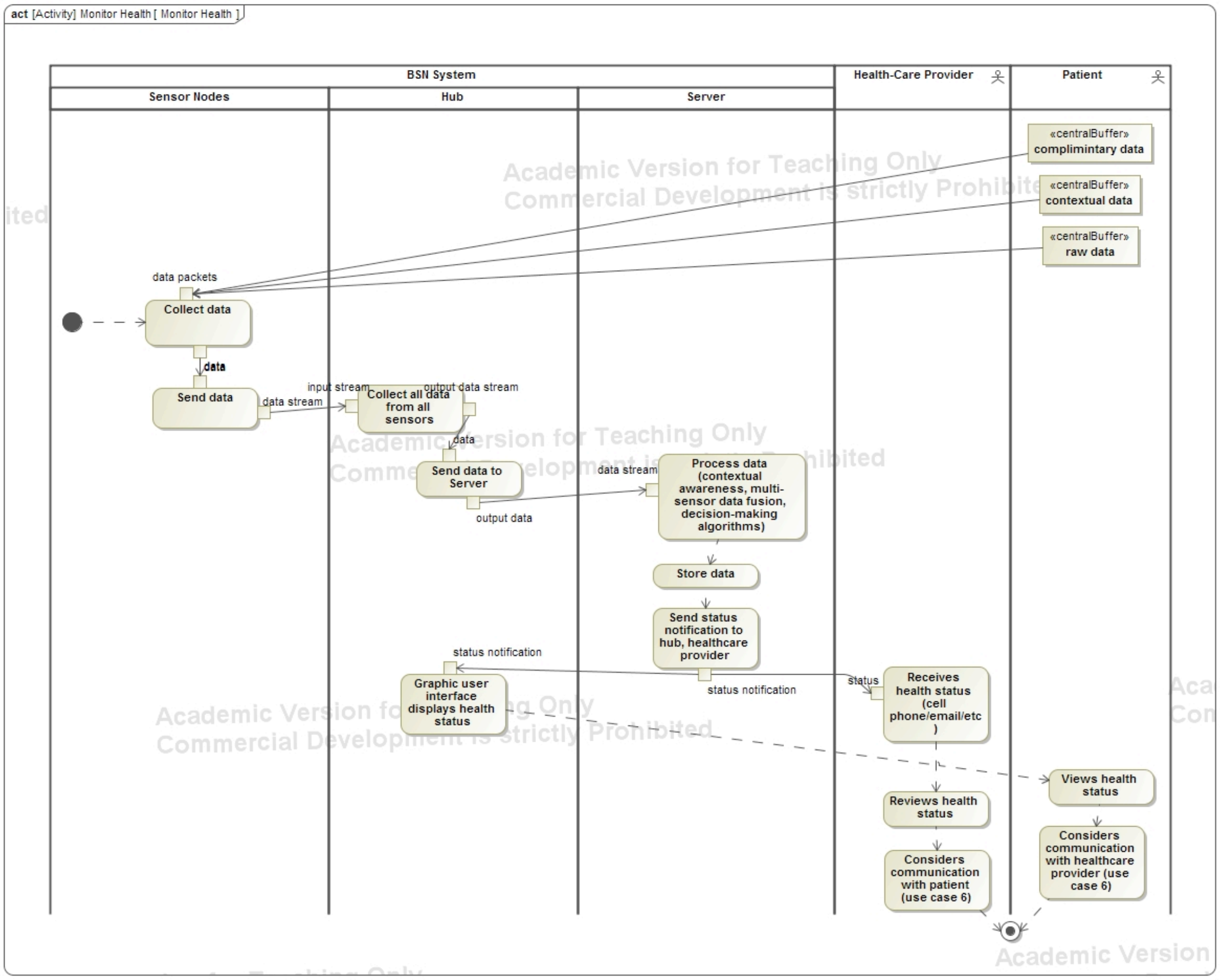

act [Activity] Monitor Health [ Monitor Health ]
BSN System
Health-Care Provider
Patient
Sensor Nodes
Hub
Server
«centralBuffer» complimintary data
«centralBuffer» contextual data
«centralBuffer» raw data
data packets
Collect data
data
Send data
data stream
input stream
Collect all data from all sensors
output data stream
data
Send data to Server
output data
data stream
Process data (contextual awareness, multi-sensor data fusion, decision-making algorithms)
Store data
Send status notification to hub, healthcare provider
status notification
Graphic user interface displays health status
status notification
status
Receives health status (cell phone/email/etc )
Views health status
Reviews health status
Considers communication with patient (use case 6)
Considers communication with healthcare provider (use case 6)
Academic Version for Teaching Only
Commercial Development is strictly Prohibited

Figure A.4: Use Case 4 Activity Diagram

*Post-Conditions*

Health of patient is being monitored by patient and healthcare provider.

*Derived Requirements*

1. The hub has storage capability to receive the raw physiological data.

2. Communication channel between the sensor nodes and hub is capable of transferring raw data (bandwidth-wise and interfacing).
3. Appropriate sensors can be selected to allow for multi-sensor data fusion.
4. Appropriate contextual awareness sensors can be selected for each node.
5. Communication channel between the hub and server is capable of transferring data (bandwidth-wise and interfacing).
6. Communication channel is secure between sensor nodes and hub.
7. Communication channel is reliable between sensor nodes and hub.
8. The server has storage capability to receive, store, and analyze incoming data.
9. Detection and decision-making algorithms are in place at the server to make conclusions about incoming data.
10. Capability of server to send notifications to patient hub and healthcare provider device (email/text message to cell phone/etc) based on conclusions from analysis, and store the timing and contents of the notification into that patient's database.
11. User interface of patient's hub device or healthcare provider's device (or any device with access to internet) is practical for use to view notifications.
12. User interface of patient's hub device or healthcare provider's device (or any device with access to internet) is practical for viewing their data and diagnosis.

Use Case 5: Store Accessible Health Records

*Goals 5*

1. Store data in integrative database.
2. Allow access to stored data via internet.

*Primary Actors*
Patient, Healthcare Provider

*Secondary Actors*
Not Available

*Pre-Conditions*
From Use Case 4, the data is from sensors and subsequently the hub is being sent to the server.

*Primary Flow*

1. Healthcare provider or patient requests to access stored data (through hub device or internet).
2. If the request is accepted (through something such as a username/password), then access to the health records is granted.

3. Healthcare provider or patients are capable of viewing the patient data stored in the server.

*Alternate Flow*
For the 2nd step of the primary flow: if the request to the stored data in the server is not authorized, then access to the health records is denied.

*Activity Diagram*
The following activity diagram depicts the flow of Use Case 5.

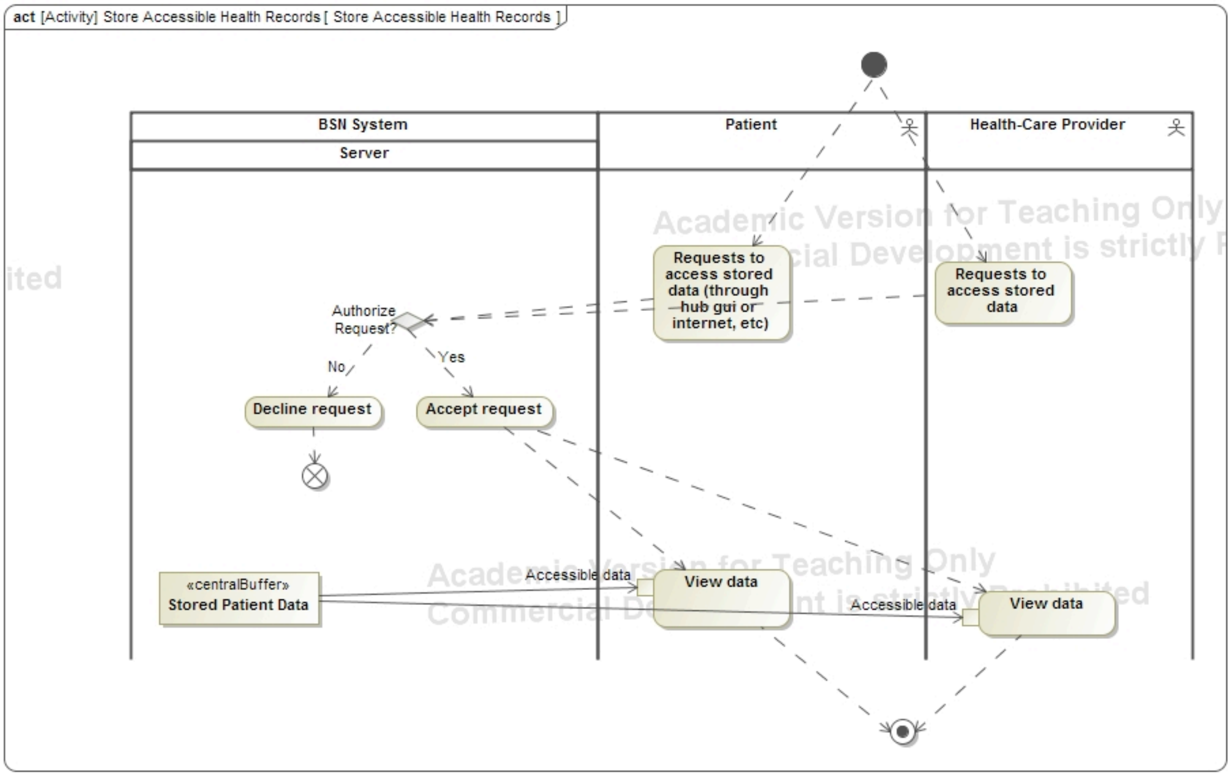

act [Activity] Store Accessible Health Records [ Store Accessible Health Records ]
BSN System
Server
Patient
Health-Care Provider
Requests to access stored data (through hub gui or internet, etc)
Requests to access stored data
Authorize Request?
No
Yes
Decline request
Accept request
«centralBuffer»
Stored Patient Data
Accessible data
View data
Accessible data
View data

Figure A.5: Use Case 5 Activity Diagram

*Post-Conditions*
Health records are stored and easily accessed by the patient and healthcare provider.

*Derived Requirements*

1. Server allows to securely store data coming from a patient into that specific patient's database.
2. User interface of patient or healthcare provider allows to securely access the patient's database server (username/pw encryption) from the hub or any device that has access to internet.
3. Access to server database for each particular patient is possible through a commonly used communication platform (such as the internet).

<u>Use Case 6:</u> Allow Patient-Healthcare Provider Communication

*Goals 6*

1. Maintain communication between healthcare provider and patient (patient can ask questions and the physician can give feedback, track progress).
2. Provide mechanism to acknowledge states of temporary lack of communication
3. Provide reliable communications [53]

*Primary Actors*

Patient, Healthcare Provider

*Secondary Actors*

Not Available

*Pre-Conditions*

BSN \ communication is set up and confirmed to be in operation.
From use case 4: the server sends a health status notification to the patient's hub (which has a graphic user interface) and the healthcare provider's device.

*Primary Flow*

1. Scenario 1: Healthcare provider provides feedback.
    a. Healthcare provider submits their feedback to the server via internet.
    b. Once submitted, the feedback is sent to the patient's hub graphical user interface.
    c. Patient reads the feedback off of the hub.
2. Scenario 2: Patient asks for feedback.
    a. Patient request feedback in hub.
    b. Hub transfers this request to the server.
    c. The server sends notification about this request to the healthcare provider's device.
    d. Healthcare provider submits response through his device to the server.
    e. Server sends this response back to the patient's hub.
    f. Patient reads the response off of the hub.
3. Scenario 3: Healthcare provider requests a checkup.
    a. Healthcare provider sends a request for a checkup with the patient, along with all time & date options for this checkup, to the server through their device.
    b. Server sends this request to the patient's hub.
    c. Patient sees the response off of the hub, and confirms a checkup time/date.
    d. Hub sends this confirmation to the server.
    e. Server relays a confirmation notification to the healthcare provider's device.
4. Scenario 4: Patient requests a checkup.

a. Patient sends a request for a checkup with the healthcare provider from their hub.
b. The hub relays this request to the server.
c. The server sends this request to the healthcare provider's device.
d. The healthcare provider provides a response through their device to the server.
e. Server sends this response to the patient's hub.
f. Patient reads the response off of the hub.

*Alternate Flow*

Each of these scenarios is optional, can be looped, and any scenario can lead to another one. For example, once a patient asks for feedback and received it from the healthcare provider, they may want to request a checkup.

In scenarios 1-4, the hub can be replaced by any device with internet access, which would give the patient access to the server itself.

*Sequence Diagram*

The communication involved in this use case was best depicted by the sequence diagram below.

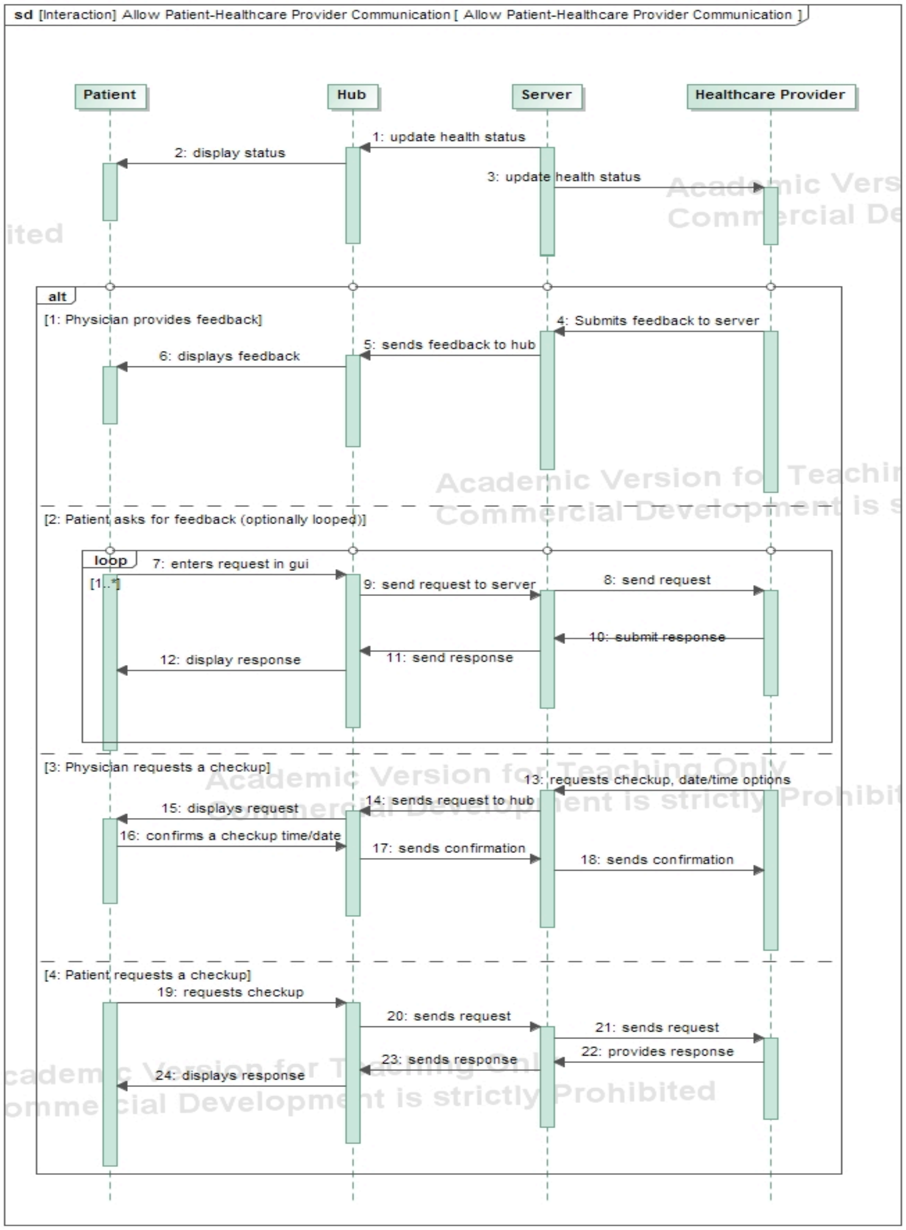


Figure A.6: Use Case 6 Sequence Diagram

*Post-Conditions*
Patient and healthcare provider are able to communicate back and forth between each other.

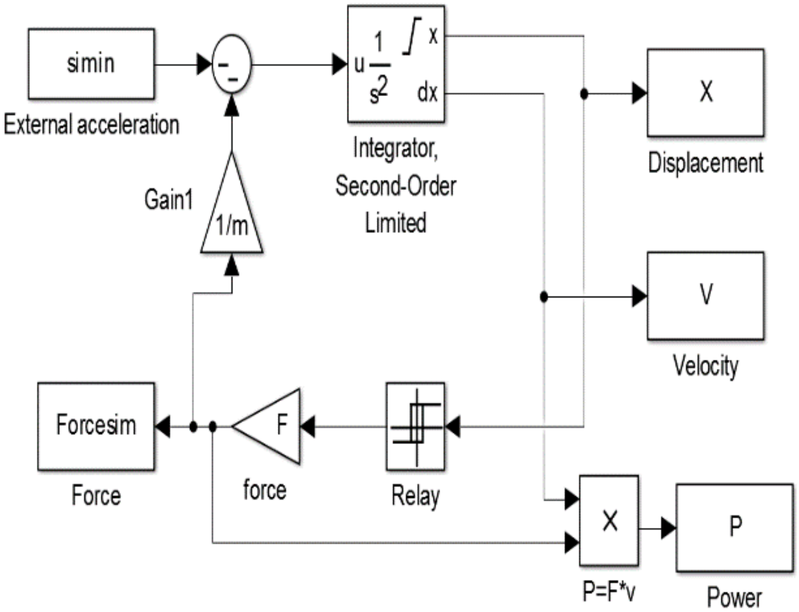
*Derived Requirements*

1. User interface of patient's hub device or healthcare provider's device (or any device with access to internet) is practical for use for communication between patient and healthcare.

2. Communication channel is secure between hub and server.
3. Communication channel is reliable between hub and server.

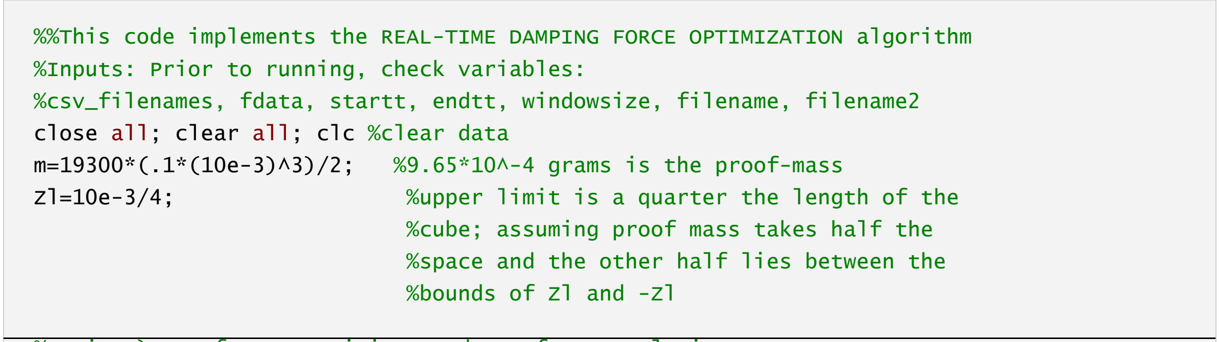
## Appendix B: Matlab Sources Code for CFPG Models

In this appendix, 6 source codes model, simulate, and analyze the CFPG device and their power generating capabilities. Appendix B.1 gives the code that implements the (real-time) optimization formulation in Equation 5 of this thesis. Appendix B.2 lays out a function utilized by the real-time optimization formulation. This function evaluates the objective function and design parameters of the optimization algorithm. Appendix B.3 lays out the code that implements the delayed-damping force adaptive tuning model described in this paper. Appendix B.4 is a piece of code that was written to aid in processing the accelerometer acceleration traces, and Appendix B.5 plots the optimization results. All three Matlab codes that simulate the CFPG model utilize the Simulink model in the diagram below. All code has been published with Matlab 2014a: Student Version.

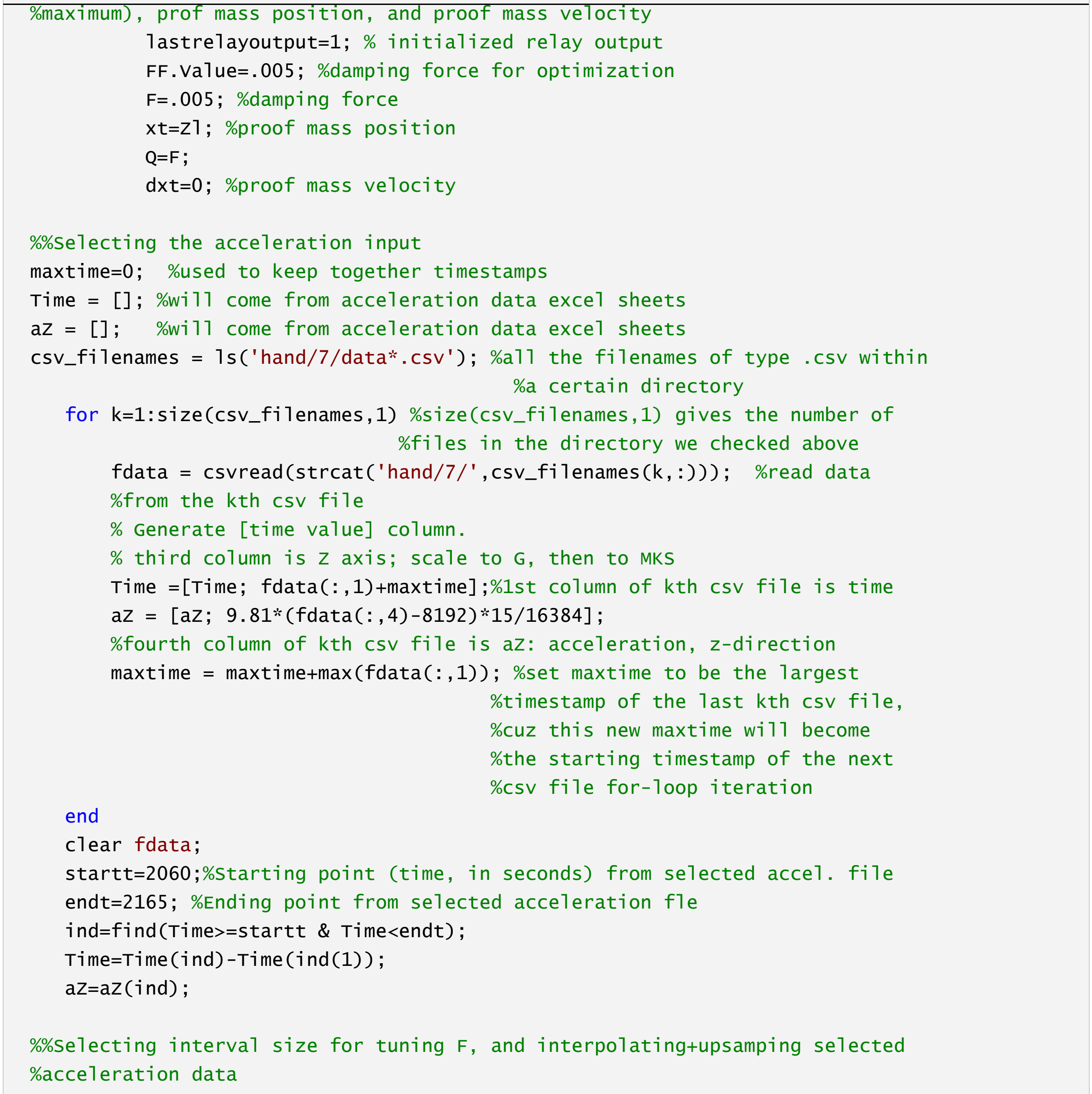


Simulink Model of CFPG, referred to code as CFPG_Simulink_Model

## *Appendix B.1: Source Code for Real-Time Damping Force Optimization*



```
%%This code implements the REAL-TIME DAMPING FORCE OPTIMIZATION algorithm
%Inputs: Prior to running, check variables:
%csv_filenames, fdata, startt, endtt, windowsize, filename, filename2
close all; clear all; clc %clear data
m=19300*(.1*(10e-3)^3)/2;   %9.65*10^-4 grams is the proof-mass
Zl=10e-3/4;                  %upper limit is a quarter the length of the
                             %cube; assuming proof mass takes half the
                             %space and the other half lies between the
                             %bounds of Zl and -Zl
```



```
%maximum), prof mass position, and proof mass velocity
          lastrelayoutput=1; % initialized relay output
          FF.Value=.005; %damping force for optimization
          F=.005; %damping force
          xt=Zl; %proof mass position
          Q=F;
          dxt=0; %proof mass velocity

%%Selecting the acceleration input
maxtime=0;  %used to keep together timestamps
Time = []; %will come from acceleration data excel sheets
aZ = [];   %will come from acceleration data excel sheets
csv_filenames = ls('hand/7/data*.csv'); %all the filenames of type .csv within
                                          %a certain directory
   for k=1:size(csv_filenames,1) %size(csv_filenames,1) gives the number of
                                %files in the directory we checked above
       fdata = csvread(strcat('hand/7/',csv_filenames(k,:)));  %read data
       %from the kth csv file
       % Generate [time value] column.
       % third column is z axis; scale to G, then to MKS
       Time =[Time; fdata(:,1)+maxtime];%1st column of kth csv file is time
       aZ = [aZ; 9.81*(fdata(:,4)-8192)*15/16384];
       %fourth column of kth csv file is aZ: acceleration, z-direction
       maxtime = maxtime+max(fdata(:,1)); %set maxtime to be the largest
                                        %timestamp of the last kth csv file,
                                        %cuz this new maxtime will become
                                        %the starting timestamp of the next
                                        %csv file for-loop iteration
   end
   clear fdata;
   startt=2060;%Starting point (time, in seconds) from selected accel. file
   endt=2165; %Ending point from selected acceleration fle
   ind=find(Time>=startt & Time<endt);
   Time=Time(ind)-Time(ind(1));
   aZ=aZ(ind);

%%Selecting interval size for tuning F, and interpolating+upsamping selected
%acceleration data
```

```
windowsize=[1]; %Interval size for adaptive tuning of damping force
maxtime=windowsize; %now maxtime is equal to the windowsize
[pts, maxlength]=maximumlength(Time,aZ,1); %(up)samples for interpolation,
% third input is 1 (for windowsize) because we want to interpolate by
% looking for the maximum number of points in 1-second windows across
% our data...our accelerometer gave us a value of Hz it samples as in we
% should expect a certain max number of points in each one second window
pts=pts*4; %Upsampling
maxlength=maxlength*4; %Upsampling
Timediff=Time(end);
Timeint=linspace(0,Timediff,pts); %Linearly-spaced interplation
aZ=interp1(Time,aZ,Timeint,'pchip','extrap')'; %Interpolation method: PCHIP

Time=Timeint';
clear Timeint;
ind=find(Time<=max(Time));
daytime=Time(ind(end));

Time=Time(ind)-Time(ind(1)); %Interpolated Time, selected from acceleration
%file, shifted to start at Time=0
aZ=aZ(ind); %Interpolated acceleration z-axis

step=Timediff/pts; %make sure step size (time, in seconds) between
%discrete data points is accurate


%%While loop for optimization. Every iteration in the while loop
%%corresponds to optimizing one window frame (of pre-specified interval
%%size, 'windowsize') by tuning F and maximizing output power

%Initialize counters used in optimization while loop
count=0;
i = 0;
remF=[];

while (i<daytime) %As long as the while-loop is still going thru selected
                  %acceleration data
     if (i+windowsize)>daytime;
         ind = find(Time>=i & Time<=daytime);
         % Finds all the indices of the current window of length windowsize
         maxtime=daytime-i;
     else
         ind=find(Time>=i & Time<i+windowsize);
     end
     i = i+windowsize; %shift i index to start the next window for the next
     %iteration of the while loop


     alpha = Time(ind)-Time(ind(1)); %Setting time input for simulation
     %(arbitrarily called alpha)
     a.time=[];%Don't put in time series for optimization
     aZindic= aZ(ind); %acceleration input into simulation
```

```
    a.signals.values=aZindic;
    a.signals.dimensions=1;
    simin=[aZindic];

    starttime=(ind(1)*step)-step-(count*windowsize); %Simulink start-time
    %of current window frame
    maxtime=step*length(alpha)-step; %Simulink end-time of current
    %window frame

    count=count+1; %frame or while-loop counter



    if length(remF)==0; %remF is an indicator for whether proof mass was
        %midflight in previous acceleration windowframe. if length(remF)=0
        %then it was not midflight and F is allowed to be optimized.
        %otherwise F remains constant

        %upload initial points from previous acceleration window frame (or
        %initial settings)
        filename=strcat('june10/current',num2str(count+num),...
            'hand 7 one s.mat'); %Change directory and name
        save(filename,'a','Zl','FF','xt','dxt','m','lastrelayoutput');
        hws=get_param('CFPG_Simulink_Model','modelworkspace');
        hws.clear;
        hws.DataSource='MAT-File';
        hws.FileName=filename;
        hws.reload;
        delete(filename);
        clear filename;


        %Call CFPG Simulink Model, specify bounds for optimization
        %design variable F
        sys='CFPG_Simulink_Model';
        open_system(sys);
        FF=sdo.getParameterFromModel('CFPG_Simulink_Model','FF.Value');
        FF.Minimum=1e-5;
        FF.Maximum=.01;

        %Log following power signal models from CFPG Simulink Model
        Powers=Simulink.SimulationData.SignalLoggingInfo;
        Powers.BlockPath='CFPG_Simulink_Model/Mean';
        Powers.OutputPortIndex=1;
        simulator=sdo.SimulationTest('CFPG_Simulink_Model');
        simulator.LoggingInfo.Signals=[Powers];

        %Specify bounds on objective (power output)
        MaxPowers=sdo.requirements.SignalBound;
        set(MaxPowers,...
            'BoundTimes', [0 20],...
            'BoundMagnitudes', [1000 1000],...
            'Type', '<=');
        requirements = struct(...%specify req'ments for the logged signals
            'MaxPowers', MaxPowers);

       %Calls function that evluates CFPG model design
        evalDesign2=@(p) CFPG_Simulink_Model_Design(p,...
            simulator,requirements);
        initDesign=evalDesign2(FF); %After evaluation, this calls
        %objective function to simulate model and evaluate design req's
        initDesign.Cleq;
        initDesign.F;


        opts=sdo.OptimizeOptions; %Set optimization
        opts.Method='patternsearch';%Select optimization method:
                                    %Pattern Search Algorithm
        opts.MethodOptions.TolFun=1e-7;%Termination Tol on Objective Fun
        opts.MethodOptions.TolX=1e-10; %Termination Tol on constraint(n/a)
        opts.MethodOptions.TolMesh=1e-5;%Termination Tol on Obj Fun + Mesh
        opts.MethodOptions.CompletePoll='on'; %Complete Point Polling
        opts.MethodOptions.CompleteSearch='on'; %Complete Point Searching
        opts.MethodOptions.MeshContraction = .99; %Mesh Contraction
        opts.MethodOptions.InitialMeshSize=.01; %Initial mesh size
        opts.MethodOptions.MeshExpansion=1.03; %Mesh expansion
        opts.MethodOptions.MaxIter=5000;%Termination by maximum iterations
        opts.MethodOptions.MaxFunEval=20*5000;%Termination by max Fun. Evals.
        opts.MethodOptions.Cache='on'; %Use cache to compare current
        %damping force value to see if they are close to previously
        %computed F value...if within tolerance, computation skipped
        opts.MethodOptions.CacheTol=1e-5; %Cache Tolerance
        opts.MethodOptions.CacheSize=1000; %Cache Size

        %Starts optimization
        [pOpt,optInfo]=sdo.optimize(evalDesign2,FF,opts);

        if optInfo.F==0 %If no power was produced, then the code
              pOpt.Value=Q;%keeps F value of previous window frame
        end

        sdo.setValueInModel('CFPG_Simulink_Model',pOpt); %Updates model
                                                         %variable values

        numofiterraw=optInfo.iterations; %used if we want to see # of
                                         %iterations it took pattern
                                         %search to find an optimal value
                                         %for current window frame F
        reasonfortermination=optInfo.exitflag;%used if we want to check
                                   %exit flag for reason of termination
        avgpower=optInfo.F; %Maximized average power (after optimzation
                            % of current window frame)
        F=pOpt.Value; %Optimal value of F (after optimization)

    else %proof mass was midflight in previous window frame, i.e.
        %F remains the same as in the previous window frame
```

```
         F=remF; %remF is an indicator for whether proof mass was
         %midflight in previous acceleration windowframe.
         avgpower=777; %Arbitrary number indicating optimization was skipped
                       %for current window frame
         numofiterraw=777;
         reasonfortermination=777;
     end


     sim('CFPG_Simulink_Model'); %Simulate our model to see output
                                        %with optimal value of F
     %Saving values of current window
     fsim=Forcesim.signals.values; %Save damping force value
     v=V.signals.values; %Save prof mass velocity
     x=X.signals.values; %Save proof mass positions
     ptime=P.time; %Save time series of simulaitons
     pp=P.signals.values; %Save instantaneous power values
     simaccel=theaccel.signals.values; %Save input acceleration values
     lastrelayoutput=LWO.signals.values(end); %Save relay output
     filename2=strcat('june10/',num2str(count+num),'hand 7 one s');
     save(filename2,'fsim',...
                  'v','alpha','aZindic','x','simaccel','ptime','step',...
                  'pp','avgpower','starttime','numofiterraw',...
                  'lastrelayoutput','reasonfortermination');

     %Discrete data means that we have to make sure initial position and
     %velocity of next window frame has changed with next step size
     xt=x(end)+(v(end)*step);
     if v(end)==0
         dxt=v(end);
         remF=[];
     else
         dxt=v(end)+((-simaccel(end)-(fsim(end)/m))*step);
         remF=abs(fsim(end));
     end
     F=abs(fsim(end)); %Initialize F value of next window frame with current
     %window frame
     FF.Value=abs(fsim(end));
     Q=abs(fsim(end));
end
```

## ***Appendix B.2: Source Code for Evaluation of Real-Time Damping Force Optimization Formulation***

```
%function that evluates CFPG model design
function design = CFPG_Simulink_Model_Design(p,simulator,requirements)

%Simulate model using the simulator input argument to simulate the model
%and log model signals. Ensure model is simulated with chosen parameters
simulator.Parameters=p;
simulator=sim(simulator); %simulates
%Simulation signal log defined by SignalLoggingName porperty
logName=get_param('CFPG_Simulink_Model','SignalLoggingName');
simLog=get(simulator.LoggedData,logName);
%Evaluate design requirements and add design objective to minimize negative
%of power (so maximize positive)
Powers=find(simLog,'Powers');
cPowers=evalRequirement(requirements.MaxPowers,Powers.Values);
design.Cleq=[cPowers(:)];
temp=permute(Powers.Values.Data,[3 2 1]);
design.F=-temp(end);
end
```

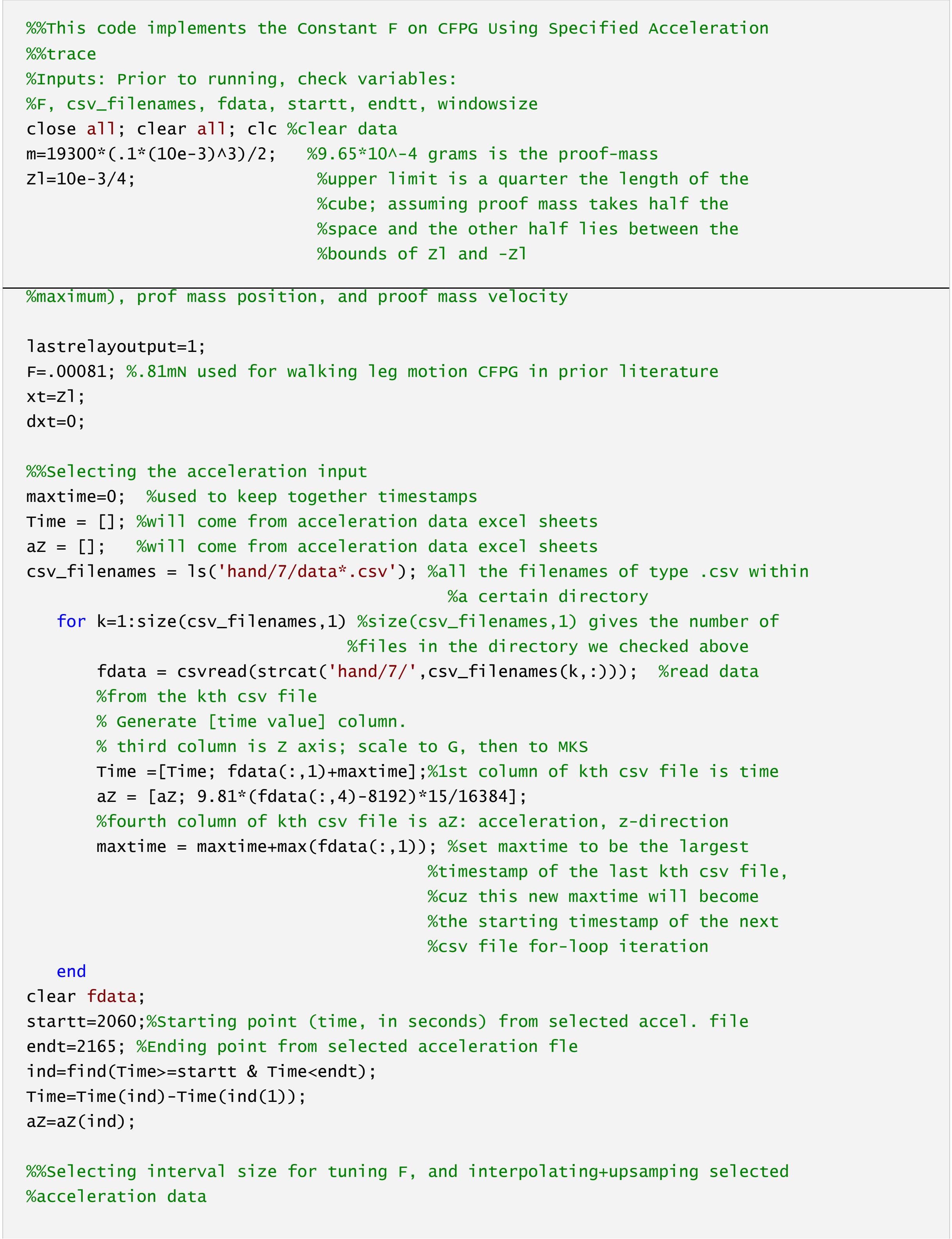
## ***Appendix B.3: Source Code for Delayed-Damping Force Adaptive Tuning***

```
%%This code implements the DELAYED DAMPING FORCE ADAPTIVE TUNING algorithm
%Inputs: Prior to running, check variables:
%F, windowsize, count, A, c, startt, endtt, csv_filenames, fdata
close all; clear all; clc %clear data
m=19300*(.1*(10e-3)^3)/2;   %9.65*10^-4 grams is the proof-mass
Zl=10e-3/4;                  %upper limit is a quarter the length of the
                             %cube; assuming proof mass takes half the
                             %space and the other half lies between the
                             %bounds of Zl and -Zl
```

```
%maximum), prof mass position, and proof mass velocity
lastrelayoutput=1; %initialized relay output
F=[0.005]; %Choose .005 if looking at hand/arm data or .01 if leg data
xt=Zl; %proof mass position
dxt=0; %proof mass velocity

%%Interval length of window frames, should be consistent with interval of
%window frames of
windowsize=[1];
num=1/windowsize;

%%Opens optimized M-files from real-time optimization, stores optimal F #'s
%Choose files and make sure names for variables 'A' and 'c' match
for count=200:1:239 %Number of windows with interval length windowsize
                    %that wants to be looked at
    A=exist(strcat('june10/',num2str(count+1),'hand 33 one s.mat'));
    if A~=0;
        c=open(strcat('june10/',num2str(count+1),'hand 33 one s.mat'));
         F=[F; c.fsim(1)];
    else
    end
end;
allF=F(1:length(F)-F);%Iteratively store values of optimal F with a delay
%So second value of allF would be the optimal value of F for the first
%window frame
allF=abs(allF);




%%Selecting the acceleration input
maxtime=0;  %used to keep together timestamps
Time = []; %will come from acceleration data excel sheets
aZ = [];   %will come from acceleration data excel sheets
csv_filenames = ls('hand/7/data*.csv'); %all the filenames of type .csv within
                                          %a certain directory
   for k=1:size(csv_filenames,1) %size(csv_filenames,1) gives the number of
                                 %files in the directory we checked above
      fdata = csvread(strcat('hand/7/',csv_filenames(k,:)));  %read data
```

```
       %from the kth csv file
       % Generate [time value] column.
       % third column is Z axis; scale to G, then to MKS
       Time =[Time; fdata(:,1)+maxtime];%1st column of kth csv file is time
       aZ = [aZ; 9.81*(fdata(:,4)-8192)*15/16384];
       %fourth column of kth csv file is aZ: acceleration, z-direction
       maxtime = maxtime+max(fdata(:,1)); %set maxtime to be the largest
                                        %timestamp of the last kth csv file,
                                        %cuz this new maxtime will become
                                        %the starting timestamp of the next
                                        %csv file for-loop iteration
   end
   clear fdata;
   startt=2060;%Starting point (time, in seconds) from selected accel. file
   endt=2165; %Ending point from selected acceleration fle
   ind=find(Time>=startt & Time<endt);
   Time=Time(ind)-Time(ind(1));
   aZ=aZ(ind);

%%Having selected interval size for tuning F, Over here the code
%interpolates and upsamples the chosen acceleration data
[pts, maxlength]=maximumlength(Time,aZ,1);
pts=pts*4; %Upsampling
maxlength=maxlength*4; %Upsampling
Timediff=Time(end);
Timeint=linspace(0,Timediff,pts); %Linearly-spaced interplation
aZ=interp1(Time,aZ,Timeint,'pchip','extrap')'; %Interpolation method: PCHIP

Time=Timeint';
clear Timeint;
ind=find(Time<=max(Time));
daytime=endt-startt;

Time=Time(ind)-Time(ind(1)); %Interpolated Time, selected from acceleration
%file, shifted to start at Time=0
aZ=aZ(ind); %Interpolated acceleration z-axis

step=Timediff/pts; %make sure step size (time, in seconds) between
%discrete data points is accurate

%%While loop for delayed adaptive tuning. Every iteration in the while loop
%%corresponds to optimizing one window frame (of pre-specified interval
%%size, 'windowsize') by using the optimal value of F from the previous
%%frame

%Initialize counters used in optimization while loop
count=0;
i = 0;
avgp=[];
while (i<daytime) %As long as the while-loop is still going thru selected
                  %acceleration data

        % Finds all the indices of the current window of length windowsize
        ind=find(Time>=i & Time<i+windowsize);
        i = i+windowsize; %shift i index to start the next window for the next
        %iteration of the while loop
         alpha = Time(ind)-Time(ind(1)); %Setting time input for simulation
         %(arbitrarily called alpha)
         a.time=[];%Don't put in time series for optimization
         aZindic= aZ(ind); %acceleration input into simulation
         a.signals.values=aZindic;
         a.signals.dimensions=1;
         simin=[aZindic];

         starttime=(ind(1)*step)-step-(count*windowsize); %Simulink start-time
         %of current window frame
         maxtime=step*length(alpha)-step; %Simulink end-time of current
         %window frame

        count=count+1;
        F=allF(count); %choose F-value
        sim(CFPG_Simulink_Model);
        pp=P.signals.values; %instantaneous power from simulation
        lastrelayoutput=LWO.signals.values(end); %last relay output


        avgp=[avgp;mean(pp)];%store avg harvested power for each interval
end
mean(avgp)%Average harvested power for selected acceleration data
```

## ***Appendix B.4: Source Code for Constant Damping Force***



```
%%This code implements the Constant F on CFPG Using Specified Acceleration
%%trace
%Inputs: Prior to running, check variables:
%F, csv_filenames, fdata, startt, endtt, windowsize
close all; clear all; clc %clear data
m=19300*(.1*(10e-3)^3)/2;   %9.65*10^-4 grams is the proof-mass
Zl=10e-3/4;                  %upper limit is a quarter the length of the
                             %cube; assuming proof mass takes half the
                             %space and the other half lies between the
                             %bounds of Zl and -Zl
```

```
%maximum), prof mass position, and proof mass velocity

lastrelayoutput=1;
F=.00081; %.81mN used for walking leg motion CFPG in prior literature
xt=Zl;
dxt=0;

%%Selecting the acceleration input
maxtime=0;  %used to keep together timestamps
Time = []; %will come from acceleration data excel sheets
aZ = [];   %will come from acceleration data excel sheets
csv_filenames = ls('hand/7/data*.csv'); %all the filenames of type .csv within
                                          %a certain directory
   for k=1:size(csv_filenames,1) %size(csv_filenames,1) gives the number of
                                %files in the directory we checked above
       fdata = csvread(strcat('hand/7/',csv_filenames(k,:)));  %read data
       %from the kth csv file
       % Generate [time value] column.
       % third column is Z axis; scale to G, then to MKS
       Time =[Time; fdata(:,1)+maxtime];%1st column of kth csv file is time
       aZ = [aZ; 9.81*(fdata(:,4)-8192)*15/16384];
       %fourth column of kth csv file is aZ: acceleration, z-direction
       maxtime = maxtime+max(fdata(:,1)); %set maxtime to be the largest
                                        %timestamp of the last kth csv file,
                                        %cuz this new maxtime will become
                                        %the starting timestamp of the next
                                        %csv file for-loop iteration
   end
clear fdata;
startt=2060;%Starting point (time, in seconds) from selected accel. file
endt=2165; %Ending point from selected acceleration fle
ind=find(Time>=startt & Time<endt);
Time=Time(ind)-Time(ind(1));
aZ=aZ(ind);

%%Selecting interval size for tuning F, and interpolating+upsamping selected
%acceleration data
```

```
windowsize=[1]; %Interval size for adaptive tuning of damping force
maxtime=windowsize; %now maxtime is equal to the windowsize
[pts, maxlength]=maximumlength(Time,aZ,1); %(up)samples for interpolation,
% third input is 1 (for windowsize) because we want to interpolate by
% looking for the maximum number of points in 1-second windows across
% our data...our accelerometer gave us a value of Hz it samples as in we
% should expect a certain max number of points in each one second window
pts=pts*4; %Upsampling
maxlength=maxlength*4; %Upsampling
Timediff=Time(end);
Timeint=linspace(0,Timediff,pts); %Linearly-spaced interplation
aZ=interp1(Time,aZ,Timeint,'pchip','extrap')'; %Interpolation method: PCHIP

Time=Time(ind)-Time(ind(1)); %Interpolated Time, selected from acceleration
%file, shifted to start at Time=0
aZ=aZ(ind); %Interpolated acceleration z-axis

step=Timediff/pts; %make sure step size (time, in seconds) between
%discrete data points is accurate



alpha = Time(ind)-Time(ind(1)); %Setting time input for simulation
%(arbitrarily called alpha)
a.time=[];%Don't put in time series for optimization
aZindic= aZ(ind); %acceleration input into simulation
a.signals.values=aZindic;
a.signals.dimensions=1;
simin=[aZindic];

starttime=(ind(1)*step)-step-(count*windowsize); %Simulink start-time
%of current window frame
maxtime=step*length(alpha)-step; %Simulink end-time of current
%window frame


sim('CFPG_Simulink_Model');
pp=P.signals.values; %Instantaneous Power signals from model simulation
mean(pp) %Average harvested power from simulation
```

## ***Appendix B.5: Source Code for Data Up-sampling and Interpolation***

```
%%Function that upsamples for interpolation,
% third input is 1 (for windowsize) because we want to interpolate by
% looking for the maximum number of points in 1-second windows across
% our data...The accelerometer gave us a value of Hz it samples as in we
% should expect a certain max number of points in each one second window
function [pts maxlength]=maximumlength(Time,aZ,windowsize)

maxlength=0;
i=0;
indic=find(Time<=max(Time));
daytime=Time(indic(end));
count=0;
while(i<daytime)%Looks for 1-secondwindow with highest # of sampled points
     if (i+windowsize)>daytime;
         ind = find(Time>=i & Time<daytime);% Find all the indices of the
                                    %current window of length windowsize
     else
         ind=find(Time>=i & Time<i+windowsize);
     end
     i = i+windowsize; %shift i index to start the next window for
                       %the next iteration of the while loop

     windleng=length(aZ(ind));
     if windleng>maxlength
         maxlength=windleng;
     else
     end
     count=count+1;
end
numofwinds=count;%Number of 1-second windows in selected acceleration trace
pts=numofwinds*maxlength; %Maximum number of points in each window
```

### *Appendix B.6: Analyzing Optimization M-Files and Plotting Results*

```
%%Read and plot saved optimization M-files. Each M-file represents a window
%frame of certain interval length
%Inputs: Prior to running, look at variables:
%num, count, A, and c
clear all; close all; clc
K=[];
L=[];
M=[];
firstiter=[];
seconditer=[];
thirditer=[];
reas=[];
num=1/40; %change denominator to windowsize
for count=200 %change this so that it reflects number of M-files
    A=exist(strcat('june10/',num2str(count+1),'hand 33 forty s.mat')); %change name
    if A~=0;
        c=open(strcat('june10/',num2str(count+1),'hand 33 forty s.mat')); %change name
          reas=[reas; c.reasonfortermination c.lastrelayoutput];
        c.ptime=c.ptime+(count/num)+c.starttime-(8*25*(1/num))+25; %
         if length(L)==0;
         elseif abs(c.fsim(1))==abs(L(end,3));
                 thirditer=[thirditer; count+1, mean(c.pp)];
         else
         end
         L=[L;c.ptime, c.pp, c.fsim, c.x, mean(c.pp)*ones(size(c.pp)), c.v];
        c.alpha=c.alpha+(count/num)-(8*25*(1/num))+25; %
         M=[M;c.ptime, c.simaccel];
         end

    else
    end
end;

%Plots below, labels describe type of plot
a=L(:,2); %or L(:,5) for avg power instead of instantaneous power
for i=1:1:length(a);
    b(i)=sum(a(1:i));
end
figure;
subplot(3,2,1);
plot(M(:,1),M(:,2));
ylabel('acceleration');
xlabel('time (s)')
subplot(3,2,2);
plot(L(:,1),L(:,2));
title('instantaneous power');
xlabel('time (s)')
subplot(3,2,3);
plot(L(:,1),abs(L(:,3)));
```

```
xlabel('time (s)')
title('F');
subplot(3,2,4);
plot(L(:,1),L(:,4));
xlabel('time (s)')
title('position');
subplot(3,2,5);
plot(L(:,1),L(:,5));
xlabel('seconds')
title('avg power');
subplot(3,2,6);
plot(L(:,1),b);
xlabel('seconds');
title('energy');

figure;
plot(L(:,1),L(:,4));
xlabel('time (s)')
title('position');

figure;
plot(L(:,1),L(:,5));
xlabel('seconds') %change name
title('avg power');

figure;
plot(L(:,1),L(:,2));
title('instantaneous power');
xlabel('time (s)')

figure;
plot(L(:,1),abs(L(:,3)));
xlabel('time (s)')
title('F');
figure;
plot(L(:,1),(L(:,6)));
xlabel('time (s)')
title('speed');

M(:,1)=M(:,1)-25;
figure;
plot(M(:,1),M(:,2));
ylabel('Acceleration (m/s^2)');
xlabel('Time (s)')
title('Arm 5');

  mean(L(:,2))%Average Harvested Power
```

# Bibliography

[1] McMurray, J.J., and S. Stewart. "Epidemiology, aetiology, and prognosis of heart failure." *Education in Heart* (2000).

[2] Panel Notes. Healthcare Panel. Conference on Systems Engineering Research 2013 (Atlanta, GA), March 21, 2013.

3] Hao, Y., and R. Foster. "Wireless body sensor networks for health-monitoring applications." *Physiological Measurements* (November 2008).

[4] Institute of Medicine. "Crossing the quality chasm: a new health system for the 21st century." Consensus Report, March 1, 2001.

[5] Brown, S.J. "Personalized Display of Health Information. U.S." Patent No. 7,555,436, June 30, 2009.

[6] Fierz, W. "Challenge of personalized health care: To what extent is medicine already Individualized and what are the future trends?" *Med Sci Monitor* (2004).

[7] de Weck, O.L., D. Ross, C.L. Magee, and C.M. Vest. *Engineering Systems: Meeting Human Needs in a Complex Technological World*. MIT Press, 2011.

[8] Pantelopoulos, A., and N.G. Bourbakis . "A Survey on Wearable Sensor-Based Systems for Health Monitoring and Prognosis." *IEEE Transactions on Systems, Man, and Cybernetics-Part C: Applications and Reviews* (January 2010).

[9] Chen, M, S. Gonzalez, A. Vasilakos, H. Cao, and V.C.M. Leung. "Body Area Networks: A Survey. Mobile Network Applications." *Mobile Networks and Applications* 16, no. 2 (August 2010): 171-193. doi:10.1007/s1103-010-0260-8.

[10] Friedenthal, S., A. Moor, and R. Steiner. *A Practical Guide to SysML, 2nd Edition: The Systems Modeling Language*. The MK/OMG Press, 2011.

[11] Wang, J., and Q. Wang. *Body Area Communications: Channel Modeling, Communication Systems, and EMC*. Wiley-IEEE Press, 2013.

[12] Obrist, B., and S. Hegnauer. "A microwave powered data transponder." *Sensors Actuators A, Phys.* 46 (1995): 244-246.

[13] Suzuki, S., T. Katane, H. Saotome, and O. Saito. "A proposal of electric power generating system for implanted medical devices." *IEEE Trans. Magn.* 35 (September 1999): 3586-3588.

[14] Sangani, K. "Power solar- The sun in your pocket." *Eng. Technology* 2, no. 8 (August 2007): 36-38.

[15] Mantiply, E. D., K. R> Pohl, S. W. Poppell, and J. Murphy. “Summary of measured radiofrequency electric and magnetic fields (10 kHz to 30 GHz) in the general and work environment.” *Bioelectromagnetics* 18, no. 8 (1997): 563-577.

[16] SEIKO. “SEIKO World’s First.” SEIKO Watches. Accessed September, 2014. http://www.seikowatches.com/heritage/worlds_first.html.

[17] Mitcheson, Paul D., Eric M. Yeatman, G. Kondala Rao, Andrew S> Holmes, and Tim C. Green. “Energy Harvesting from Human and Machine Motion for Wireless Electronic Devices.” *Proceedings of the IEEE* 96, no. 9 (September 2008): 1457-1486. doi: 10.1109/JPROC.2008.927494.

[18] Dadfarnia, Mehdi. Kamran Sayrafian, Paul D. Mitcheson, and John S. Baras. “Maximizing Output Power of a CFPG Micro Energy-Harvester for Wearable Medical Sensors.” Paper presented at the MobiHealth Conference, Athens, Greece, November 2-5, 2014.

[19] Yang, Guang-Zhong, and Magdi Yacoub. "Body sensor networks." Springer 2006.

[20] Roundy, Shad, Paul Kenneth Wright, and Jan M. Rabaey. *Energy scavenging for wireless sensor networks*. Norwell, 2003.

[21] Kazmierski, Tom J., and Steve Beeby. *Energy Harvesting Systems: Principles, Modeling and Applications.* Springer 2011.

[22] Gandu, Kondalarao. “Power Processing for Electrostatic Microgenerators.” PhD diss., Imperial College, 2011.

[23] Mitcheson, Paul D., Tim C. Green, Eric M. Yeatman, and Andrew S. Holmes. "Architectures for vibration-driven micropower generators." *Microelectromechanical Systems, Journal of* 13, no. 3 (2004): 429-440.

[24] Mitcheson, Paul D., Tim C. Green, and Eric M. Yeatman. "Power processing circuits for electromagnetic, electrostatic and piezoelectric inertial energy scavengers." *Microsystem Technologies* 13, no. 11-12 (2007): 1629-1635.

[25] Shenck, Nathan S., and Joseph A. Paradiso. "Energy scavenging with shoe-mounted piezoelectrics." *Ieee Micro* 21, no. 3 (2001): 30-42.

[26] Mitcheson, Paul D., Tom Sterken, Cairan He, M. Kiziroglou, E. M. Yeatman, and R. Puers. "Electrostatic microgenerators." *Measurement and Control* 41, no. 4 (2008): 114-119.

[27] Von Buren, Thomas, Paul D. Mitcheson, Tim C. Green, Eric M. Yeatman, Andrew S. Holmes, and Gerhard Troster. "Optimization of inertial micropower

generators for human walking motion." *Sensors Journal, IEEE* 6, no. 1 (2006): 28-38.

[28] Mitcheson, Paul David. "Analysis and optimisation of energy-harvesting micro-generator systems." PhD diss., Imperial College, 2005.

[29] Mitcheson, P. D., D. C. Yates, E. M. Yeatman, T. C. Green, and A. S. Holmes. "Modelling for optimisation of self-powered wireless sensor nodes." *Proc. Int. Wkshp. Wearable and Implantable Body Sensor Networks (Apr. 2005)* (2005).

[30] Mitcheson, Paul D., Peng Miao, Bernard H. Stark, E. M. Yeatman, A. S. Holmes, and T. C. Green. "MEMS electrostatic micropower generator for low frequency operation." *Sensors and Actuators A: Physical* 115, no. 2 (2004): 523-529.

[31] Mitcheson, Paul D., and Tim C. Green. "Maximum effectiveness of electrostatic energy harvesters when coupled to interface circuits." *Circuits and Systems I: Regular Papers, IEEE Transactions on* 59, no. 12 (2012): 3098-3111.

[32] Yarkony, Nathalie, Kamran Sayrafian-Pour, and Antonio Possolo. "Statistical modeling of harvestable kinetic energy for wearable medical sensors." In *World of Wireless Mobile and Multimedia Networks (WoWMoM), 2010 IEEE International Symposium on a*, pp. 1-5. IEEE, 2010.

[33] Gorlatova, Maria, John Sarik, Mina Cong, Ioannis Kymissis, and Gil Zussman. "Movers and shakers: Kinetic energy harvesting for the internet of things." *arXiv preprint arXiv:1307.0044* (2013).

[34] Wei, Hu, Hu Guoqing, Wei Xin, and Xie Xiaozhu. "Modeling and Simulation of Electrostatic Comb-drive Actuators with Modelica." In *Measuring Technology and Mechatronics Automation (ICMTMA), 2010 International Conference on*, vol. 2, pp. 679-682. IEEE, 2010.

[35] Miao, P., P. D. Mitcheson, A. S. Holmes, E. M. Yeatman, T. C. Green, and B. H. Stark. "MEMS inertial power generators for biomedical applications." *Microsystem Technologies* 12, no. 10-11 (2006): 1079-1083.

[36] Torczon, Virginia. "On the convergence of pattern search algorithms." *SIAM Journal on optimization* 7, no. 1 (1997): 1-25.

[37] Gulf Coast Data Concepts, LLC. “Data Acquisition Products.” Gulf Coast Data Concepts Online Store. Last Accessed: October, 2013. http://www.gcdataconcepts.com/products.html

[38] Mathworks, Inc. “Global Optimization Toolbox Data Sheet.” Mathworks Products. Last Modified: 2012. http://www.mathworks.com/products/datasheets/pdf/global-optimization-toolbox.pdf.

[39] Mathworks, Inc. “Simulink Design Optimization Toolbox Data Sheet.” Mathworks Products. Last Modified: 2014. http://www.mathworks.com/products/datasheets/pdf/simulink-design-optimization.pdf.

[40] Otto, Chris, Aleksandar Milenkovic, Corey Sanders, and Emil Jovanov. "System architecture of a wireless body area sensor network for ubiquitous health monitoring." *Journal of Mobile Multimedia* 1, no. 4 (2006): 307-326.

[41] Mathworks, Inc. “How Pattern Search Polling Works.” Matlab & Simulink 2014b Documentation. Last accessed: September 2014. http://www.mathworks.com/help/gads/how-pattern-search-polling-works.html.

[42] Rouleau, Guy. “Using Discrete Data as an input to your Simulink model.” *Guy and Seth of Simulink* (blog). http://blogs.mathworks.com/seth/2012/02/09/using-discrete-data-as-an-input-to-your-simulink-model/.

[43] Mathworks, Inc. “pchip: Piecewise Cubic Hermite Interpolating Polynomial.” Matlab & Simulink 2014a Documentation. Last accessed: September 2014. http://www.mathworks.com/help/matlab/ref/pchip.html.

[44] Gorlatova, Maria, Aya Wallwater, and Gil Zussman. "Networking low-power energy harvesting devices: Measurements and algorithms." *Mobile Computing, IEEE Transactions on* 12, no. 9 (2013): 1853-1865.

[45] Huang, Longbo, and Michael J. Neely. "Utility optimal scheduling in energy-harvesting networks." *IEEE/ACM Transactions on Networking (TON)* 21, no. 4 (2013): 1117-1130.

[46] Wang, Qingsi, and Mingyan Liu. "When simplicity meets optimality: Efficient transmission power control with stochastic energy harvesting." In *INFOCOM, 2013 Proceedings IEEE*, pp. 580-584. IEEE, 2013.

[47] Kanjilal, Partha Pratim. *Adaptive prediction and predictive control*. No. 52. IET, 1995.

[48] Sarpeshkar, Rahul. "Ultra-Low Power Bioelectronics." Cambridge Uniersity Press 2010.

[49] Mitcheson, Paul D., Elizabeth K. Reilly, T. Toh, Paul K. Wright, and Eric M. Yeatman. "Performance limits of the three MEMS inertial energy generator transduction types." *Journal of Micromechanics and Microengineering* 17, no. 9 (2007): S211.

[50] Ventura, Joan, and Kaushik Chowdhury. "Markov modeling of energy harvesting body sensor networks." In *Personal Indoor and Mobile Radio Communications (PIMRC), 2011 IEEE 22nd International Symposium on*, pp. 2168-2172. IEEE, 2011.

[51] Baldus, Heribert, Karin Klabunde, and Guido Muesch. "Reliable set-up of medical body-sensor networks." In *Wireless Sensor Networks*, pp. 353-363. Springer Berlin Heidelberg, 2004.

[52] Chu, Maurice, Sanjoy Mitter, and Feng Zhao. "An information architecture for distributed inference on ad hoc sensor networks." In *PROCEEDINGS OF THE ANNUAL ALLERTON CONFERENCE ON COMMUNICATION CONTROL AND COMPUTING*, vol. 41, no. 1, pp. 90-99. The University; 1998, 2003.

[53] Milenković, Aleksandar, Chris Otto, and Emil Jovanov. "Wireless sensor networks for personal health monitoring: Issues and an implementation." *Computer communications* 29, no. 13 (2006): 2521-2533.

[54] Yarkony, N., K. Sayrafian, and A. Possolo, "Energy Harvesting from the Human Leg Motion", In 8th International Conference on Pervasive Computing Technologies for Healthcare, Oldenburg, Germany, May 20-23, 2014.

[55] Mathworks, Inc. “Improving Simulink Design Optimization Performance Using Parallel Computing.” Matlab & Simulink 2014b Documentation. Last accessed: September 2014.
http://www.mathworks.com/company/newsletters/articles/improving-simulink-design-optimization-performance-using-parallel-computing.html.