WPI

Functional Analysis of an Ankle-Foot Orthosis: The Intrepid Dynamic Exoskeletal Orthosis (IDEO)

Submitted by:
Andrew Lucy
Worcester Polytechnic Institute '19
Mechanical Engineering Major

A Major Qualifying Project Report Submitted to the Faculty of Worcester Polytechnic Institute in partial fulfillment of the requirements for the Degree of Bachelor of Science in Mechanical Engineering

April 25, 2019

Submitted by:

Andrew Lucy

WPI Class of 2019

Mechanical Engineering Major

Danvers, MA

Advisor:

Holly Ault, Ph.D.

Associate Professor, Mechanical Engineering

Director of Assistive Technology Resource Center

This report represents work of WPI undergraduate students submitted to the faculty as evidence of a degree requirement. WPI routinely publishes these reports on its website without editorial or peer review. For more information about the projects program at WPI, see http://www.wpi.edu/Academics/Projects

Abstract

Lower limb injuries that impact a patient's ability to walk can also lead to pain in the knees, back, and hips. This project investigated the functionality of an ankle-foot orthosis known as the Intrepid Dynamic Exoskeletal Orthosis (IDEO) that attempts to salvage injured limbs using basic principles of biomechanics. With help from the designer of the device, a patient that uses the device, and many others, the team studied the function of the IDEO in order to model its ability to transfer energy to the affected limb and restore normal gait function to the patient. We found that the IDEO stores energy while the patient walks using favorable material properties to replace the function of injured parts of the body. In addition, the device provides substantial support at the knee during the stance phase of the gait cycle. Further development in the modelling of joint reactions should be explored to assist in the evolution of similar medical devices.

Acknowledgments

The team would especially like to thank Charlene van Cott for her willingness to participate in this study. She not only supplied her personal medical device, but also supplied important information regarding its use as well.

We would also like to thank Ryan Blanck, who was an especially important resource who provided insight into the intended design of the device and all relevant components of it.

Special thanks to Professor Holly Ault, without whom this project would have not been possible. Her continued support throughout this project was critical to its success, as her comments and suggestions were important factors to the progress made by the team.

Also, the team would like to thank Professor Karen Troy, Professor Tiffiny Butler, and Lisa Wall for their constant support and assistance. All of these individuals assisted in the successful development of experimental procedures and accurate modelling related to this project.

Table of Contents

Table of Contents

Table of Contents

List of Figures

List of Figures

List of Tables

Motivation

Limb Salvage

Today, some patients and veterans with injuries and diseases that affect their lower body and impede their normal walking gait would rather amputate their injured limbs than try to rehabilitate them (Van Cott, 2018). The injuries that affect lower limb function and the ability to walk include ankle fusions, partial-foot amputations, fractures, tarsal coalitions and other lower extremity dysfunctions (Hanger Clinic, 2018). Patients with these types of injuries are typically incapable of walking normally, and therefore experience pain in other parts of the body, such as the hips, knees, and back (Intermountain Healthcare, 2018). In addition, patients with lower leg trauma experience atrophy in other muscles critical for healthy gait, and consequently require physical therapy to regain strength in these muscles (Stride Strong, 2019).

Specifically, a common lower limb injury known as drop foot affects a patient's ability to walk. This disability results in an inability to lift the foot off of the ground. A patient who exhibits drop foot will drag their toes on the ground when attempting to walk. The muscles of the foot that would lift the foot are deficient due to either nerve damage, spinal cord injury, or direct trauma to the associated muscles, tendons or ligaments. These factors associated with drop foot inhibit the motion required for healthy gait (WebMD, 2019). This means that when walking, a patient with drop foot will experience more pain in adjacent parts of the body as described above.

Patients with injuries such as drop foot have few good options to treat lower limb injuries. Most people cannot afford to or would prefer not to be confined to a wheelchair. Use of prosthetic limbs may require invasive surgeries and can be expensive. The use of orthoses in limb salvage allows the user to retain their lower limbs and correct any abnormalities in their movements over time. For permanently damaged patients, typical ankle/foot orthoses allow for more normal movements (Hanger Clinic, 2019).

Anatomy & Physiology of Targeted Areas

The IDEO replaces the function of injured lower body parts. To understand the functions that it replaces, we must first understand the normal function of these body segments The full anatomy targeted by the IDEO is complex, but the basic functions of the segments within the ankle, foot, and knee are fundamentally easy to understand.

` The following graphic highlights the axes associated with the ankle and the different directions it can move. When the angle of the joint between the tibia and foot increases, the joint is in dorsiflexion. The opposite of this motion is known as plantar flexion, during which the angle between segments increases (Samuel, 2019). These types of motion, as well as a visual representation of proximal and distal motion are shown in Figure 1.

Figure 1: Planes of motion relative to the foot. (Samuel, 2019)

There are 26 bones in the human foot; nineteen of those are related to the middle foot and toes. The remaining seven bones comprise the ankle and hind foot and are held together by a variety of ligaments.

The major bones in the foot are the talus (2) and calcaneus (1) bones, which connect the bones of the leg to the heel. These are known as the tarsal bones. The metatarsal bones (3-7) support and comprise the arch of the foot, while the phalanges (13-26) extend to the toes (Sports Podiatry Resource Inc., 2019). This layout is shown in Figure 2.

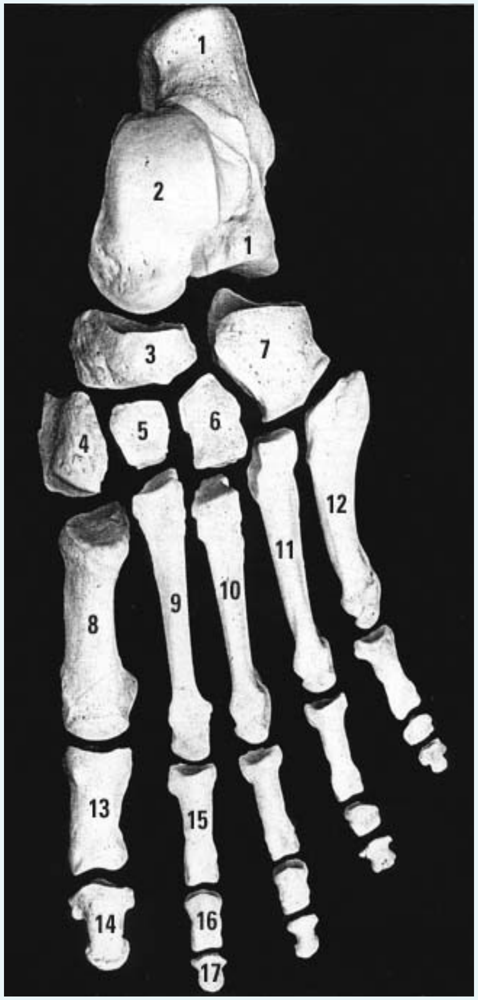


Figure 2: Bones of the human foot (numbered). (Sports Podiatry Resource Inc., 2019)

Two basic concepts in ankle and foot motion are pronation and supination. These are both induced by changes in weight distribution during gait. Different rotations about multiple joints in the ankle and foot occur during these changes in distribution. The tibia rotates internally or externally, while the talus and calcaneus move opposite to each other during either pronation or supination.

Pronation is characterized by plantar flexion at the talus bone, internal rotation of the tibia, and eversion of the calcaneus bone. This occurs as forces on the foot become displaced over its length, causing slight elongation and flattening of the segment. Conversely, supination involves dorsiflexion at the talus bone, external rotation of the tibia, and inversion of the calcaneus bone. When in supination, weight on the foot is distributed laterally to the outside portion of the foot (Souza et. al., 2010). These inverse mechanical operations are diagrammed in Figure 3.

Movements of bones are caused by generation of forces and moments by muscles. Extension of the knee is caused by a contraction of the quadriceps muscles, while knee flexion is caused by a contraction of the hamstrings. Flexion of the hip is caused by the contraction of the rectus femoris, while extension is due to contractions of the adductor and gluteus muscles. A large variety of muscles within the foot and lower leg affect plantar flexion of the foot and ankle. The full gait cycle involves all of these types of movements, as energy is transferred between segments (Loudon et al., 2008).

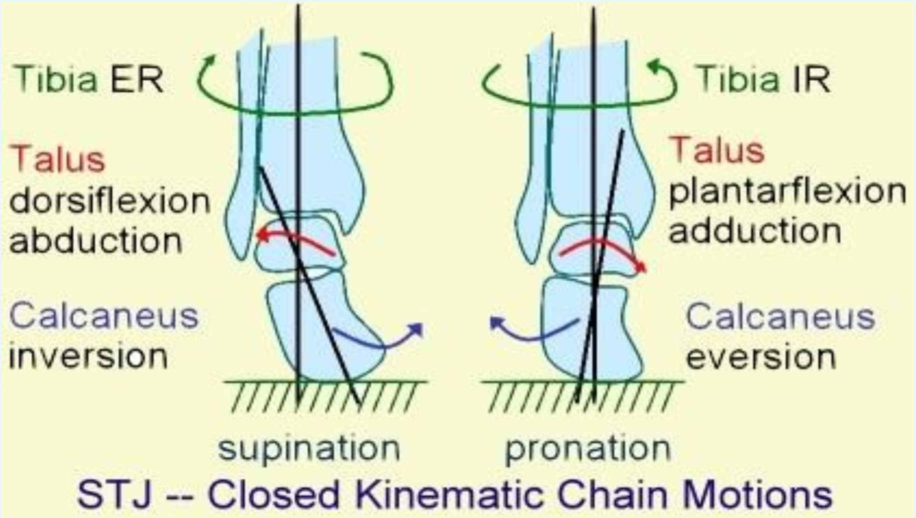
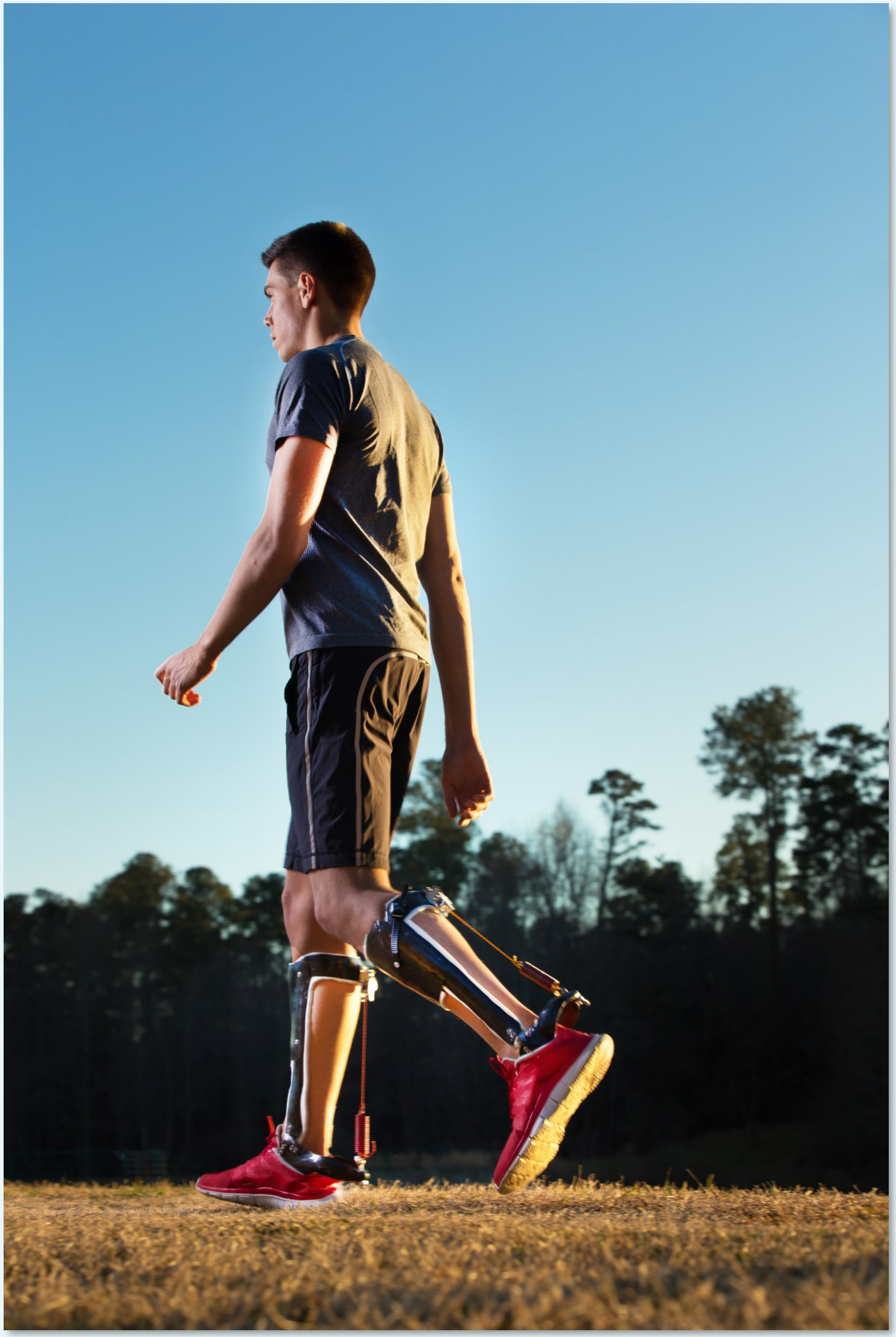


Figure 3: Planes of motion relative to the foot and ankle. (Souza et. al., 2010)



Aspects of Gait

The Gait Cycle

Since the gait of an individual is entirely specific to that person, it is important to realize that normal gait is relative, but we can identify the difference between healthy and unhealthy gait. The normal walking gait of a human includes two distinct phases of leg movement: the stance phase and the swing phase.

One full gait cycle begins and ends with contact of one heel to the ground, as shown in Figure 4. After heel contact, weight is focused on the engaged leg as the foot begins to flatten during pronation. At this point, the entire weight of the body is supported by the muscles, joints, and ligaments in the working leg.

Heel strike is the initial stage of the stance phase. This is a short period of the gait cycle which begins the moment the heel touches the ground. The upper leg is flexed at around 30° from vertical in the hip as the knee is fully extended and in line with the upper leg.

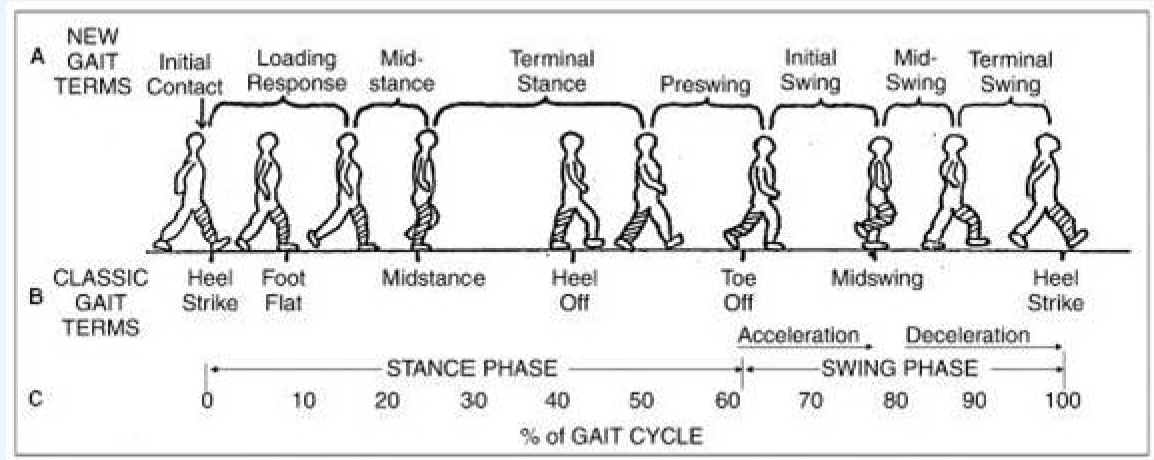


Figure 4: Phases of the gait cycle. (Physiopedia, 2019)

The ankle moves from a neutral - normally supinated 5° - position into plantar flexion. After this, knee flexion of 5° begins and increases, acting as a shock absorber.

In foot flat, the body absorbs impact in the foot by rolling into pronation. The hip moves slowly into extension, as the femur crosses past the frontal plane. The knee flexes to 15° to 20°. Ankle plantar flexion increases to 10-15° as the foot rolls further into midstance.

In midstance the hip moves from 10° of flexion to extension. The knee fully flexes and then begins to extend. While the weight is distributed away from the body, the ankle becomes supinated and dorsiflexed to 5°. During this phase, the body is supported by the single leg in contact with the ground, since the other foot and leg is entering the swing phase. After this period of force absorption, the body begins to propel itself forward.

The heel off phase begins once the heel leaves the floor. In this phase, the body weight is divided over the heads of the metatarsal bones. Here can we see 10-15° of extension in the hip joint, which then goes into flexion. The knee becomes flexed 0-5° and the ankle undergoes supination and plantar flexes.

During the toe-off phase, the hip becomes less extended and the toes leave the ground. The knee becomes flexed 35-40° and plantar flexion of the ankle increases to 20°.

In the early swing phase the hip extends to 10° and then flexes due to 20° with lateral rotation. The knee flexes to 40-60°, and the ankle goes from 20° of plantar flexion to dorsiflexion, to end in a neutral position

In the mid swing phase the hip flexes to 30° and the ankle becomes dorsiflexed. The knee flexes 60° but then extends approximately 30°.

The late swing phase begins with hip flexion of 25-30°, a locked extension of the knee and a neutral position of the ankle (Loudon et al., 2008) (Shultz et al., 2005).

Typical Forces Experienced During Gait

The following graphs, Figures 5-7, show typical values for ground reaction forces in three dimensions (Vaughan et al., 1999). Note that the typical stance phase consumes approximately 60% of the gait cycle, then the subject's gait enters the swing phase. For the sake of this paper, the X direction will denote the anterior-posterior (AP) direction. The Y direction will correspond to the medial-lateral (ML) direction, and the Z direction will be in the vertical (V) direction.

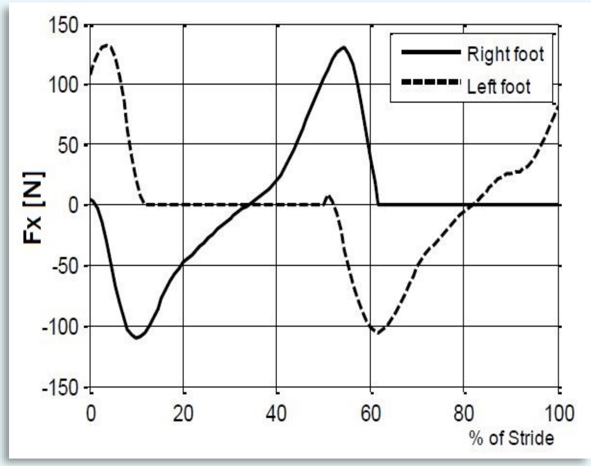


Figure 5: Typical Ground Reaction Force in X direction for both feet. (Vaughn et al., 1999)

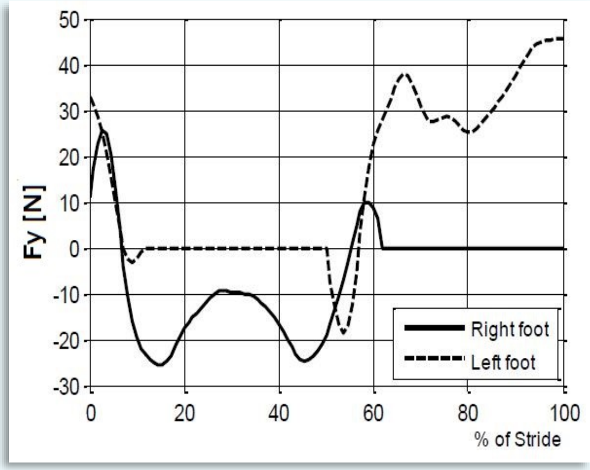


Figure 6: Typical Ground Reaction Force in Y direction for both feet. (Vaughn et al., 1999)

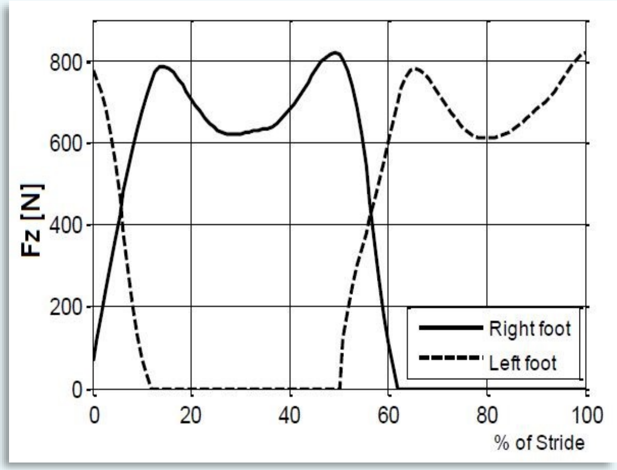


Figure 7: Typical Ground Reaction Force in Z direction for both feet. (Vaughn et al., 1999)

During one full gait cycle, the vertical ground reaction force has two distinct peaks. These points take place when: 1) the leg opposite to the plant leg begins to touch the ground again after swinging and 2) directly after the opposite toe leaves the ground and enters the swing phase. Typically, the peaks in this force will be greater than the body weight of the person, as muscle force from the legs and trunk will also act on the ground in addition to weight of the entire body. The AP ground reaction force will typically increase as the subject's foot rolls through stance phase, as the applied force shifts from front to back. The ML ground reaction force is the smallest in magnitude, but follows an expected pattern as the foot and ankle roll from supination to pronation. The two troughs visible in this medial-lateral plot correspond to the same peaks seen in the vertical ground reaction force plots (Winter, 2009).

Energy Transfer & Power Generation

The healthy human body always moves together. Muscles are the only element of the body within individual body segments that can produce work through contractions, as adjacent body segments absorb the resulting energy to complete movements and transfer energy as well.

Loads absorbed by the body create moments on other body segments, and therefore require the use of muscles to distribute them. At different points in the body, adjacent segments often produce opposite work. The total energy and the exchange of energy within segments is the sum of the potential, kinetic, and rotational energy, shown in the equation below.

$$E_{tot,\ seg} = mgh + mv^2/2 + Iw^2/2$$

As the body replenishes its cells with oxygen, it allows the body to perform work and expend this energy in its segments. This overall model of energy flow is important because it takes into account metabolic energy, which is important in determining efficiency in movements. For our study, we care more about the individual movement of the segments.

The link segment model is one way to look at the different ways the body segments move. Each body segment has the capability to move adjacent ones depending on specific conditions. When analyzing the motion of the leg, it is much easier to use the link segment model since it clearly outlines the length of segments and points of connection. A comparison of the link segment model to the normal anatomical model is shown in Figure 8.

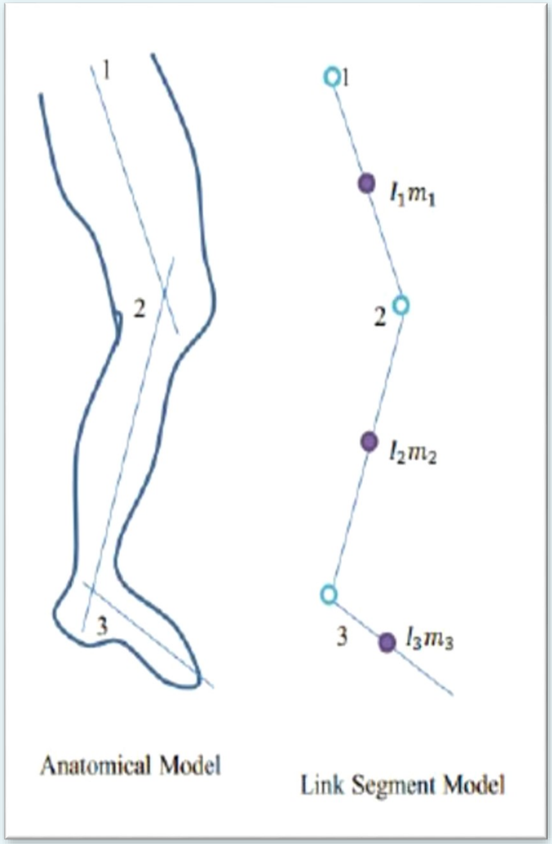


Figure 8: Body part descriptions in the Anatomical Model versus the Link Segment Model (Nisan Amirudin et al., 2014)

The only source within the human body capable of mechanical energy generation is the muscles. This mechanical energy turns into mechanical power via contractions once time elapses. At joints between body segments, mechanical power is defined by the product of net moment of a body segment generated by a muscle and the angular velocity of the same body segment, shown below. This equation allows us to quantify the power at specific joints over time.

$$P_m = M_j * w_j$$

Another equation will allow us to calculate instantaneous power at these joints. This equation is the dot product of force acting on the joint as a part of a segment and velocity of the joint. This means that when a reaction force, F_j acts on the end of a segment, the point directly at the joint will be moving in one direction with a specified velocity, V_j . The angle between the resultant force on a segment and corresponding velocity is defined as Θ_1. This equation is shown below.

$$P = FV\cos\theta = FxVx + FyVy$$

Mechanical energy and power generation are caused by concentric contractions of muscles in body segments, and absorption is caused by eccentric contraction of muscles. The specific amount, type, and direction of power generated or absorbed at each segment can be calculated according to these variables (Winter, 2009). A full description of power transfers is shown in Table 1.

Aspects of Gait Energy Transfer &

Table 1: Mechanical energy generation, absorption and transfer amongst segments during walking. (Winter et al., 1980)

Description of movement	Type of contraction	Directions of segmental ang. velocities	Muscle function	Amount, type and direction of power
Both segments rotating in opposite directions (a) joint angle decreasing	Concentric	ω_1 M ω_2	Mechanical energy generation	$M\omega_1$ generated to segment 1 $M\omega_2$ generated to segment 2.
(b) joint angle increasing	Eccentric	ω_1 M ω_2	Mechanical energy absorption	$M\omega_1$ absorbed from segment 1. $M\omega_2$ absorbed from segment 2.
Both segments rotating in same direction (a) joint angle decreasing (e.g. $\omega_1 > \omega_2$)	Concentric	ω_1 M ω_2	Mechanical energy generation and transfer	$M(\omega_1 - \omega_2)$ generated to segment 1. $M\omega_2$ transferred to segment 1 from 2.
(b) joint angle increasing (e.g. $\omega_2 > \omega_1$)	Eccentric	ω_1 M ω_2	Mechanical energy absorption and transfer	$M(\omega_2 - \omega_1)$ absorbed from segment 2. $M\omega_1$ transferred to segment 1 from 2.
(c) joint angle constant ($\omega_1 = \omega_2$)	Isometric (dynamic)	ω_1 M ω_2	Mechanical energy transfer	$M\omega_2$ transferred from segment 2 to 1.
One segment fixed (e.g. segment 1.) (a) joint angle decreasing ($\omega_1 = 0$, $\omega_2 > 0$)	Concentric	M ω_2	Mechanical energy generation	$M\omega_2$ generated to segment 2
(b) joint angle increasing ($\omega_1 = 0$, $\omega_2 > 0$)	Eccentric	M ω_2	Mechanical energy absorption	$M\omega_2$ absorbed from segment 2.
(c) joint angle constant ($\omega_1 = \omega_2 = 0$)	Isometric (static)	M	No mechanical energy function	Zero.

Anthropometric Data

Since it would be extremely difficult to physically weigh individual body segments, biomechanics uses anthropometry to define standardized proportions of segments. Many different lengths and masses can be obtained using these tables. Masses of all body segments can be obtained by only knowing the total body mass of the individual. Similarly, lengths of proximal and distal ends of segments can be calculated once we know the length of the segment as a whole. The following Anthropometric table in Table 2 provides guidelines for the anatomical length, segment weight, segment center of mass locations, and segment center of gyration locations for all body segments.

Table 2: Anthropometric Data. (Winter, 2009)

Table 3.1 Anthropometric Data

Segment	Definition	Segment Weight/ Total Body Weight	Center of Mass/ Segment Length		Radius of Gyration/ Segment Length			Density
			Proximal	Distal	C of G	Proximal	Distal	
Hand	Wrist axis/knuckle II middle finger	0.006 M	0.506	0.494 P	0.297	0.587	0.577 M	1.16
Forearm	Elbow axis/ulnar styloid	0.016 M	0.430	0.570 P	0.303	0.526	0.647 M	1.13
Upper arm	Glenohumeral axis/elbow axis	0.028 M	0.436	0.564 P	0.322	0.542	0.645 M	1.07
Forearm and hand	Elbow axis/ulnar styloid	0.022 M	0.682	0.318 P	0.468	0.827	0.565 P	1.14
Total arm	Glenohumeral joint/ulnar styloid	0.050 M	0.530	0.470 P	0.368	0.645	0.596 P	1.11
Foot	Lateral malleolus/head metatarsal II	0.0145 M	0.50	0.50 P	0.475	0.690	0.690 P	1.10
Leg	Femoral condyles/medial malleolus	0.0465 M	0.433	0.567 P	0.302	0.528	0.643 M	1.09
Thigh	Greater trochanter/femoral condyles	0.100 M	0.433	0.567 P	0.323	0.540	0.653 M	1.05
Foot and leg	Femoral condyles/medial malleolus	0.061 M	0.606	0.394 P	0.416	0.735	0.572 P	1.09
Total leg	Greater trochanter/medial malleolus	0.161 M	0.447	0.553 P	0.326	0.560	0.650 P	1.06
Head and neck	C7-T1 and 1st rib/ear canal	0.081 M	1.000	— PC	0.495	1.116	— PC	1.11
Shoulder mass	Sternoclavicular joint/ glenohumeral axis	—	0.712	0.288	—	—	—	1.04
Thorax	C7-T1/T12-L1 and diaphragm*	0.216 PC	0.82	0.18	—	—	—	0.92
Abdomen	T12-L1/L4-L5*	0.139 LC	0.44	0.56	—	—	—	—
Pelvis	L4-L5/greater trochanter*	0.142 LC	0.105	0.895	—	—	—	—
Thorax and abdomen	C7-T1/L4-L5*	0.355 LC	0.63	0.37	—	—	—	—
Abdomen and pelvis	T12-L1/greater trochanter*	0.281 PC	0.27	0.73	—	—	—	1.01
Trunk	Greater trochanter/ glenohumeral joint*	0.497 M	0.50	0.50	—	—	—	1.03
Trunk head neck	Greater trochanter/ glenohumeral joint*	0.578 MC	0.66	0.34 P	0.503	0.830	0.607 M	—
HAT	Greater trochanter/ glenohumeral joint*	0.678 MC	0.626	0.374 PC	0.496	0.798	0.621 PC	—
HAT	Greater trochanter/mid rib	0.678	1.142	—	0.903	1.456	—	—

*NOTE: These segments are presented relative to the length between the greater trochanter and the glenohumeral joint.

Source Codes: M, Dempster via Miller and Nelson; *Biomechanics of Sport*, Lea and Febiger, Philadelphia, 1973. P, Dempster via Plagenhoef; *Patterns of Human Motion*, Prentice- Hall, Inc. Englewood Cliffs, N.J., 1971. L, Dempster via Plagenhoef from living subjects; *Patterns of Human Motion*, Prentice-Hall, Inc., Englewood Cliffs, N.J., 1971. C, Calculated.

The IDEO

Orthosis Basic Function

An orthosis is a device attached or applied to the external surface of the body to improve function, restrict or enforce motion, or support a body segment (Chung, 2008). The Intrepid Dynamic Exoskeletal Orthosis (IDEO) is a biomedical device designed to aid patients that have sustained trauma or injury that impedes their normal walking gait. The IDEO utilizes carbon fiber to conform to the shape of the patient's injured leg and disperse stored energy in the materials to allow the user to walk. The main function of the device involves absorbing normal body forces at the knee cuff and using the subsequent flexion and extension in the struts to support the foot and ankle as the orthosis carries these body segments. This process is outlined in Figure 9.

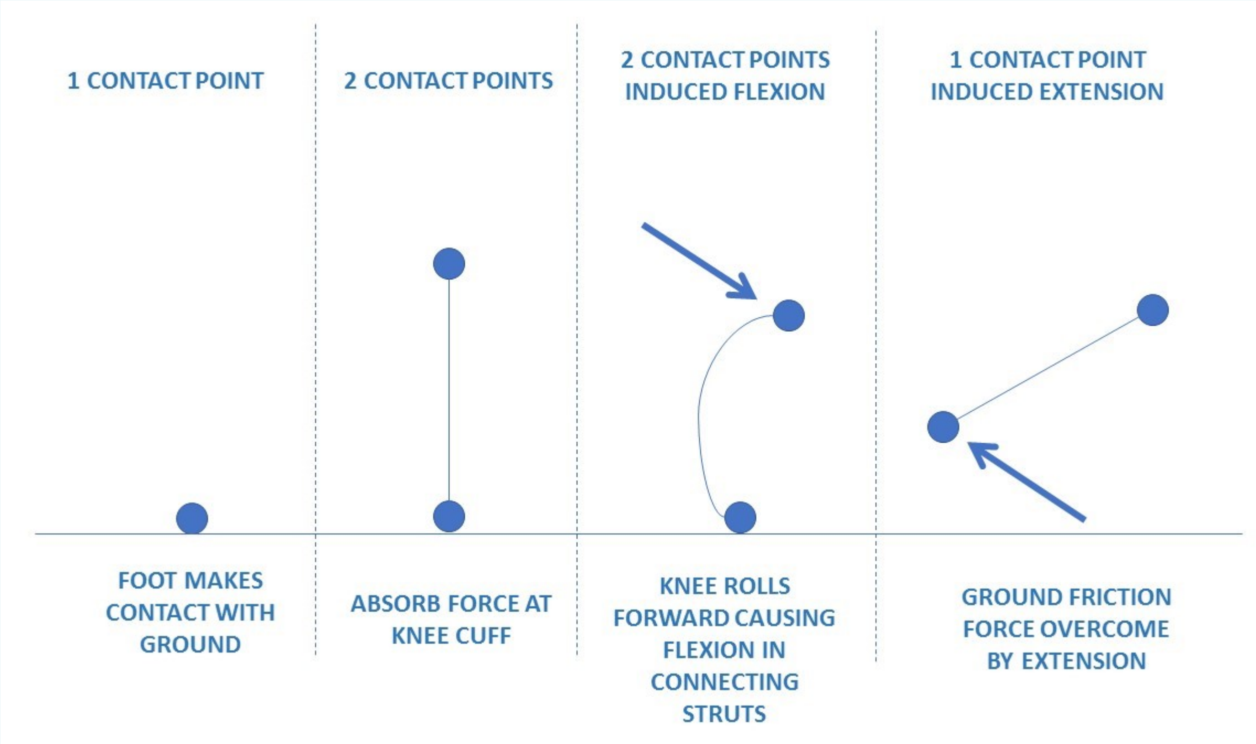


Figure 9: Flexion and extension of IDEO struts, simplified

IDEO Components

The IDEO is a cutting-edge device that combines our understanding of the human body with techniques that utilize the properties of materials. But the device itself is not autonomous, as the user drives the capabilities of the device. Patients that rehab their core glutes and hamstrings will be able to recover and return to normal movement faster (Blanck, 2018).

The IDEO is comprised of three major components as shown in Figure 10. Component 1 is the knee cuff, which can be tightened with velcro straps so that the user can apply force just below the knee. Component 2 consists of the two long struts along the posterior of the device. These struts are made of unidirectional carbon fiber that comprise the only active portion of the device. Component 3 is the foot bed, which cradles the foot and allows for contact at the ground without direct force applied to the foot.

Figure 10: The IDEO and its components, numbered. (Hanger Clinic, 2018)

IDEO Design Goals

For the user, the IDEO can salvage a completely flaccid foot and redistribute the weight of the body elsewhere within the orthosis. For patients with drop foot, this device can be extremely effective to relieve pain associated with walking. The same principle applies to the calf and lower leg and foot muscles, as the device is capable of replacing their function. Experienced and rehabilitated users of the IDEO are able to walk, run, jump, and move laterally with good control.

The dynamic struts implemented in the design of the IDEO are strong enough to sustain the weight of the body as well as forces caused by muscles, all while sustaining normal gait. A bending moment is created about the heel once it makes contact with the ground, resulting in a flexure of the material as the foot and leg roll into midstance. This flexion is directly translated to the energy that an injured leg is not able to provide.

All of the active portions of the device are made of carbon fiber. Two dynamic carbon fiber struts along the rear of the orthosis constitute the main segment of the device. Two opposing forces, at the knee cuff and foot, cause deflection within the orthosis. The struts are loaded during the heel strike to toe off phases of the gait cycle. As the toe off phase ends and the swing phase begins, the material is returning to its original shape (Blanck, 2018).

Ryan Blanck is the head prosthetist responsible for the design of the IDEO. He is the clinic manager at the Hanger Clinic in Gig Harbor, Washington, where he is currently leading the ExoSym Program. He evaluates patients based on their individual injury and determines how the IDEO/ExoSym Program could benefit them. Each patient that qualifies for the program undergoes a body optimization program that involves extensive therapy and device fitting procedures. One example of advice given to an IDEO candidate is to emphasize the roll from the Heel Strike phase of the gait cycle to the Toe Off phase. This will allow for a more even distribution of impact force along the entire foot as the user focuses on good running and walking form.

The ExoSym is a device that uses the same biomechanical principles as the IDEO, but with an improved design based around the natural symmetry of the body. Some IDEO users did not achieve full bodily symmetry after extensive use and optimization the device. A new version of the device has implemented a strap at the high ankle, as running and movement causes the foot to come out of the custom molded carbon fiber exoskeleton. This strap holds the shin in place while the rest of the leg moves normally with the IDEO, as seen in Figure 11. Since the IDEO is the basis for the ExoSym, we will study the motion and function of the IDEO to determine the most relevant aspects of the design.

Figure 11: An ExoSym user running with a shin strap. (Hanger Clinic, 2018)

Properties of Materials

The chemical composition and structure of each material determine its mechanical properties, including strength, modulus, ductility, and compliance. These properties outline the mechanical behavior of each material when loaded under different stresses and strains.

Elastic modulus of a material is defined by the ratio of the force exerted upon a substance or body to the resultant deformation. Mathematically, this is represented by applied force in Newtons divided by the resultant strain in elongation percentage, during the elastic portion of loading.

Materials that exhibit strong mechanical properties are capable of sustaining heavy loads without permanently deforming (Maggs, 2012). The behavior of the material depends on the yield point, geometry, and external loading pattern.

The toughness of a material is defined by the amount of energy that can be absorbed by a material before it experiences permanent deformation. This can be quantified as the area under the stress-strain curve at the yield point. Similarly, a material will store energy equal to the area under the curve at a specified stress or strain value in the entire elastic region of the stress-strain curve. We can also calculate strain energy from the loading conditions and properties of a specific material.

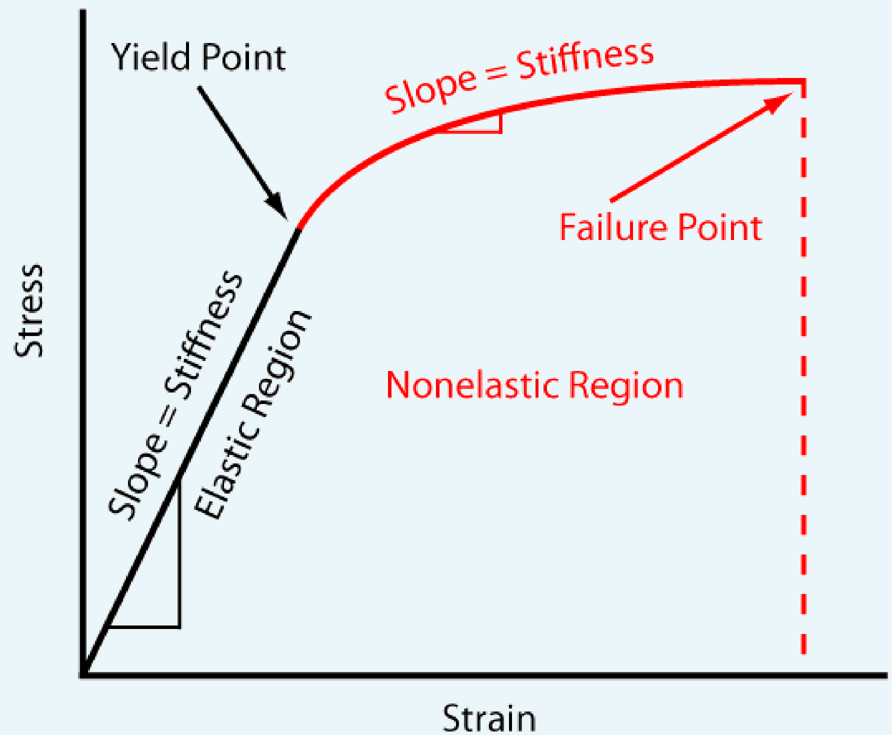


Figure 12: Typical Stress-Strain curve and relevant information. (University of Texas - Arlington, n.d.)

Project Goal

According to IDEO users and clinicians, the IDEO allows patients to walk more normally than without the device. This normal motion can be described by the motion of healthy body segments. Since walking is painful for patients with lower leg trauma, gathering motion data for injured segments without any aid would not be a preferred method to show the function of the device. We know that injured body segments will produce unnatural patterns in gait, but we would still like to demonstrate how the device works.

The goal of this project is to quantify the motion corresponding to an entirely healthy set of body segments and show how the IDEO can allow for lower limb motion comparable to healthy lower limbs.

For this study, we studied the walking gait of an IDEO user. After being seriously injured in a tour overseas with the Army, she participated in Ryan Blanck's rehabilitation clinic to treat her injuries. She experienced lower leg trauma and exhibits the symptoms of drop foot. With the aid provided by the Hanger Clinic, she is now an active member of the community and uses the IDEO every day. Because of the sustained effectiveness of the IDEO, she is able to serve others as a firefighter, EMT, and police officer part time without being inhibited by her injury (Van Cott, 2018).

Identifying IDEO Function

To demonstrate the function of the IDEO, ground reaction force measurements were gathered on one IDEO user. Motion Sensing was used in conjunction with force plates to determine the reaction forces in the body experienced during gait. The use of force plates allows us to gather ground reaction force data. By consequently using Motion Sensing to gather displacement data, we are able to calculate the velocity and acceleration of each body segment. By com-

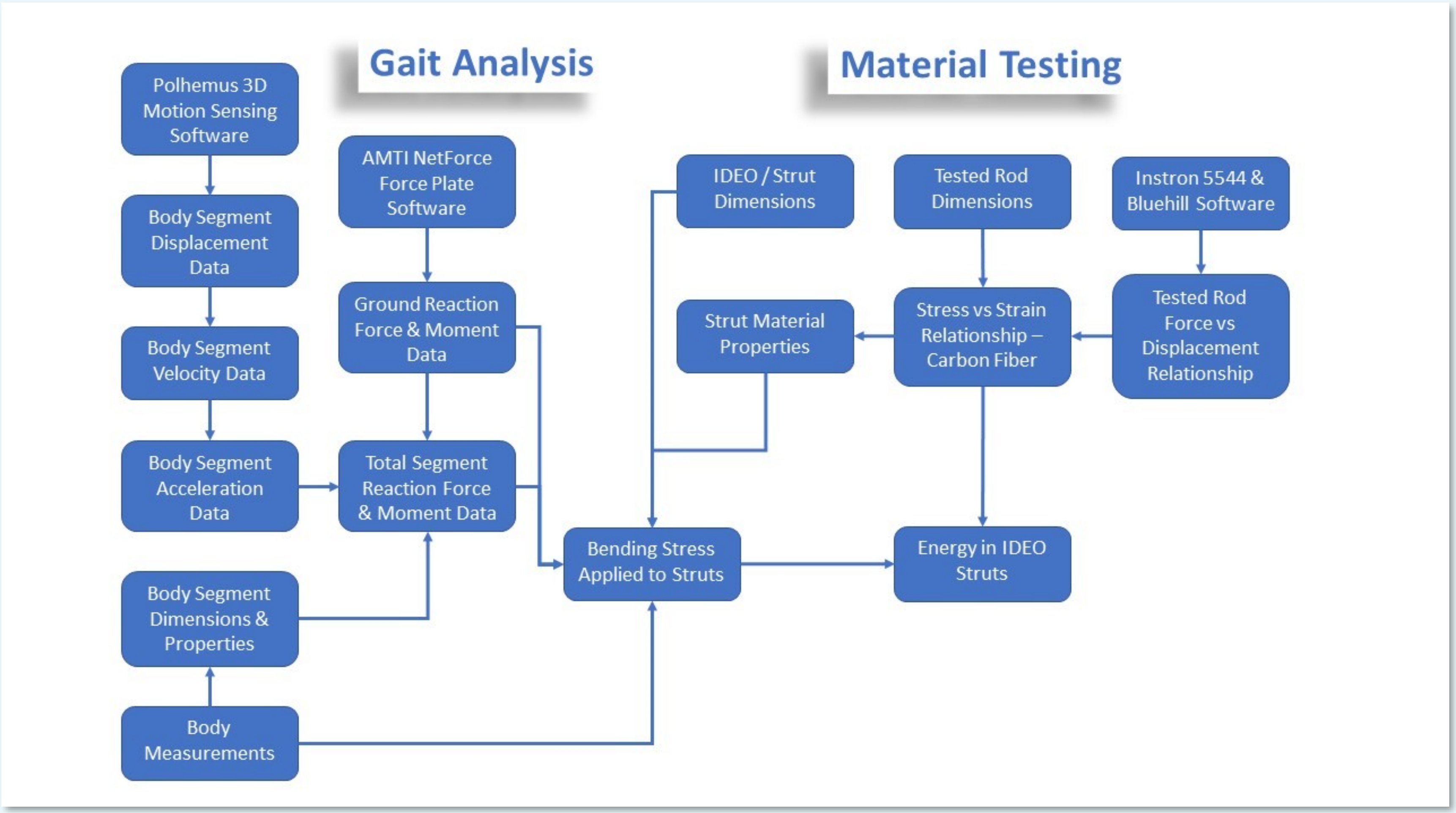


Figure 13: Methods of Identifying IDEO Function

bining the principles of kinetics and kinematics, we are able to use the external forces acting on a subject with the internal motion of segments to determine reaction forces in the body.

Material testing was performed on the active components of the IDEO to obtain the mechanical properties. The stress-strain relationship of the carbon fiber struts was obtained to relate it to bending stress experienced during gait. This allows us to quantify the stored energy in the device during use. The following graphic shows the flow of information and data utilized for this study.

Force Plate Data

The AMTI Force Plate can read forces and moments in three dimensions applied to the top surface. This allows us to create a Ground Reaction Force vector that can be factored into our force equations during the stance phase of the leg being measured. Since the stance phase involves Heel Strike to Toe Off, we can normalize our force data to become a function of this full phase. The AMTINetForce software sampling rate is set at 120 Hz.

3D Motion Sensing

The Polhemus G4 Motion Sensing system produces motion data at each sensor connected. These sensors are connected using sensor hubs that transmit data wirelessly to the PC. The displacement is relative to the Polhemus Source Box, which sets the origin of the global coordinate system.

When connected properly, the data for linear and Euler displacement at each sensor can be transmitted in real time. The data are measured in all three dimensions, with X, Y, and Z linear displacement and Euler angles relative to X, Y, and Z. The sampling rate for the Polhemus G4 system is 120 Hz and cannot be adjusted.

Gait Analysis

An analysis of the patient's gait was performed using both Motion Sensing and a Force Plate. We gathered data on the patient's healthy left limb as well as her injured right limb equipped with the IDEO. Five trials were performed for her injured leg, and four trials were performed for her healthy leg. Prior to testing, her relevant body segments were measured, and we found her total body mass to equal 91.62 kg. Masses of relevant body segments were obtained using the Anthropometric table described earlier.

The gait analysis experiment involved a runway created using three wooden platforms and a force plate. The first two wooden platforms were placed before the force plate, and the last platform was placed after the force plate. A more complete diagram of this setup is shown in Figure 14.

The Polhemus source box is positioned at the beginning of the runway along one side to create the origin of our global coordinate system. According to this Polhemus System, the +X direction is forward, down the runway created by the wooden platforms, the +Y direction is to the right of the leftmost edge of the platforms, and the +Z direction is above the source box. This creates a Left Handed Coordinate System, which is not typical, but still usable.

After the sensor hubs, force plate, and source box were all initialized, the sensors were placed on her body. The medial-lateral leg and foot widths were measured to obtain the Y coordinates relevant for the middle of the leg and foot. The leg segment distance was measured

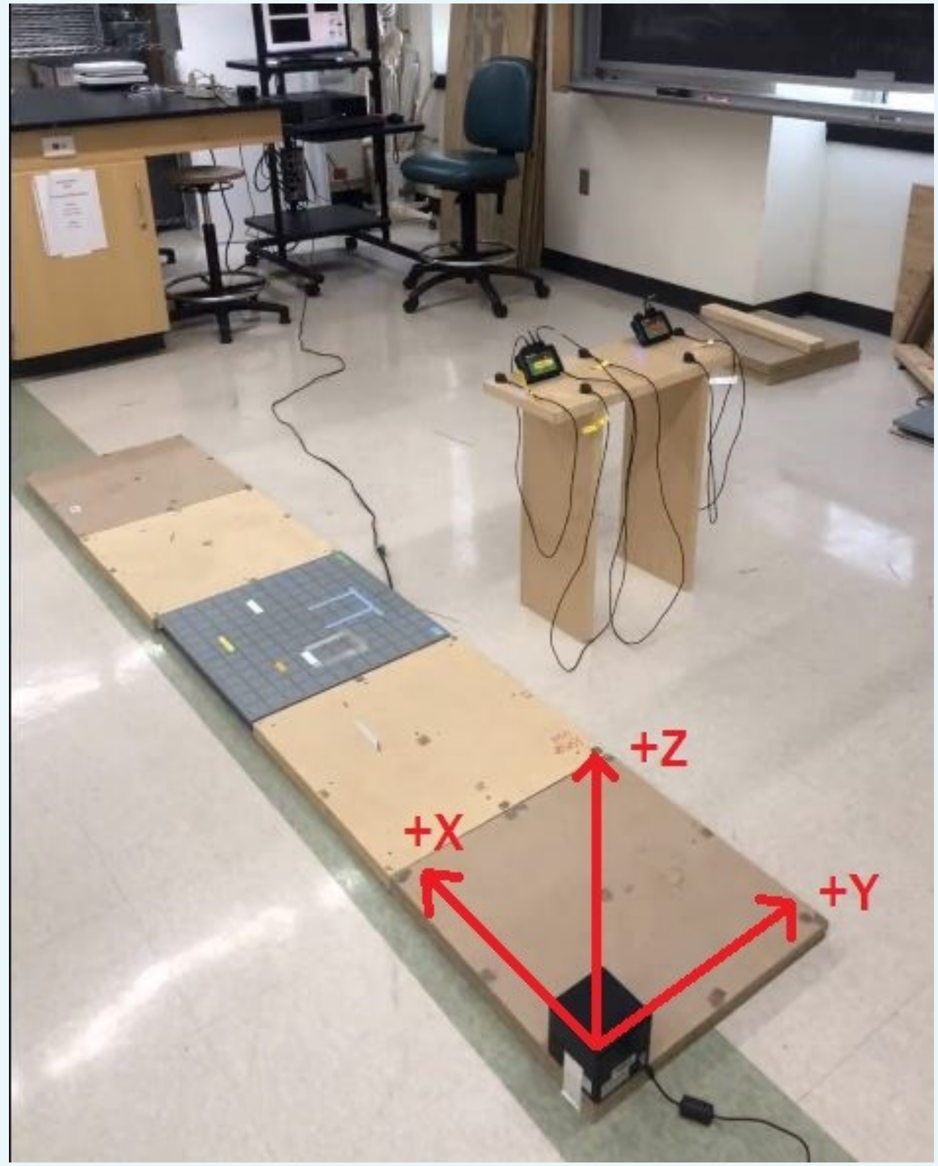


Figure 14: Gait Analysis full equipment setup, with labeled global coordinate system

from femoral condyles to medial malleolus to find total length of the lower leg. The foot segment distance was measured from lateral malleolus to head metatarsal II to obtain the length of the foot. These Anthropometric values allow for approximate values for proximal and distal lengths of segments from the center of mass. All sensor locations on the subject's legs can be visualized in Figures 15 and 16.

The first two locations on the following list were used to track the motion of the foot and leg. The last three sensors on this list create body segment axes on the foot and leg that have an angular orientation that we were able to track.

For Healthy (Left) Leg & Foot Calculations, sensors were placed at each of the following:

- the midpoint of the foot segment on the lateral side of the foot (COM of the foot)
- 43.3% of the way down the leg from the knee (0.433*leg length), on the lateral side of the leg (COM of the leg)
- the femoral condyle on the lateral side of the leg
- the heel of the foot on the lateral side
- the ends of the metatarsal bones in the foot on the lateral side

Since the IDEO combines the foot, ankle, and leg into one body segment, we can track this segment using one sensor located at the center of mass.

For Injured (Right) Leg/IDEO Calculations, sensors were placed at each of the following:

- 60.6% of the way down the leg from the knee (0.606*leg length), on the lateral side of the leg (COM of total segment)
- the femoral condyle on the lateral side of the leg
- the heel of the foot on the lateral side
- the ends of the metatarsal bones in the foot on the lateral side
- Below the lower end of the struts on the IDEO

After each sensor is placed correctly, the patient should be able to move unrestricted by connecting wires. The patient started by standing still on the edge of the first platform. The synchronized data collection began when the patient first began walking. The second step was centered on the force plate, and the patient came to a stop after stepping off the end of the final platform.

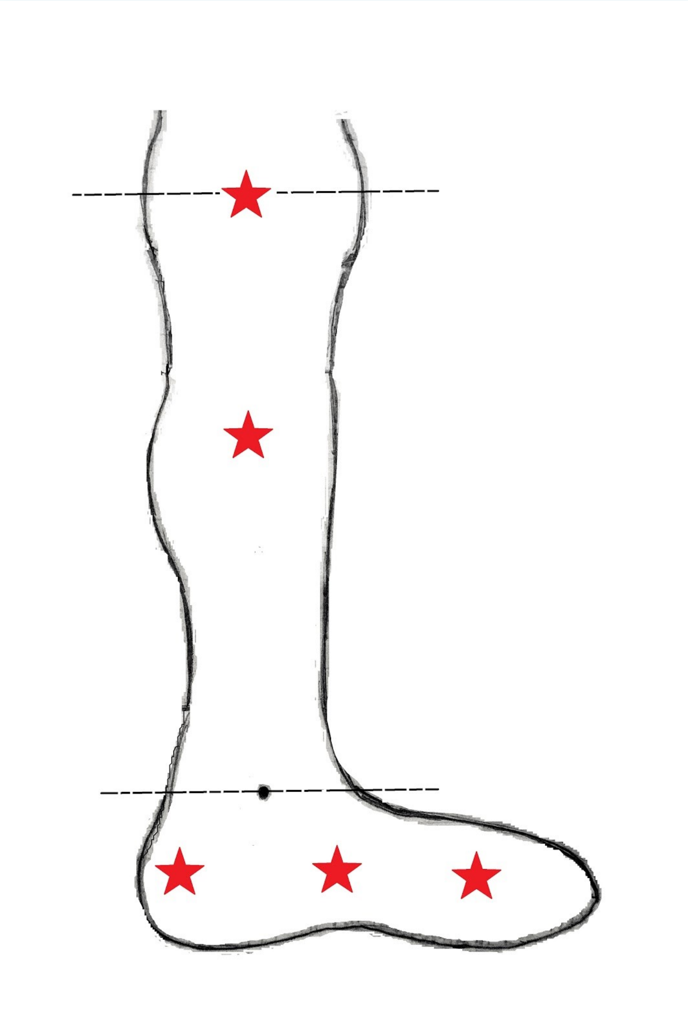


Figure 15: Sensor locations on subject's healthy leg

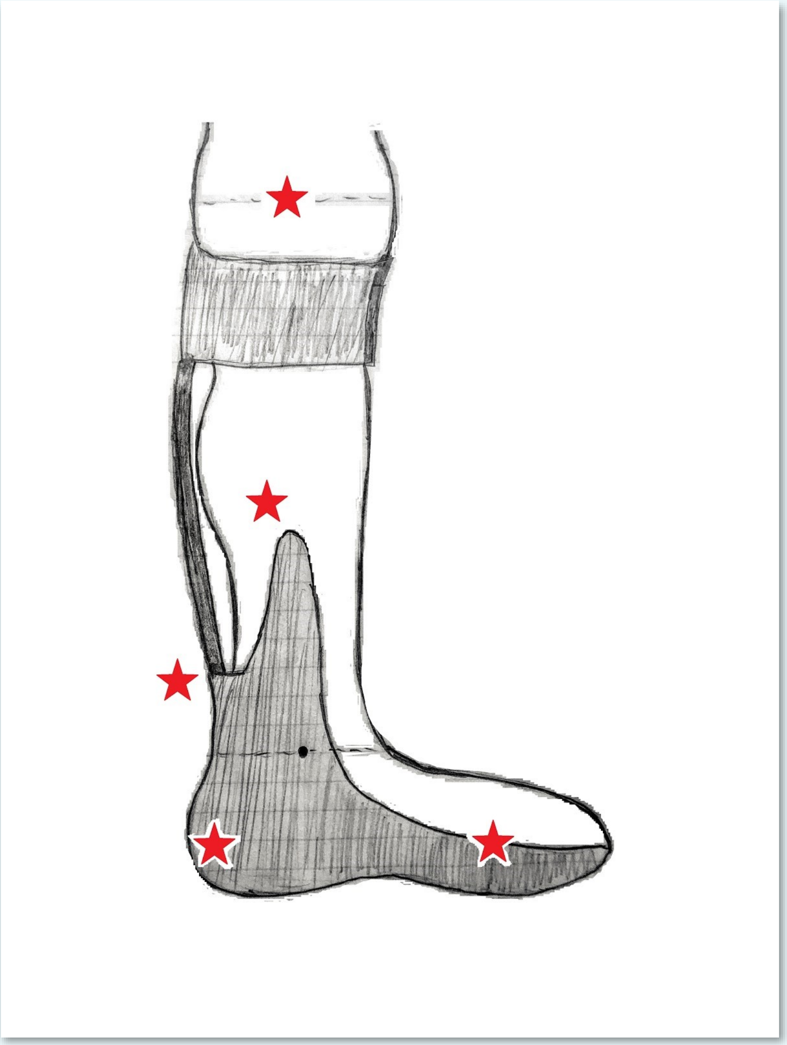


Figure 16: Sensor locations on subject's injured leg

Motion & Force Data Analysis

The AMTINetForce software produced both a .txt file and a .bsf file that were used to find relevant data. The .txt file contained the force and moment values over time in all three dimensions. BioAnalysis was used to determine the exact elapsed time when the Heel Strike phase begins and when the Toe Off phase ends, using data from the .bsf file. The Polhemus PiMgr software produced a .csv file that contained all of the linear and Euler displacement data along with the corresponding sample number.

MatLab was used to organize and manipulate all the data such that they could be used in force equations. The full MatLab scripts for Healthy Calculations, Injured Calculations and Calculations related to Energy are all available in Appendix A.

Once a full set of raw data was gathered for each trial, the raw data was truncated to only include samples that occurred when the subject's foot was in contact with the force plate. This truncated displacement data was filtered using a 5th order low-pass Butterworth filter to reduce high frequency noise. A Fast Fourier Transform (FFT) was performed on the truncated data to determine the relevant frequencies appropriate for our filter.

The displacement data during stance phase was obtained from the Polhemus sensors. This data was differentiated to determine the corresponding velocity and acceleration of each relevant body segment. The acceleration of these components is critical for the force

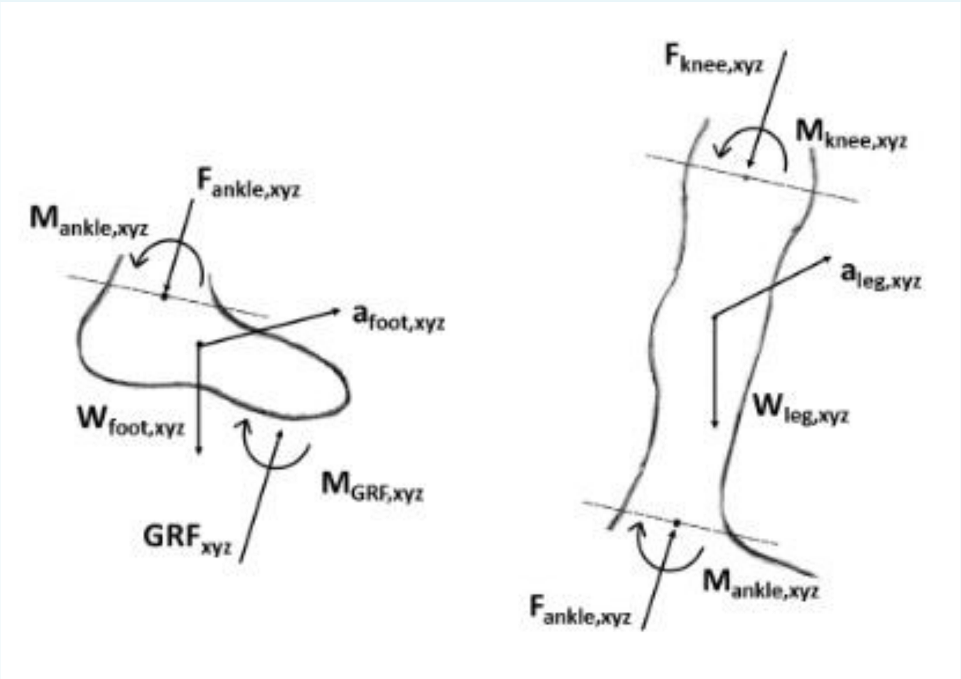


Figure 17: Free Body Diagrams of the Foot and Ankle during one step

calculations that we need to determine joint reaction forces. The following free body diagrams show the forces acting on body segments during gait. Figure 17 shows how forces are acting on healthy body segments, and Figure 18 shows how the same types of forces act on the user's injured foot and leg with the IDEO.

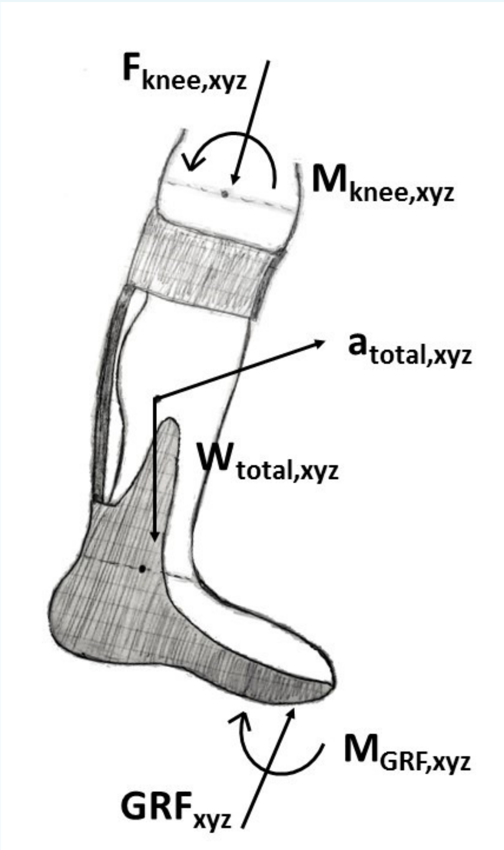


Figure 18: Free Body Diagram of the combined segments created by the IDEO

Because we know that the sum of the forces in a system equal mass times acceleration, the equations below show how these joint reaction forces were calculated using the ground reaction force, acceleration, and masses of relevant body segments. Note that for the injured leg calculations, the 'total' mass (m), Weight (W), and acceleration (a) take into account the IDEO, foot, and leg. This is because the ankle joint is replaced by the function of the IDEO, and therefore the IDEO creates one large body segment that absorbs the Ground Reaction Force (GRF). The forces along with acceleration are all in three dimensions to determine reaction force in three dimensions.

Healthy leg equations:

$$F_{ankle,\ xyz} = (m_{foot} * a_{foot,\ xyz}) - GRF_{xyz} - W_{foot,\ xyz}$$

$$F_{knee,\ xyz} = (m_{leg} * a_{leg,\ xyz}) + F_{ankle,\ xyz} - W_{leg,\ xyz}$$

Injured leg equation:

$$F_{knee,\ xyz} = (m_{total} * a_{total,\ xyz}) - GRF_{xyz} - W_{total,\ xyz}$$

All trials for both legs were normalized in one plot as a function of the stance phase. This normalization allows us to notice changes in force distribution at the same point in the total percent of the stance phase. This was done for both Ground Reaction Forces and Knee Reaction Forces in three dimensions.

To ensure that the user was walking with similar cadence between both legs, the total time spent in stance phase was gathered for each trial on each leg. Figure 18 and Table 3 show these values over each trial. This set of data for total stance times indicates that the injured leg exhibits a slightly shorter stance time, but we cannot determine whether or not the user's cadence is even or not because we did not measure the total cycle time for each step. Duration of stance phase for the healthy leg is very slightly longer than that of the injured leg by only 0.075 seconds on average. This difference may have an impact on the overall gait of the subject, but based on this variable alone we cannot make any definite conclusions about the cadence of the user's gait.

Table 3: Stance times for each foot over all trials

	Healthy Leg Stance Time [s]	Injured Leg (IDEO) Stance Time [s]
Trial 1	1.000	0.900
Trial 2	1.034	0.966
Trial 3	1.083	0.992
Trial 4	0.958	0.908
Trial 5	-	0.950
Average ± St. Dev	1.018 ± 0.053	0.943 ± 0.039

The data gathered for the user's injured leg with the IDEO was similar to that of her healthy leg. Slight variability was observed between the trials for the injured leg in the X and Y directions. The greatest amount of variability was observed in the Y direction (lateral) for both legs. This is true partially due to the fact that Y reaction forces are smaller in magnitude than others, and slight changes in movements between trials can influence this data with noise. Please note that the values reaction forces in the Y direction are inverted, as the right and left leg both have medial and lateral components in opposite directions.

It is clear from the data that both the ground and knee reaction forces experienced with the IDEO are more delayed, as the expected peaks in the data come later in the stance phase. As shown in the typical forces experienced during gait, we expected a certain number of peaks for each set of data in each direction. We expected two pronounced peaks in the X and Z directions, and three peaks in the Y direction. The data gathered clearly show these peaks with good accuracy.

Ground Reaction Forces

The ground reaction forces observed in the data show interesting patterns. These external forces drive the kinematics of the body. Since the ankle, knee, and hip reactions are related to these forces, these results are extremely valuable.

The results for ground reaction force in the X direction (forward) show a strong correlation between both legs. However, slight differences in the two peak values were observed, as the injured leg showed peaks with lesser magnitude than the healthy leg. Both of the peaks for the X direction also took longer to develop during stance phase. For the Y and Z components, similar observations were made. For both directions, all of the expected peaks had similar magnitude. Also, the earlier peaks took longer to develop, but the final peak for each X and Y occurred at the same point of the stance phase. The plots for all Ground Reaction Force trials are shown in Figures 19 —24.

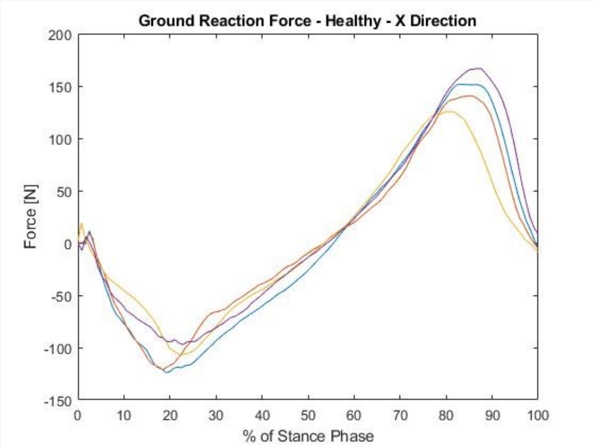


Figure 19: Healthy Ground Reaction Force - X Direction

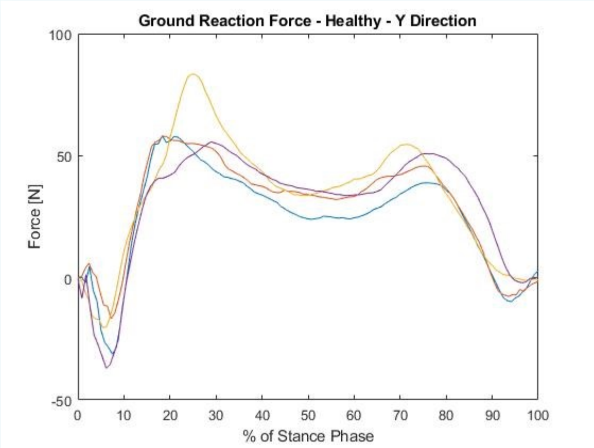


Figure 20: Healthy Ground Reaction Force - Y Direction

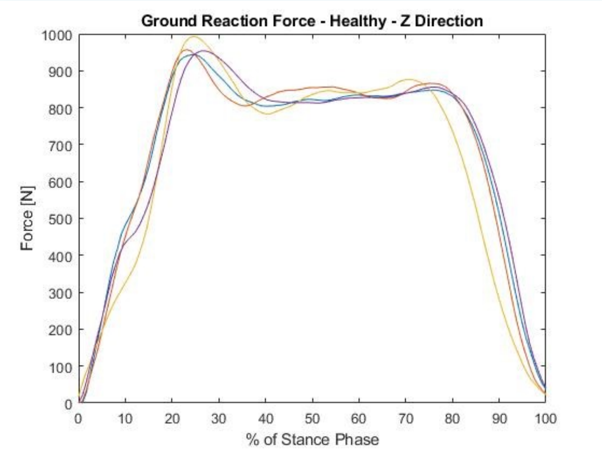


Figure 21: Healthy Ground Reaction Force - Z Direction

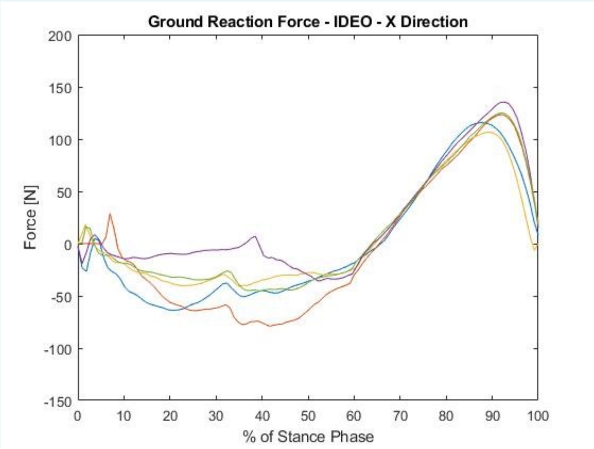


Figure 22: IDEO Ground Reaction Force - X Direction

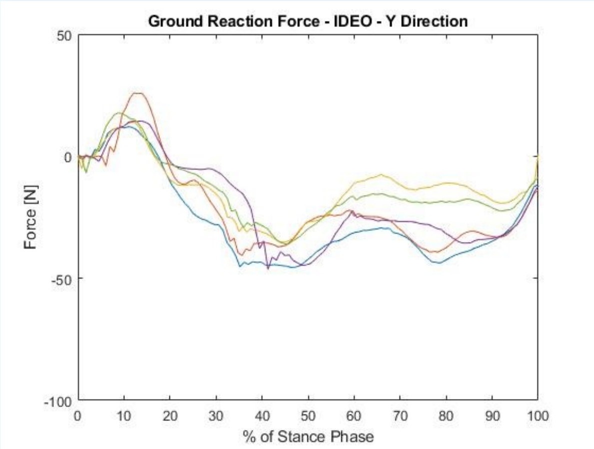


Figure 23: IDEO Ground Reaction Force - Y Direction

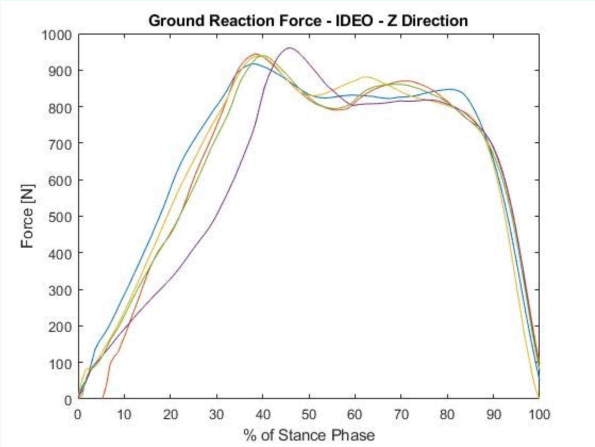


Figure 24: IDEO Ground Reaction Force - Z Direction

Knee Reaction Forces

Since the ground reaction force is directly correlated to the knee reaction force, they follow similar patterns seen in Figures 25-30. The observed differences for peaks in knee reaction force are the same as the differences in ground reaction force peaks. In general, there was limited variability in all trials measuring the subject's healthy leg. There are clear patterns in the reaction forces in all three dimensions. This is especially true for the X and Z reaction forces, which exhibit peaks and troughs at similar instances. The knee reaction force in the Y direction (lateral) between trials varied more than in the other directions.

The data gathered for the IDEO knee reaction force was much more interesting than the data for the healthy leg. For the X direction, there is initially a large amount of variability before reaching midstance and toe off. After this point, the data becomes much more coherent, and a clear pattern can be observed. Similarly, the knee reaction force in the Y direction shows a clear pattern at first, but becomes more scattered once midstance is achieved. Note that the value for knee reaction force in the Y direction is opposite to the healthy leg, since positive Y values are to the right of the subject and vice versa. The pattern observed for the Z direction (vertical) knee reaction force was very similar to the healthy leg. However, the patterns between X and Y trials were slightly different for each leg, but the peaks of the same magnitude can clearly be observed for all of these trials.

Identifying IDEO Function — Knee Reaction Forces

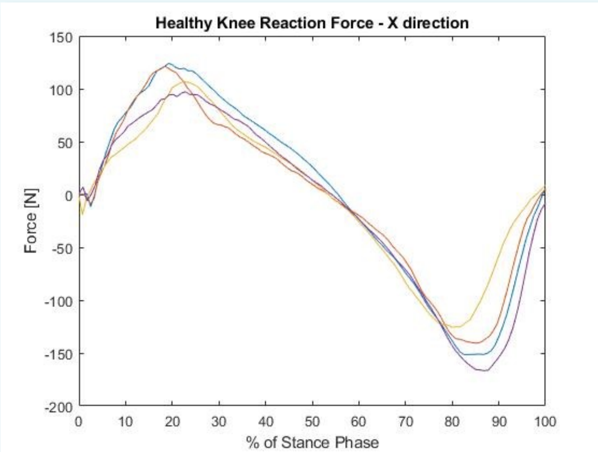


Figure 25: Healthy Knee Reaction Force - X Direction

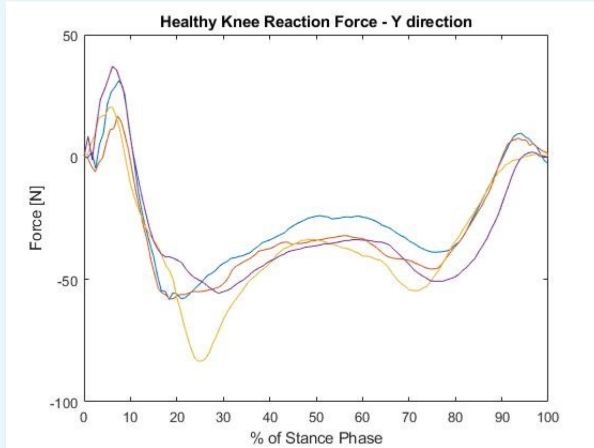


Figure 26: Healthy Knee Reaction Force - Y Direction

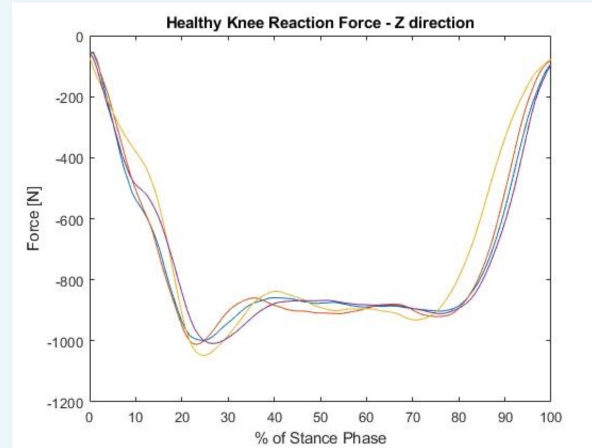


Figure 27: Healthy Knee Reaction Force - Z Direction

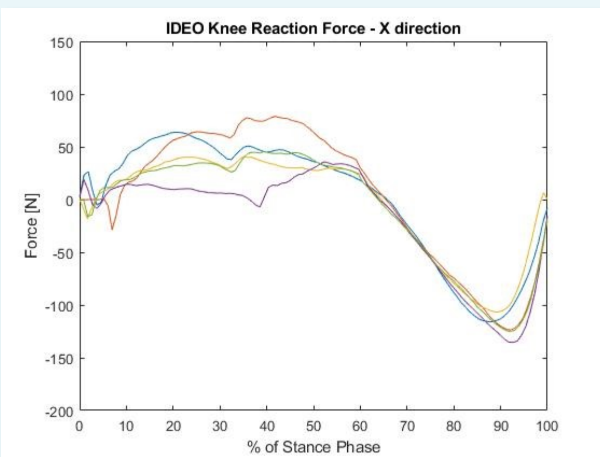


Figure 28: IDEO Knee Reaction Force - X Direction

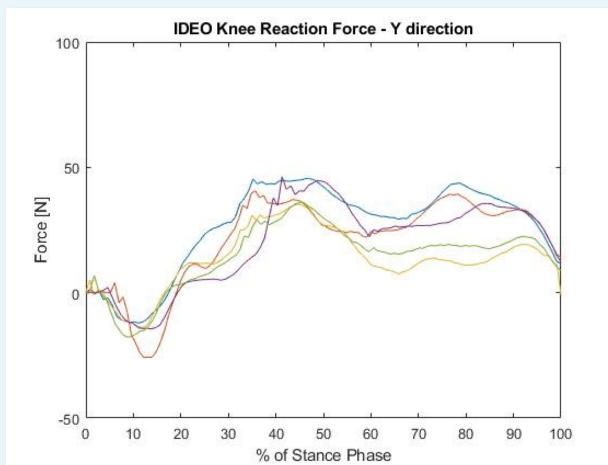


Figure 29: IDEO Knee Reaction Force - Y Direction

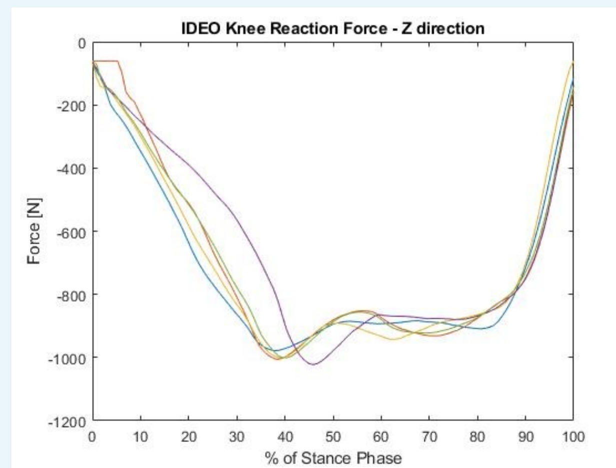


Figure 30: IDEO Knee Reaction Force - Z Direction

Using Instron 5544 & Bluehill Software

As previously described, the IDEO uses two carbon fiber struts along the posterior end of the device. During walking, the flexion in these struts is the driving action that allows for energy storage and transfer, replacing the function of injured muscles. As reported by the inventor of the IDEO, the device was modelled to utilize the struts as the only active components within the device. This means that the foot bed and knee cuff are rigid bodies that do not deflect significantly when force is applied during gait.

To accurately model the energy storage that the device utilizes, the properties of the struts were acquired. A three point bend test was performed on solid, pultruded, unidirectional, carbon fiber rods to obtain the stress-strain relationship for this material. This was done using an Instron 5544 machine in conjunction with Bluehill Instron software as shown in Figure 31 above. This test applied force to the midpoint of the rod, which was simply supported at both ends. The test was stopped once the rod had visibly broken, shown in Figure 32.

This combination of hardware and software provided the relationship between force and displacement at the midpoint of our carbon fiber rods. To create the stress-strain curve for our specific material, we used the dimensions of the rods tested. By performing this test, we were able to determine the elastic modulus of the material and the full stress-strain relationship by using the following equation.

$$E = (PL^3) / (48*d*I)$$

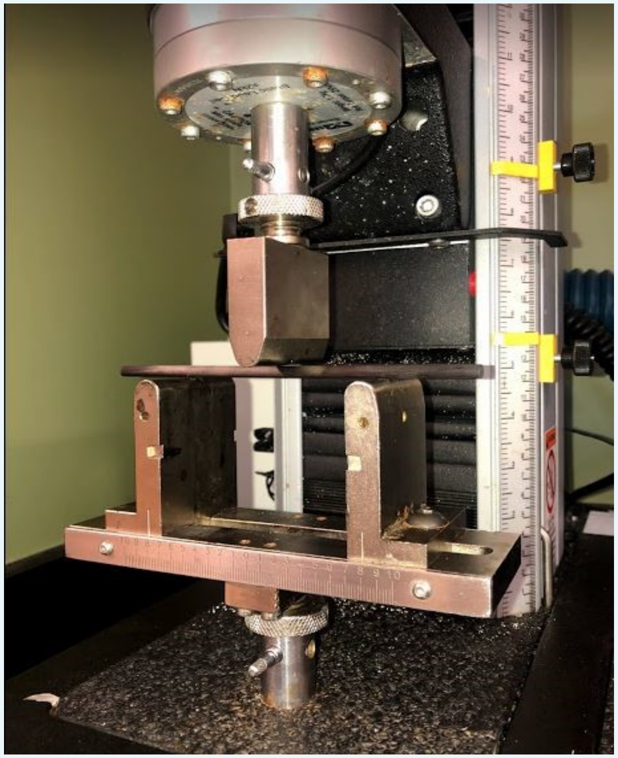


Figure 31: Three Point Bend Test experiment using Instron 5544

Figure 32: Broken Carbon Fiber rod after bending experiment

Stress-Strain Relationship of Active Carbon Fiber Struts

After acquiring the relationship between force and extension using the Instron 5544, a stress-strain curve was calculated using the dimensions of the carbon fiber struts used in the IDEO. The elastic limit for this specific material occurs around 6530 MPa. This type of carbon fiber exhibits a flexural modulus of 326 GPa. Although carbon fiber with much higher stiffness is available, the combined relationship between stiffness, weight, and cost are all favorable for this application. The following graphic shows the force vs displacement curve obtained from the Intron experiment in Figure 33.

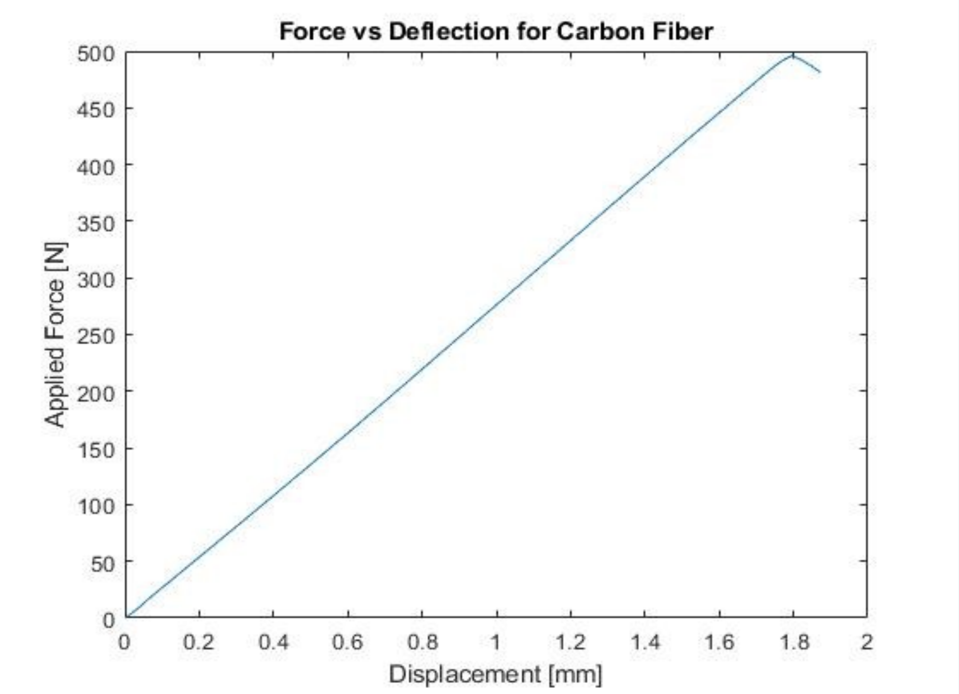


Figure 33: Force vs Displacement curve for Carbon Fiber using Instron 5544

Bending Stress and Energy

Since the IDEO uses carbon fiber struts identical to the rods tested, we are able to calculate the strain energy stored in these struts during gait using forces and moments acting on the struts. The following free body diagram shows how the forces acting on the IDEO relate to the coordinate system created by it. The X and Z axes as related to the struts are different than the X and Z axes as related to the Global Coordinate System (GCS) that was used to determine relevant forces.

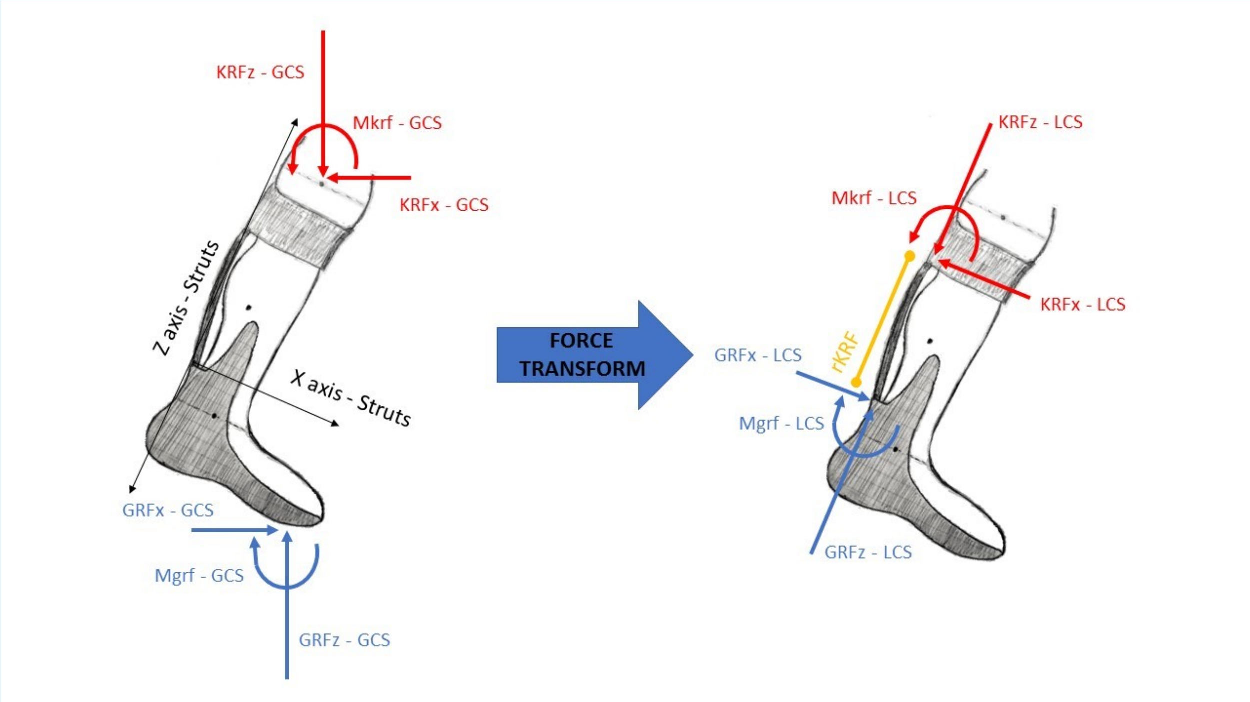


Figure 34: Free Body Diagrams of a user's injured leg before and after force transformations

As shown in Figure 34 above, the ground reaction force acting on the foot bed of the IDEO and the knee reaction force are the forces that create flexion in the struts. The knee and ground reaction forces are resolved into global X and Z components, as this is the specific data we have gathered over the stance phase. These vectors were transformed using a force transformation matrix that utilized the change in orientation of the struts relative to the original Global Coordinate System. This matrix is available within the MatLab code in Appendix A. The result created a Local Coordinate System (LCS) that includes forces acting on both ends of the struts.

These forces create moments acting on the struts, and by using the measured dimensions of the IDEO and relevant body segments, we calculated the total bending moment. This was done by summing the moments and forces with relevant moment arms. The plot that shows this bending moment over the stance phase for all trials is shown in Figure 35.

The moments acting on the struts cause bending stress, and the following equation shows how bending stress was calculated. This stress is plotted over stance phase for all trials, shown in Figure 36.

$$Stress_{max,\ bending} = (M_{total} * y) / I$$

$$= M_{GRF} + M_{KRF} + (rKRF * F_{KRF,x}) * y / I_{IDEO}$$

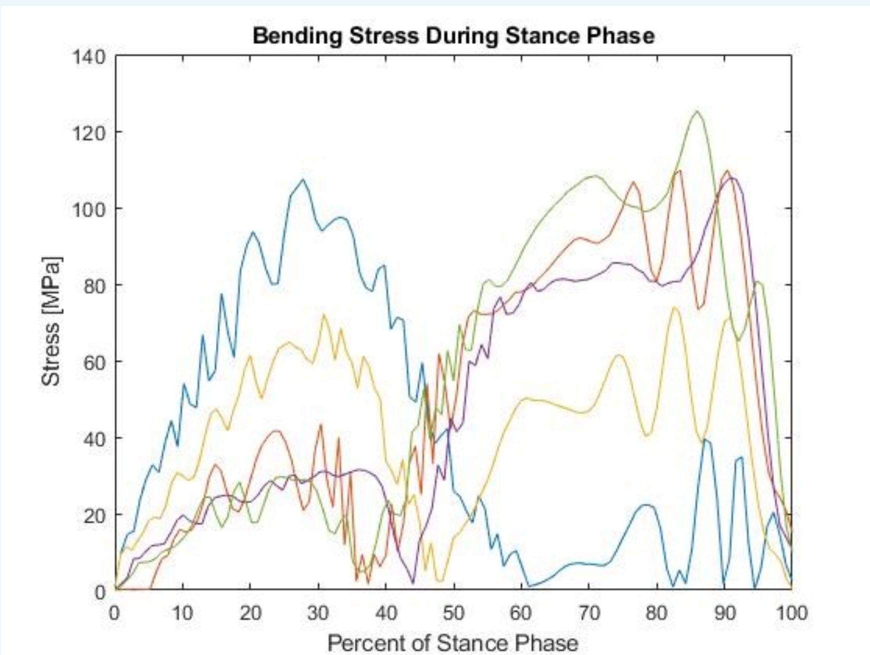


Figure 36: Total Bending Stress applied to IDEO struts

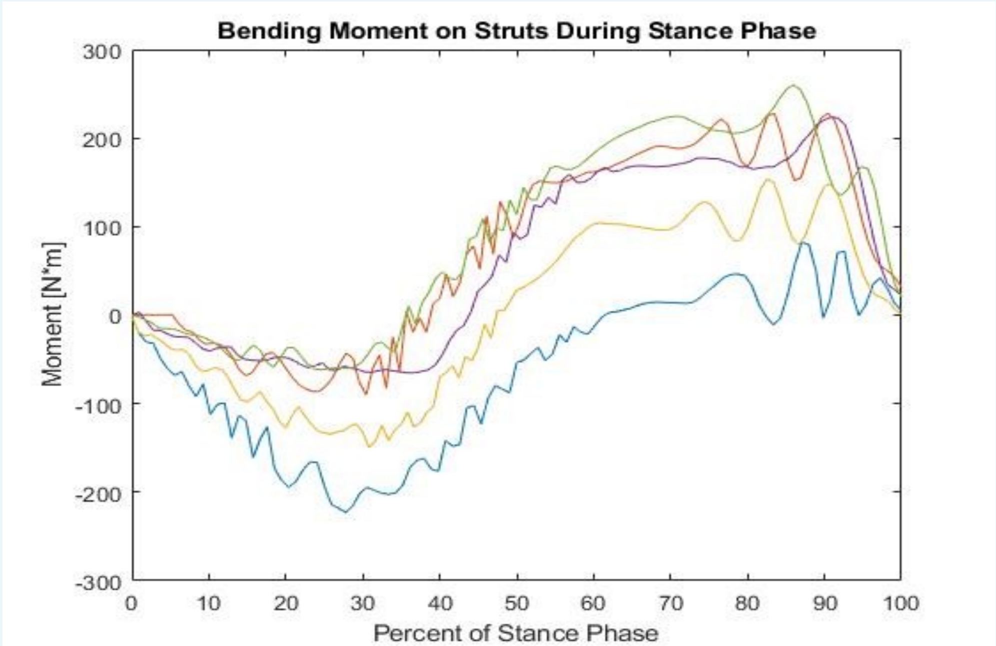


Figure 35: Total Bending Moment applied to IDEO struts

The maximum amount of bending stress occurs anywhere between 70-130 MPa over the five trials. Since the yield limit for this material is 6530 MPa, the device can withstand this amount of stress on a consistent basis.

Using this model, we are able to quantify the strain energy stored in the device during the gait cycle in Joules. The strain energy is calculated using the following equation on page 36. This stored energy is a critical component to the function of the device, since an injured foot will not be able to generate and absorb this amount of energy without causing pain. Figure 37 shows this amount of strain energy stored in the struts of the IDEO during stance phase.

$$U = (M_{total}\text{^}2 * L) / (2*E*I)$$

The strain energy in the struts of the IDEO ranges anywhere from 4-11 e+13 Joules. There are two distinct peaks in this graph, as well as for the graphs for bending moment and bending stress. The combined body segment created by the IDEO allow for bending of the struts at the points when the X reaction force at the knee, the Ground Reaction Moment, and Knee Reaction Moment all experience peaks. These peaks are all occurring at relatively the same percent of gait cycle (at heel strike and toe off), and the combined total stored energy shows this pattern as well. For most of the trials, the bending at heel strike seems to be relatively small compared to the bending at toe off.

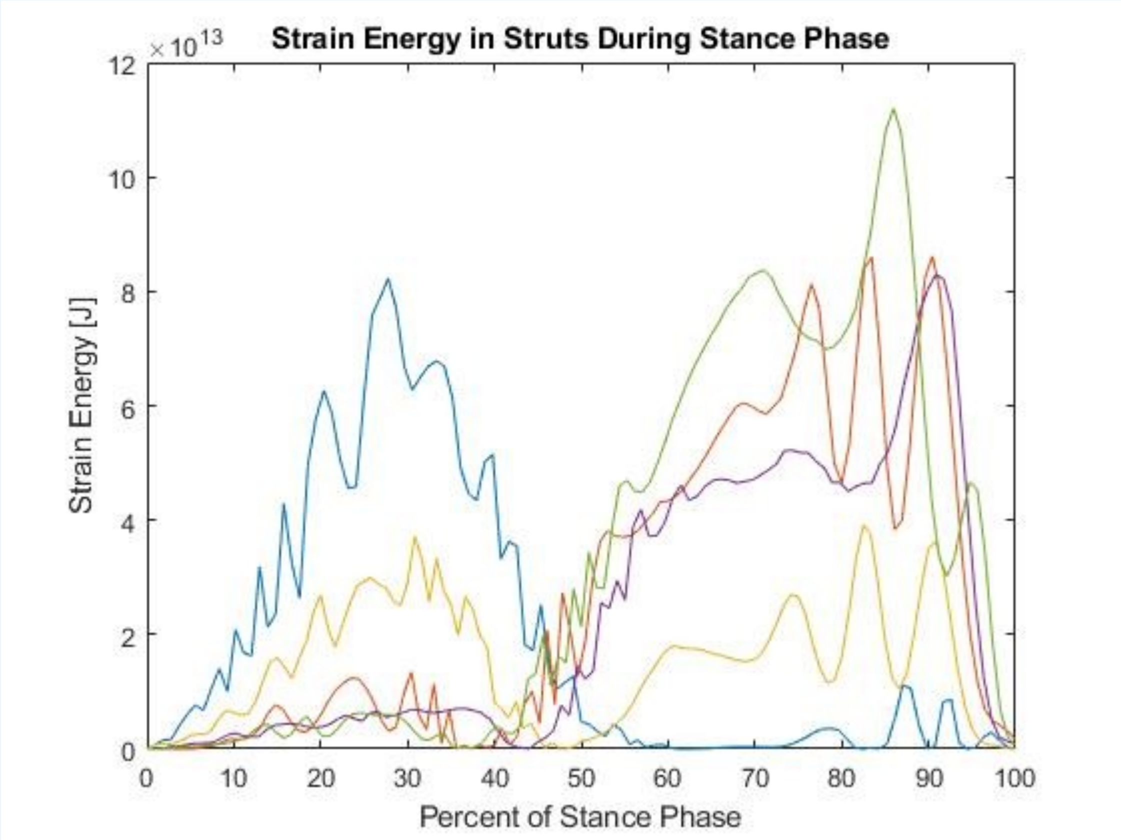


Figure 37: Total strain energy stored in IDEO struts

Conclusions

The overall gait and balance of the subject was determined to be similar across both legs. However, very slight differences in duration of the stance phase were noted. Healthy gait involves even cyclic motion, and visible limping could have been identified by these factors. However, the IDEO does allow for evenly distributed forces and limited shifts in balance during gait.

Force and Motion trials all showed similar patterns in all three axes. However, there were some noted discrepancies between the data sets that indicate how the IDEO impacts the user's gait.

In the X direction, IDEO peak reaction forces were lower than healthy peak forces, and both peaks at Heel Strike and Toe Off took longer to develop. This decrease in magnitude at the injured leg indicates that the user is rolling through her stance phase, as opposed to pushing or pulling with her foot through this step. In the Y direction, all reaction force peaks had similar magnitude, but the first two peaks experienced at Heel Strike and Midstance occurred later in the stance phase than healthy peaks. Similarly, all reaction force peaks in the Z direction had similar magnitude, but the first peak experienced at Heel Strike took longer to develop within the stance phase.

All of the expected peaks in reaction force data at Heel Strike for the subject's injured leg took longer to develop than that of the healthy leg. This is potentially due to the fact that each IDEO user is taught to focus on striking the ground with the heel in order to roll through Midstance and Toe Off (Blanck, 2018). This focus on perfecting gait to adjust to the injury could be causing the shifts in Heel Strike peak force data.

The material properties of the struts of the IDEO are favorable for this application. The maximum applied bending stress is so low relative to the stiffness of the material that use of the device while walking will not impact the life of the device. We assumed that the rest of the device is a rigid body for modelling purposes, but in reality, some small amounts energy are also potentially stored or dissipated in slight flexion of the knee cuff and foot bed as well. For the first peak in stored energy, it seems that the inability

Conclusions

to plantar flex the foot leads to some bending in the struts, as the ground force is all at the heel at first. As expected, there are greater amounts of flexion of the struts at toe off, and it appears that almost all of the energy is restored to the leg before toe off. This study examined the mechanical performance and function of the IDEO and the results show that the device is not only capable of restoring function of injured body segments, but can also sustain this performance over a long period of time. The reaction forces allowable by the device and the mechanical responses within the device were quantified and show this to be true.

Limitations and Recommendations for Further Study

Modelling of biomechanics is typically based on Anthropometric data. This study also used Anthropometric tables to determine relevant dimensions and values to model the gait of the user. These values are not exact, and the results of our experiments should take this into account.

The team first attempted to model the gait of an individual using modelling softwares such as SolidWorks and Creo. This plan involved first creating a human model that would be made to walk. The software could then output data similar to the forces and moments quantified in this study. This was intended to be done using the kinematic data gathered using Motion Capture or Motion Sensing. However, using inverse kinematics to calculate these forces proved to be inconsistent with expected values. The team used a fourbar mechanism to test this theory. The results of this test were insufficient to proceed to a human model. Similarly, other attempts to create the human assembly itself proved to be difficult, as the combination of softwares used did not allow for sufficient material and mechanical properties.

For this type of model to produce accurate results, the overall material properties should be optimized to the exact type of human model tested, i.e. a 75th percentile female. Additionally, the linkages that connect all body segments should match the specific joints in the body. Simple revolute and pin joints will not be sufficient in the modelling of the human body, as this would indicate an ideal joint instead of one that can experience soft body deformation. Further development in the computational modelling of human gait should be explored, as this could provide insight into the importance of many different variables associated with gait.

This project was not able to identify the joint torques and angles experienced over the gait cycle for the user. This was due to the fact that we could not accurately track relevant distances over time that would correspond to relevant moment arms used in calculations. To accurately track these results, the team would need more accurate sensing equipment that would correlate with force sensing. Specifically, the distance between the point at ground reaction force (center of pressure, COP) and other points in the body would need to be accurately identified over time to produce joint reaction moments. These data would provide additional insight into the function of the IDEO and how its reactions compare to the reactions for the user's healthy leg.

There were multiple assumptions in the modelling of the bending experienced in the device. As mentioned in the report, the device is intended to be rigid at the knee cuff and foot plate, allowing for bending only in the struts. However, slight deflections in these other components of the device allow for force absorption and energy storage over time. This project only identified the bending of one component of the device, but deflections in other components should not be ignored entirely. The bending model used a three point bend test with simple supports to create the stress-strain profile for our material. During use, the struts are not simply supported, and are buried in the carbon fiber layup of the other components. This discrepancy could have affected the results of the bending and energy experiments. Similarly, any inaccuracies from the motion data gathered over time will have affected the accuracy of forces and energy calculated. Slight movements of the user's foot within the device are expected to be minimal, but this motion should not be ignored as it can and may have affected our results. The team initially intended on using strain gauges to measure variables indicating material properties over time, but due to a lack of resources and compatibility between software used, the team used a bending beam model instead. It would be interesting to redo this experiment with and without strain gauges to see if the models match or not.

Lastly, while this test examined the duration of stance phase for each leg, the team was not able to definitively claim whether or not the subject's cadence during gait was even. This is a way to show whether or not the user is limping while using the device. For this experiment to be more robust, a metronome or other device could have been used to refer the user to a common cadence for her steps. This would have allowed us to make claims on whether or not the device allows for an even or symmetric gait.

References

Blanck, Ryan (September 26, 2018) Personal Interview.

Chung (2018). Lower Limb Orthoses. Braddom's Rehabilitation Care: A Clinical Handbook. 75-84.

Hanger Clinic (2018). ExoSym Leg Brace. Retrieved from: http://www.hangerclinic.com/bracing-support/adult-le/pages/limb-salvage-Exosym.aspx

Hanger Clinic (2019). Adult Lower Limb and Leg Orthotics. Retrieved from: http://www.hangerclinic.com/bracing-support/adult-le/pages/default.aspx

Intermountain Healthcare (2018). Pain Management - Hip and Leg Pain. Retrieved from: https://intermountainhealthcare.org/services/pain-management/conditions/hip-and-leg-pain/

Loudon J, et al (2008). Human Kinetics. The clinical orthopedic assessment guide. (2) 395-408.

Maggs, S (2012). Important Mechanical Properties, University of Warwick. Retrieved from: https://warwick.ac.uk/fac/sci/wmg/globalcontent/courses/ebm/mant/materials/properties_of_materials/

Nisam Amirudin, A., Parasuraman, S Kadirvelu, A., Khan, A., Elamvazuthi, I. (2014). Biomechanics of Hip, Knee and Ankle Joint Loading during Ascent and Descent Walking. Procedia Computer Science. 42. 336 - 344. 10.1016/j.procs.2014.11.071.

Physiopedia. (2019). Gait . Retrieved from: https://www.physio-pedia.com/index.php?title=Gait&oldid=206012

Samuel, Leslie (2019). Interactive Biology. *Medical Terminology Explained. Retrieved from: http://www.interactive-biology.com/4412/medical-terminology-explained/*

Shultz, SJ et al. (2005). Human Kinetics. Examination of musculoskeletal injuries. (2) 55-60.

Souza, T. R., Pinto, R. Z., Trede, R. G., Kirkwood, R. N., & Fonseca, S. T. (2010). Temporal couplings between rearfoot–shank complex and hip joint during walking. Clinical biomechanics, 25(7), 745-748.

Sports Podiatry Resource Inc. (2019). Foot Skeletal Structure. Retrieved from: http://sportspodiatry.com/footcare/index.htm

Stride Strong (2019). Physical Therapy - Knee & Leg Injury Treatment & Therapy. Retrieved from: https://stridestrong.com/leg-injury-therapy/

University of Texas - Arlington, Dept. of Kinesiology (n.d.). Mechanical Properties of Materials. Retrieved from: https://wweb.uta.edu/faculty/ricard/Classes/KINE-3301/Notes/Lesson-14.html

Van Cott, Charlene (September 7, 2018). Personal interview.

Vaughan, S.L., Davis, B.L. & O'Connor, J.C. (1999). Dynamics of Human Gait (2nd ed.). Kiboho Publishers.

Winter, D.A. (2009). Biomechanics and Motor Control of Human Movement (4th ed.). Hoboken, NJ: John Wiley & Sons.

Winter, D.A., Robertson, E., Gordon, D. (1980). Mechanical energy generation, absorption and transfer amongst segments during walking. Journal of Biomechanics, 13(10). 845-854.

WebMD (2019). Foot Drop: Causes, Symptoms, and Treatment. Retrieved from: https://www.webmd.com/a-to-z-guides/foot-drop-causes-symptoms-treatments

Appendix A: Full MatLab Code - Injured Leg & Energy Calculations

Two MatLab Scripts were created to model the forces and motion of the subject, with one script analyzing all healthy data and the other script analyzing all data related to the user's injured leg. Subsequent calculations related to material properties, bending, stress, and energy are also contained in the injured leg's script. Many different sections of this code were progressively edited for robustness, and the following code could be used to generate any plots associated with this project. The script for the injured leg comes first in Appendix A then the script for the healthy leg comes in Appendix B.

```
%% Import files for coincident GRF (txt) and motion (csv) of all trials

fnameforcetrial1 = "IDEOtrial1txt.txt" ;

rawforcedatatrial1 = importdata (fnameforcetrial1) ;

fnamemotiontrial1 = "IDEOtrial1csv.csv" ;

rawmotiondatatrial1 = csvread (fnamemotiontrial1) ;

% Stance times

% trial 1: 0.525 , 1.425

% trial 2: 0.742 , 1.708

% trial 3: 0.575 , 1.567

% trial 4: 0.617 , 1.525

% trial 5: 0.483 , 1.433

% INPUTS for this type of trial

HStimetrial1 = (0.525) ; % input [s]

TOtimetrial1 = (1.425) ; % input [s]

Lthick = (0.1099) ; %input [m] 4.33 in

Lwidth = (0.1295) ; %input [m] 5.1 in

Llength = (0.4064) ; %input [m] 16 in

footlength = (0.1575) ; %input [m] 6.2 in

mbody = (91.6257) ; %input [kg] 202 pounds

mIDEO = (0.68) ; %input [kg] 1.5 pounds

% extracting column data for force *check y force

xforcetrial1 = rawforcedatatrial1 (:,1) ;

yforcetrial1 = rawforcedatatrial1 (:,2) ; % positive should be right side (lateral for right foot)

zforcetrial1 = rawforcedatatrial1 (:,3) ;

% set sampling rate and inital time domain for raw data

Fs = 120 ;

forcesamplestrial1 = length (rawforcedatatrial1) ;

forcetimetrial1 = (0:(1/Fs):((forcesamplestrial1/Fs)-(1/Fs))) ;

% plot raw force data over time in three dimensions
```

```
% extract sample numbers from motion data to create time domain
% delete the column with non numericals

motionsamplestrial1 = rawmotiondata-trial1(:,4);
motiontimetrial1 = (0:(1/Fs):(((length(motionsamplestrial1)/2)/Fs)-(1/Fs))) ;

% organize data by sensor/hub with if statements

hub1sensor1trial1 = zeros((floor(length(motionsamplestrial1)/2)), 6) ;

j = 1;
for i = 1 : (length(rawmotiondatatrial1))
    if (rawmotiondatatrial1(i,2) == 1) && (rawmotiondatatrial1(i,3) == 1)
        hub1sensor1trial1(j, :) = rawmotiondatatrial1(i, 5:10);
        j = j + 1;
    end
end

hub1sensor2trial1 = zeros((floor(length(motionsamplestrial1)/2)), 6) ;

j = 1;
for i = 1 : (length(rawmotiondatatrial1))
    if (rawmotiondatatrial1(i,2) == 1) && (rawmotiondatatrial1(i,3) == 2)
        hub1sensor2trial1(j, :) = rawmotiondatatrial1(i, 5:10);
        j = j + 1;
    end
end

hub1sensor3trial1 = zeros((floor(length(motionsamplestrial1)/2)), 6) ;

j = 1;
for i = 1 : (length(rawmotiondatatrial1))
    if (rawmotiondatatrial1(i,2) == 1) && (rawmotiondatatrial1(i,3) == 3)
        hub1sensor3trial1(j, :) = rawmotiondatatrial1(i, 5:10);
        j = j + 1;
    end
end

hub2sensor1trial1 = zeros((floor(length(motionsamplestrial1)/2)), 6) ;

j = 1;
for i = 1 : (length(rawmotiondatatrial1))
    if (rawmotiondatatrial1(i,2) == 2) && (rawmotiondatatrial1(i,3) == 1)
        hub2sensor1trial1(j, :) = rawmotiondatatrial1(i, 5:10);
        j = j + 1;
    end
end

hub2sensor2trial1 = zeros((floor(length(motionsamplestrial1)/2)), 6) ;

j = 1;
for i = 1 : (length(rawmotiondatatrial1))
    if (rawmotiondatatrial1(i,2) == 2) && (rawmotiondatatrial1(i,3) == 2)
        hub2sensor2trial1(j, :) = rawmotiondatatrial1(i, 5:10);
        j = j + 1;
    end
end
```

```
% Sensor locations (lateral)
% 1,1 = COM of IDEO
% 1,2 = femoral condyles at knee
% 1,3 = heel
% 2,1 = back of struts - not lateral
% 2,2 = toes - metatarsals

% here we can fix the sign of each set of data if we want, and correct the
% value of the displacement data for the segment using the COM of the
% total segment

mfootandleg = 0.061*mbody ;
msegment = mfootandleg + mIDEO ;

Ithick = ((0.5*Lthick) + 0.0502) / msegment ;
Iwidth = (Lwidth + 0.02) / msegment ;
Ilength = ((0.394*Llength) + 0.205) / msegment ;

intom = 0.0254 ; %[m/in]

%extract displacement data , use off-sets for COM

IDEOdispXtrial1 = intom*hub1sensor1trial1(:,1) - Ithick ;
IDEOdispYtrial1 = intom*hub1sensor1trial1(:,2) - Iwidth ;
IDEOdispZtrial1 = intom*hub1sensor1trial1(:,3) - Ilength ;

% plot raw displacement data over time in three dimensions

%Normalized Data inputs for full step with force plate

HSsampletrial1 = floor(HStimetrial1*Fs) - 2 ; %accounts for double differentia-tion
TOsampletrial1 = floor (TOtimetrial1*Fs) ;
normalsampleswrongtrial1 = (0:(TOsampletrial1 - HSsampletrial1)) ;
normalsamplestrial1 = transpose (normalsampleswrongtrial1) ;

%make the displacement data normalized

normalIDEOdispXtrial1 = IDEOdispXtrial1 (HSsampletrial1:TOsampletrial1) ;
normalIDEOdispYtrial1 = IDEOdispYtrial1 (HSsampletrial1:TOsampletrial1) ;
normalIDEOdispZtrial1 = IDEOdispZtrial1 (HSsampletrial1:TOsampletrial1) ;

%FILTER NORMALIZED DISPLACEMENT DATA, THEN DIFFERENTIATE IT

%design fifth order butterworth filter

Wp = 5/60;
Ws = 20/60;

[n,Wn] = buttord(Wp,Ws,3,60) ;

[z,p,k] = butter(n,Wn);
filter1 = zp2sos(z,p,k);

freqz(filter1,66,120)
title(sprintf('n = %d Butterworth Low-pass Filter',n))

%filter foot and leg displacement data
```

```
& plot

filteredIDEOdispXtrial1 = filtfilt
(filter1, n, normalIDEOdispXtrial1) ;

filteredIDEOdispYtrial1 = filtfilt
(filter1, n, normalIDEOdispYtrial1) ;

filteredIDEOdispZtrial1 = filtfilt
(filter1, n, normalIDEOdispZtrial1) ;

%use normalsamples time domain and plot
filtered displacement data

%differentiate these displacement val-
ues to get velocity

filteredIDEOvelXtrial1 = diff
(filteredIDEOdispXtrial1) ;

filteredIDEOvelYtrial1 = diff
(filteredIDEOdispYtrial1) ;

filteredIDEOvelZtrial1 = diff
(filteredIDEOdispZtrial1) ;

% make time domain and plot

veltimetrial1 = 0:(1/Fs):((1/Fs)*length
(filteredIDEOvelXtrial1) - (1/Fs)) ;

%differentiate these velocity values to
get accleration

finalIDEOaccXtrial1 = diff
(filteredIDEOvelXtrial1) ;

finalIDEOaccYtrial1 = diff
(filteredIDEOvelYtrial1) ;

finalIDEOaccZtrial1 = diff
(filteredIDEOvelZtrial1) ;

%create time domain & plot

acceltimetrial1 = (1/Fs) * (0:(length
(finalIDEOaccXtrial1)-1)) ;

% combine variables into 3D matrices

normalxforcetrial1 = xforcetrial1
((HSsampletrial1+2):TOsampletrial1) ;

normalyforcetrial1 = yforcetrial1
((HSsampletrial1+2):TOsampletrial1) ;

normalzforcetrial1 = zforcetrial1
((HSsampletrial1+2):TOsampletrial1) ;

IDEOacctrial1 = [finalIDEOaccXtrial1
finalIDEOaccYtrial1 finalIDEOac-
cZtrial1] ; %[m/s^2]

GRFtrial1 = [normalxforcetrial1 nor-
malyforcetrial1 normalzforcetrial1] ; %
[N]

%use IDEOacc with GRF to calculate knee
reaction force

gravity = [0 0 9.8] ; %[m/s^2] Z direc-
tion!

KRF3Dtrial1 = (msegment.*IDEOacctrial1)
- GRFtrial1 - (msegment.*gravity) ;

%take out components of KRF3D

krXtrial1 = KRF3Dtrial1(:,1) ;

krYtrial1 = KRF3Dtrial1(:,2) ;

krZtrial1 = KRF3Dtrial1(:,3) ;
```

```
kneetimetrial1 = acceltimetrial1 ;


%% Import files for coincident GRF (txt)
and motion (csv) of all trials


fnameforcetrial2 = "IDEOtrial2txt.txt" ;

rawforcedatatrial2 = importdata
(fnameforcetrial2) ;


fnamemotiontrial2 =
"IDEOtrial2csv.csv" ;

rawmotiondatatrial2 = csvread
(fnamemotiontrial2) ;


% Stance times

% trial 1: 0.525 , 1.425

% trial 2: 0.742 , 1.708

% trial 3: 0.575 , 1.567

% trial 4: 0.617 , 1.525

% trial 5: 0.483 , 1.433


% INPUTS for this type of trial


HStimetrial2 = (0.742) ; % input [s]

TOtimetrial2 = (1.708) ; % input [s]

% extracting column data for force
*check y force


xforcetrial2 = rawforcedatatrial2(:,1) ;

yforcetrial2 = rawforcedatatrial2(:,2) ;
% positive should be right side (lateral
for right foot)

zforcetrial2 = rawforcedatatrial2(:,3) ;


% set sampling rate and inital time do-
main for raw data


forcesamplestrial2 = length
(rawforcedatatrial2) ;

forcetimetrial2 = (0:(1/Fs):
((forcesamplestrial2/Fs)-(1/Fs))) ;


% plot raw force data over time in three
dimensions


% extract sample numbers from motion da-
ta to create time domain

% delete the column with non numericals


motionsamplestrial2 = rawmotiondata-
trial2(:,4);

motiontimetrial2 = (0:(1/Fs):(((length
(motionsamplestrial2)/2)/Fs)-(1/Fs))) ;


% organize data by sensor/hub with if
statements


hub1sensor1trial2 = zeros((floor(length
(motionsamplestrial2)/2)), 6) ;


j = 1;

for i = 1 : (length
(rawmotiondatatrial2))

    if (rawmotiondatatrial2(i,2) == 1)
&& (rawmotiondatatrial2(i,3) == 1)

        hub1sensor1trial2(j, :) = rawmo-
tiondatatrial2(i, 5:10);

        j = j + 1;

    end

end


hub1sensor2trial2 = zeros((floor(length
(motionsamplestrial2)/2)), 6) ;


j = 1;

for i = 1 : (length
(rawmotiondatatrial2))

    if (rawmotiondatatrial2(i,2) == 1)
```

```
&& (rawmotiondatatrial2(i,3) == 2)

        hub1sensor2trial2(j, :) = rawmotiondatatrial2(i, 5:10);

        j = j + 1;

    end

end


hub1sensor3trial2 = zeros((floor(length(motionsamplestrial2)/2)), 6) ;


j = 1;

for i = 1 : (length(rawmotiondatatrial2))

    if (rawmotiondatatrial2(i,2) == 1) && (rawmotiondatatrial2(i,3) == 3)

        hub1sensor3trial2(j, :) = rawmotiondatatrial2(i, 5:10);

        j = j + 1;

    end

end


hub2sensor1trial2 = zeros((floor(length(motionsamplestrial2)/2)), 6) ;


j = 1;

for i = 1 : (length(rawmotiondatatrial2))

    if (rawmotiondatatrial2(i,2) == 2) && (rawmotiondatatrial2(i,3) == 1)

        hub2sensor1trial2(j, :) = rawmotiondatatrial2(i, 5:10);

        j = j + 1;

    end

end


hub2sensor2trial2 = zeros((floor(length(motionsamplestrial2)/2)), 6) ;


j = 1;

for i = 1 : (length(rawmotiondatatrial2))

    if (rawmotiondatatrial2(i,2) == 2) && (rawmotiondatatrial2(i,3) == 2)

        hub2sensor2trial2(j, :) = rawmotiondatatrial2(i, 5:10);

        j = j + 1;

    end

end


% Sensor locations (lateral)

% 1,1 = COM of IDEO

% 1,2 = femoral condyles at knee

% 1,3 = heel

% 2,1 = back of struts - not lateral

% 2,2 = toes - metatarsals


% here we can fix the sign of each set of data if we want, and correct the

% value of the displacement data for the segment using the COM of the

% total segment


%extract displacement data , use offsets for COM


IDEOdispXtrial2 = intom*hub1sensor1trial2(:,1) - Ithick ;

IDEOdispYtrial2 = intom*hub1sensor1trial2(:,2) - Iwidth ;

IDEOdispZtrial2 = intom*hub1sensor1trial2(:,3) - Ilength ;


% plot raw displacement data over time in three dimensions


%Normalized Data inputs for full step with force plate
```

```
HSsampletrial2 = floor(HStimetrial2*Fs) - 2 ; %accounts for double differentiation

TOsampletrial2 = floor(TOtimetrial2*Fs) ;

normalsampleswrongtrial2 = (0:(TOsampletrial2 - HSsampletrial2)) ;

normalsamplestrial2 = transpose(normalsampleswrongtrial2) ;

%make the displacement data normalized

normalIDEOdispXtrial2 = IDEOdispXtrial2(HSsampletrial2:TOsampletrial2) ;

normalIDEOdispYtrial2 = IDEOdispYtrial2(HSsampletrial2:TOsampletrial2) ;

normalIDEOdispZtrial2 = IDEOdispZtrial2(HSsampletrial2:TOsampletrial2) ;

%FILTER NORMALIZED DISPLACEMENT DATA, THEN DIFFERENTIATE IT

%design fifth order butterworth filter

%filter foot and leg displacement data & plot

filteredIDEOdispXtrial2 = filtfilt(filter1, n, normalIDEOdispXtrial2) ;

filteredIDEOdispYtrial2 = filtfilt(filter1, n, normalIDEOdispYtrial2) ;

filteredIDEOdispZtrial2 = filtfilt(filter1, n, normalIDEOdispZtrial2) ;

%use normalsamples time domain and plot filtered displacement data

%differentiate these displacement values to get velocity

filteredIDEOvelXtrial2 = diff(filteredIDEOdispXtrial2) ;

filteredIDEOvelYtrial2 = diff(filteredIDEOdispYtrial2) ;

filteredIDEOvelZtrial2 = diff(filteredIDEOdispZtrial2) ;

% make time domain and plot

veltimetrial2 = 0:(1/Fs):((1/Fs)*length(filteredIDEOvelXtrial2) - (1/Fs)) ;

%differentiate these velocity values to get accleration

finalIDEOaccXtrial2 = diff(filteredIDEOvelXtrial2) ;

finalIDEOaccYtrial2 = diff(filteredIDEOvelYtrial2) ;

finalIDEOaccZtrial2 = diff(filteredIDEOvelZtrial2) ;

%create time domain & plot

acceltimetrial2 = (1/Fs) * (0:(length(finalIDEOaccXtrial2)-1)) ;

% combine variables into 3D matrices

normalxforcetrial2 = xforcetrial2((HSsampletrial2+2):TOsampletrial2) ;

normalyforcetrial2 = yforcetrial2((HSsampletrial2+2):TOsampletrial2) ;

normalzforcetrial2 = zforcetrial2((HSsampletrial2+2):TOsampletrial2) ;
```

```
IDEOacctrial2 = [finalIDEOaccXtrial2 finalIDEOaccYtrial2 finalIDEOaccZtrial2] ; %[m/s^2]

GRFtrial2 = [normalxforcetrial2 normalyforcetrial2 normalzforcetrial2] ; % [N]

%use IDEOacc with GRF to calculate knee reaction force

KRF3Dtrial2 = (msegment.*IDEOacctrial2) - GRFtrial2 - (msegment.*gravity) ;

%take out components of KRF3D

krXtrial2 = KRF3Dtrial2(:,1) ;
krYtrial2 = KRF3Dtrial2(:,2) ;
krZtrial2 = KRF3Dtrial2(:,3) ;

kneetimetrial2 = acceltimetrial2 ;

%% Import files for coincident GRF (txt) and motion (csv) of all trials

fnameforcetrial3 = "IDEOtrial3txt.txt" ;
rawforcedatatrial3 = importdata (fnameforcetrial3) ;

fnamemotiontrial3 = "IDEOtrial3csv.csv" ;
rawmotiondatatrial3 = csvread (fnamemotiontrial3) ;

% Stance times
% trial 1: 0.525 , 1.425
% trial 2: 0.742 , 1.708
% trial 3: 0.575 , 1.567
% trial 4: 0.617 , 1.525
% trial 5: 0.483 , 1.433

% INPUTS for this type of trial

HStimetrial3 = (0.575) ; % input [s]
TOtimetrial3 = (1.576) ; % input [s]

% extracting column data for force *check y force

xforcetrial3 = rawforcedatatrial3(:,1) ;
yforcetrial3 = rawforcedatatrial3(:,2) ; % positive should be right side (lateral for right foot)
zforcetrial3 = rawforcedatatrial3(:,3) ;

% set sampling rate and inital time domain for raw data

forcesamplestrial3 = length (rawforcedatatrial3) ;
forcetimetrial3 = (0:(1/Fs):((forcesamplestrial3/Fs)-(1/Fs))) ;

% plot raw force data over time in three dimensions

% extract sample numbers from motion data to create time domain
% delete the column with non numericals

motionsamplestrial3 = rawmotiondatatrial3(:,4);
motiontimetrial3 = (0:(1/Fs):(((length (motionsamplestrial3)/2)/Fs)-(1/Fs))) ;

% organize data by sensor/hub with if
```

```
statements

hub1sensor1trial3 = zeros((floor(length
(motionsamplestrial3)/2)), 6) ;

j = 1;
for i = 1 : (length
(rawmotiondatatrial3))
    if (rawmotiondatatrial3(i,2) == 1)
&& (rawmotiondatatrial3(i,3) == 1)
        hub1sensor1trial3(j, :) = rawmo-
tiondatatrial3(i, 5:10);
        j = j + 1;
    end
end

hub1sensor2trial3 = zeros((floor(length
(motionsamplestrial3)/2)), 6) ;

j = 1;
for i = 1 : (length
(rawmotiondatatrial3))
    if (rawmotiondatatrial3(i,2) == 1)
&& (rawmotiondatatrial3(i,3) == 2)
        hub1sensor2trial3(j, :) = rawmo-
tiondatatrial3(i, 5:10);
        j = j + 1;
    end
end

hub1sensor3trial3 = zeros((floor(length
(motionsamplestrial3)/2)), 6) ;

j = 1;
for i = 1 : (length
(rawmotiondatatrial3))
    if (rawmotiondatatrial3(i,2) == 1)
&& (rawmotiondatatrial3(i,3) == 3)
        hub1sensor3trial3(j, :) = rawmo-
tiondatatrial3(i, 5:10);
        j = j + 1;
    end
end

hub2sensor1trial3 = zeros((floor(length
(motionsamplestrial3)/2)), 6) ;

j = 1;
for i = 1 : (length
(rawmotiondatatrial3))
    if (rawmotiondatatrial3(i,2) == 2)
&& (rawmotiondatatrial3(i,3) == 1)
        hub2sensor1trial3(j, :) = rawmo-
tiondatatrial3(i, 5:10);
        j = j + 1;
    end
end

hub2sensor2trial3 = zeros((floor(length
(motionsamplestrial3)/2)), 6) ;

j = 1;
for i = 1 : (length
(rawmotiondatatrial3))
    if (rawmotiondatatrial3(i,2) == 2)
&& (rawmotiondatatrial3(i,3) == 2)
        hub2sensor2trial3(j, :) = rawmo-
tiondatatrial3(i, 5:10);
        j = j + 1;
    end
end

% Sensor locations (lateral)
% 1,1 = COM of IDEO
% 1,2 = femoral condyles at knee
% 1,3 = heel
% 2,1 = back of struts - not lateral
% 2,2 = toes - metatarsals

% here we can fix the sign of each set
of data if we want, and correct the
```

```
% value of the displacement data for the segment using the COM of the

% total segment



%extract displacement data , use offsets for COM


IDEOdispXtrial3 = intom*hub1sensor1trial3(:,1) - Ithick ;

IDEOdispYtrial3 = intom*hub1sensor1trial3(:,2) - Iwidth ;

IDEOdispZtrial3 = intom*hub1sensor1trial3(:,3) - Ilength ;


% plot raw displacement data over time in three dimensions



%Normalized Data inputs for full step with force plate


HSsampletrial3 = floor(HStimetrial3*Fs) - 2 ; %accounts for double differentia-tion

TOsampletrial3 = floor (TOtimetrial3*Fs) ;

normalsampleswrongtrial3 = (0:(TOsampletrial3 - HSsampletrial3)) ;

normalsamplestrial3 = transpose (normalsampleswrongtrial3) ;


%make the displacement data normalized


normalIDEOdispXtrial3 = IDEOdispXtrial3 (HSsampletrial3:TOsampletrial3) ;

normalIDEOdispYtrial3 = IDEOdispYtrial3 (HSsampletrial3:TOsampletrial3) ;

normalIDEOdispZtrial3 = IDEOdispZtrial3 (HSsampletrial3:TOsampletrial3) ;


%FILTER NORMALIZED DISPLACEMENT DATA, THEN DIFFERENTIATE IT


%design fifth order butterworth filter


%filter foot and leg displacement data & plot

filteredIDEOdispXtrial3 = filtfilt (filter1, n, normalIDEOdispXtrial3) ;

filteredIDEOdispYtrial3 = filtfilt (filter1, n, normalIDEOdispYtrial3) ;

filteredIDEOdispZtrial3 = filtfilt (filter1, n, normalIDEOdispZtrial3) ;


%use normalsamples time domain and plot filtered displacement data


%differentiate these displacement values to get velocity


filteredIDEOvelXtrial3 = diff (filteredIDEOdispXtrial3) ;

filteredIDEOvelYtrial3 = diff (filteredIDEOdispYtrial3) ;

filteredIDEOvelZtrial3 = diff (filteredIDEOdispZtrial3) ;


% make time domain and plot


veltimetrial3 = 0:(1/Fs):((1/Fs)*length (filteredIDEOvelXtrial3) - (1/Fs)) ;
```

```
%differentiate these velocity values to get accleration

finalIDEOaccXtrial3 = diff (filteredIDEOvelXtrial3) ;

finalIDEOaccYtrial3 = diff (filteredIDEOvelYtrial3) ;

finalIDEOaccZtrial3 = diff (filteredIDEOvelZtrial3) ;

%create time domain & plot

acceltimetrial3 = (1/Fs) * (0:(length (finalIDEOaccXtrial3)-1)) ;

% combine variables into 3D matrices

normalxforcetrial3 = xforcetrial3 ((HSsampletrial3+2):TOsampletrial3) ;

normalyforcetrial3 = yforcetrial3 ((HSsampletrial3+2):TOsampletrial3) ;

normalzforcetrial3 = zforcetrial3 ((HSsampletrial3+2):TOsampletrial3) ;

IDEOacctrial3 = [finalIDEOaccXtrial3 finalIDEOaccYtrial3 finalIDEOaccZtrial3] ; %[m/s^2]

GRFtrial3 = [normalxforcetrial3 normalyforcetrial3 normalzforcetrial3] ; % [N]

%use IDEOacc with GRF to calculate knee reaction force

KRF3Dtrial3 = (msegment.*IDEOacctrial3) - GRFtrial3 - (msegment.*gravity) ;

%take out components of KRF3D

krXtrial3 = KRF3Dtrial3(:,1) ;

krYtrial3 = KRF3Dtrial3(:,2) ;

krZtrial3 = KRF3Dtrial3(:,3) ;

kneetimetrial3 = acceltimetrial3 ;

%% Import files for coincident GRF (txt) and motion (csv) of all trials

fnameforcetrial4 = "IDEOtrial4txt.txt" ;

rawforcedatatrial4 = importdata (fnameforcetrial4) ;

fnamemotiontrial4 = "IDEOtrial4csv.csv" ;

rawmotiondatatrial4 = csvread (fnamemotiontrial4) ;

% Stance times

% trial 1: 0.525 , 1.425

% trial 2: 0.742 , 1.708

% trial 3: 0.575 , 1.567

% trial 4: 0.617 , 1.525

% trial 5: 0.483 , 1.433

% INPUTS for this type of trial

HStimetrial4 = (0.617) ; % input [s]

TOtimetrial4 = (1.525) ; % input [s]

% extracting column data for force *check y force

xforcetrial4 = rawforcedatatrial4(:,1) ;

yforcetrial4 = rawforcedatatrial4(:,2) ;
```

```
% positive should be right side (lateral for right foot)
zforcetrial4 = rawforcedatatrial4(:,3) ;


% set sampling rate and inital time domain for raw data


forcesamplestrial4 = length(rawforcedatatrial4) ;
forcetimetrial4 = (0:(1/Fs):((forcesamplestrial4/Fs)-(1/Fs))) ;


% plot raw force data over time in three dimensions




% extract sample numbers from motion data to create time domain
% delete the column with non numericals


motionsamplestrial4 = rawmotiondatatrial4(:,4);
motiontimetrial4 = (0:(1/Fs):(((length(motionsamplestrial4)/2)/Fs)-(1/Fs))) ;


% organize data by sensor/hub with if statements


hub1sensor1trial4 = zeros((floor(length(motionsamplestrial4)/2)), 6) ;


j = 1;
for i = 1 : (length(rawmotiondatatrial4))
    if (rawmotiondatatrial4(i,2) == 1) && (rawmotiondatatrial4(i,3) == 1)
        hub1sensor1trial4(j, :) = rawmotiondatatrial4(i, 5:10);
        j = j + 1;
    end
end


hub1sensor2trial4 = zeros((floor(length(motionsamplestrial4)/2)), 6) ;


j = 1;
for i = 1 : (length(rawmotiondatatrial4))
    if (rawmotiondatatrial4(i,2) == 1) && (rawmotiondatatrial4(i,3) == 2)
        hub1sensor2trial4(j, :) = rawmotiondatatrial4(i, 5:10);
        j = j + 1;
    end
end


hub1sensor3trial4 = zeros((floor(length(motionsamplestrial4)/2)), 6) ;


j = 1;
for i = 1 : (length(rawmotiondatatrial4))
    if (rawmotiondatatrial4(i,2) == 1) && (rawmotiondatatrial4(i,3) == 3)
        hub1sensor3trial4(j, :) = rawmotiondatatrial4(i, 5:10);
        j = j + 1;
    end
end


hub2sensor1trial4 = zeros((floor(length(motionsamplestrial4)/2)), 6) ;


j = 1;
for i = 1 : (length(rawmotiondatatrial4))
    if (rawmotiondatatrial4(i,2) == 2) && (rawmotiondatatrial4(i,3) == 1)
        hub2sensor1trial4(j, :) = rawmotiondatatrial4(i, 5:10);
```

```
        j = j + 1;
    end
end

hub2sensor2trial4 = zeros((floor(length
(motionsamplestrial4)/2)), 6) ;

j = 1;
for i = 1 : (length
(rawmotiondatatrial4))
    if (rawmotiondatatrial4(i,2) == 2)
&& (rawmotiondatatrial4(i,3) == 2)
        hub2sensor2trial4(j, :) = rawmo-
tiondatatrial4(i, 5:10);
        j = j + 1;
    end
end

% Sensor locations (lateral)
% 1,1 = COM of IDEO
% 1,2 = femoral condyles at knee
% 1,3 = heel
% 2,1 = back of struts - not lateral
% 2,2 = toes - metatarsals

% here we can fix the sign of each set
of data if we want, and correct the
% value of the displacement data for the
segment using the COM of the
% total segment

%extract displacement data , use offsets
for COM

IDEOdispXtrial4 =
intom*hub1sensor1trial4(:,1) - Ithick ;
IDEOdispYtrial4 =
intom*hub1sensor1trial4(:,2) - Iwidth ;
IDEOdispZtrial4 =
intom*hub1sensor1trial4(:,3) - Ilength ;

% plot raw displacement data over time
in three dimensions

%Normalized Data inputs for full step
with force plate

HSsampletrial4 = floor(HStimetrial4*Fs)
- 2 ; %accounts for double differentia-
tion
TOsampletrial4 = floor
(TOtimetrial4*Fs) ;
normalsampleswrongtrial4 = (0:
(TOsampletrial4 - HSsampletrial4)) ;
normalsamplestrial4 = transpose
(normalsampleswrongtrial4) ;

%make the displacement data normalized

normalIDEOdispXtrial4 = IDEOdispXtrial4
(HSsampletrial4:TOsampletrial4) ;
normalIDEOdispYtrial4 = IDEOdispYtrial4
(HSsampletrial4:TOsampletrial4) ;
normalIDEOdispZtrial4 = IDEOdispZtrial4
(HSsampletrial4:TOsampletrial4) ;

%FILTER NORMALIZED DISPLACEMENT DATA,
THEN DIFFERENTIATE IT

%design fifth order butterworth filter

%filter foot and leg displacement data &
plot
```

```
filteredIDEOdispXtrial4 = filtfilt
(filter1, n, normalIDEOdispXtrial4) ;

filteredIDEOdispYtrial4 = filtfilt
(filter1, n, normalIDEOdispYtrial4) ;

filteredIDEOdispZtrial4 = filtfilt
(filter1, n, normalIDEOdispZtrial4) ;

%use normalsamples time domain and plot
filtered displacement data

%differentiate these displacement values
to get velocity

filteredIDEOvelXtrial4 = diff
(filteredIDEOdispXtrial4) ;

filteredIDEOvelYtrial4 = diff
(filteredIDEOdispYtrial4) ;

filteredIDEOvelZtrial4 = diff
(filteredIDEOdispZtrial4) ;

% make time domain and plot

veltimetrial4 = 0:(1/Fs):((1/Fs)*length
(filteredIDEOvelXtrial4) - (1/Fs)) ;

%differentiate these velocity values to
get accleration

finalIDEOaccXtrial4 = diff
(filteredIDEOvelXtrial4) ;

finalIDEOaccYtrial4 = diff
(filteredIDEOvelYtrial4) ;

finalIDEOaccZtrial4 = diff
(filteredIDEOvelZtrial4) ;

%create time domain & plot

acceltimetrial4 = (1/Fs) * (0:(length
(finalIDEOaccXtrial4)-1)) ;

% combine variables into 3D matrices

normalxforcetrial4 = xforcetrial4
((HSsampletrial4+2):TOsampletrial4) ;

normalyforcetrial4 = yforcetrial4
((HSsampletrial4+2):TOsampletrial4) ;

normalzforcetrial4 = zforcetrial4
((HSsampletrial4+2):TOsampletrial4) ;

IDEOacctrial4 = [finalIDEOaccXtrial4 fi-
nalIDEOaccYtrial4 finalIDEOaccZtrial4] ;
%[m/s^2]

GRFtrial4 = [normalxforcetrial4 nor-
malyforcetrial4 normalzforcetrial4] ; %
[N]

%use IDEOacc with GRF to calculate knee
reaction force

KRF3Dtrial4 = (msegment.*IDEOacctrial4)
- GRFtrial4 - (msegment.*gravity) ;

%take out components of KRF3D

krXtrial4 = KRF3Dtrial4(:,1) ;

krYtrial4 = KRF3Dtrial4(:,2) ;

krZtrial4 = KRF3Dtrial4(:,3) ;

kneetimetrial4 = acceltimetrial4 ;

%% Import files for coincident GRF (txt)
and motion (csv) of all trials

fnameforcetrial5 = "IDEOtrial5txt.txt" ;
```

```
rawforcedatatrial5 = importdata
(fnameforcetrial5) ;

fnamemotiontrial5 =
"IDEOtrial5csv.csv" ;
rawmotiondatatrial5 = csvread
(fnamemotiontrial5) ;

% Stance times
% trial 1: 0.525 , 1.425
% trial 2: 0.742 , 1.708
% trial 3: 0.575 , 1.567
% trial 4: 0.617 , 1.525
% trial 5: 0.483 , 1.433

% INPUTS for this type of trial

HStimetrial5 = (0.483) ; % input [s]
TOtimetrial5 = (1.433) ; % input [s]

% extracting column data for force
*check y force

xforcetrial5 = rawforcedatatrial5(:,1) ;
yforcetrial5 = rawforcedatatrial5(:,2) ;
% positive should be right side (lateral
for right foot)
zforcetrial5 = rawforcedatatrial5(:,3) ;

% set sampling rate and inital time do-
main for raw data

forcesamplestrial5 = length
(rawforcedatatrial5) ;
forcetimetrial5 = (0:(1/Fs):
((forcesamplestrial5/Fs)-(1/Fs))) ;

% plot raw force data over time in three
dimensions

% extract sample numbers from motion da-
ta to create time domain
% delete the column with non numericals

motionsamplestrial5 = rawmotiondata-
trial5(:,4);
motiontimetrial5 = (0:(1/Fs):(((length
(motionsamplestrial5)/2)/Fs)-(1/Fs))) ;

% organize data by sensor/hub with if
statements

hub1sensor1trial5 = zeros((floor(length
(motionsamplestrial5)/2)), 6) ;

j = 1;
for i = 1 : (length
(rawmotiondatatrial5))
    if (rawmotiondatatrial5(i,2) == 1)
&& (rawmotiondatatrial5(i,3) == 1)
        hub1sensor1trial5(j, :) = rawmo-
tiondatatrial5(i, 5:10);
        j = j + 1;
    end
end

hub1sensor2trial5 = zeros((floor(length
(motionsamplestrial5)/2)), 6) ;

j = 1;
for i = 1 : (length
(rawmotiondatatrial5))
    if (rawmotiondatatrial5(i,2) == 1)
&& (rawmotiondatatrial5(i,3) == 2)
        hub1sensor2trial5(j, :) = rawmo-
tiondatatrial5(i, 5:10);
        j = j + 1;
    end
```

```
end


hub1sensor3trial5 = zeros((floor(length
(motionsamplestrial5)/2)), 6) ;


j = 1;
for i = 1 : (length
(rawmotiondatatrial5))
    if (rawmotiondatatrial5(i,2) == 1)
&& (rawmotiondatatrial5(i,3) == 3)
        hub1sensor3trial5(j, :) =
rawmotiondatatrial5(i, 5:10);
        j = j + 1;
    end
end


hub2sensor1trial5 = zeros((floor(length
(motionsamplestrial5)/2)), 6) ;


j = 1;
for i = 1 : (length
(rawmotiondatatrial5))
    if (rawmotiondatatrial5(i,2) == 2)
&& (rawmotiondatatrial5(i,3) == 1)
        hub2sensor1trial5(j, :) =
rawmotiondatatrial5(i, 5:10);
        j = j + 1;
    end
end


hub2sensor2trial5 = zeros((floor(length
(motionsamplestrial5)/2)), 6) ;


j = 1;
for i = 1 : (length
(rawmotiondatatrial5))
    if (rawmotiondatatrial5(i,2) == 2)
&& (rawmotiondatatrial5(i,3) == 2)
        hub2sensor2trial5(j, :) =
rawmotiondatatrial5(i, 5:10);
        j = j + 1;
    end
end


% Sensor locations (lateral)
% 1,1 = COM of IDEO
% 1,2 = femoral condyles at knee
% 1,3 = heel
% 2,1 = back of struts - not lateral
% 2,2 = toes - metatarsals


% here we can fix the sign of each set
of data if we want, and correct the
% value of the displacement data for
the segment using the COM of the
% total segment


%extract displacement data , use off-
sets for COM


IDEOdispXtrial5 =
intom*hub1sensor1trial5(:,1) - Ithick ;
IDEOdispYtrial5 =
intom*hub1sensor1trial5(:,2) - Iwidth ;
IDEOdispZtrial5 =
intom*hub1sensor1trial5(:,3) -
Ilength ;


% plot raw displacement data over time
in three dimensions


%Normalized Data inputs for full step
with force plate


HSsampletrial5 = floor(HStimetrial5*Fs)
```

```
- 2 ; %accounts for double differentiation

TOsampletrial5 = floor
(TOtimetrial5*Fs) ;

normalsampleswrongtrial5 = (0:
(TOsampletrial5 - HSsampletrial5)) ;

normalsamplestrial5 = transpose
(normalsampleswrongtrial5) ;

%make the displacement data normalized

normalIDEOdispXtrial5 = IDEOdispXtrial5
(HSsampletrial5:TOsampletrial5) ;

normalIDEOdispYtrial5 = IDEOdispYtrial5
(HSsampletrial5:TOsampletrial5) ;

normalIDEOdispZtrial5 = IDEOdispZtrial5
(HSsampletrial5:TOsampletrial5) ;

%FILTER NORMALIZED DISPLACEMENT DATA,
THEN DIFFERENTIATE IT

%design fifth order butterworth filter

%filter foot and leg displacement data
& plot

filteredIDEOdispXtrial5 = filtfilt
(filter1, n, normalIDEOdispXtrial5) ;

filteredIDEOdispYtrial5 = filtfilt
(filter1, n, normalIDEOdispYtrial5) ;

filteredIDEOdispZtrial5 = filtfilt
(filter1, n, normalIDEOdispZtrial5) ;

%use normalsamples time domain and plot
filtered displacement data

%differentiate these displacement values to get velocity

filteredIDEOvelXtrial5 = diff
(filteredIDEOdispXtrial5) ;

filteredIDEOvelYtrial5 = diff
(filteredIDEOdispYtrial5) ;

filteredIDEOvelZtrial5 = diff
(filteredIDEOdispZtrial5) ;

% make time domain and plot

veltimetrial5 = 0:(1/Fs):((1/Fs)*length
(filteredIDEOvelXtrial5) - (1/Fs)) ;

%differentiate these velocity values to
get accleration

finalIDEOaccXtrial5 = diff
(filteredIDEOvelXtrial5) ;

finalIDEOaccYtrial5 = diff
(filteredIDEOvelYtrial5) ;

finalIDEOaccZtrial5 = diff
(filteredIDEOvelZtrial5) ;

%create time domain & plot

acceltimetrial5 = (1/Fs) * (0:(length
(finalIDEOaccXtrial5)-1)) ;

% combine variables into 3D matrices

normalxforcetrial5 = xforcetrial5
((HSsampletrial5+2):TOsampletrial5) ;

normalyforcetrial5 = yforcetrial5
((HSsampletrial5+2):TOsampletrial5) ;

normalzforcetrial5 = zforcetrial5
((HSsampletrial5+2):TOsampletrial5) ;
```

```
IDEOacctrial5 = [finalIDEOaccXtrial5 finalIDEOaccYtrial5 finalIDEOaccZtrial5] ; %[m/s^2]

GRFtrial5 = [normalxforcetrial5 normalyforcetrial5 normalzforcetrial5] ; % [N]

%use IDEOacc with GRF to calculate knee reaction force

KRF3Dtrial5 = (msegment.*IDEOacctrial5) - GRFtrial5 - (msegment.*gravity) ;

%take out components of KRF3D

krXtrial5 = KRF3Dtrial5(:,1) ;
krYtrial5 = KRF3Dtrial5(:,2) ;
krZtrial5 = KRF3Dtrial5(:,3) ;

kneetimetrial5 = acceltimetrial5 ;

%% PLOTS FOR ALL TRIALS COMBINED

%Reaction Force over percent of Stance Phase

%create percent time domain for every trial

percentztrial1 = kneetimetrial1(length(kneetimetrial1)) ;

for i = 1 : (length(kneetimetrial1))
    stpercenttrial1(i) = kneetimetrial1(i) * (100 / percentztrial1)  ;
end

percentztrial2 = kneetimetrial2(length(kneetimetrial2)) ;

for i = 1 : (length(kneetimetrial2))
    stpercenttrial2(i) = kneetimetrial2(i) * (100 / percentztrial2)  ;
end

percentztrial3 = kneetimetrial3(length(kneetimetrial3)) ;

for i = 1 : (length(kneetimetrial3))
    stpercenttrial3(i) = kneetimetrial3(i) * (100 / percentztrial3)  ;
end

percentztrial4 = kneetimetrial4(length(kneetimetrial4)) ;

for i = 1 : (length(kneetimetrial4))
    stpercenttrial4(i) = kneetimetrial4(i) * (100 / percentztrial4)  ;
end

percentztrial5 = kneetimetrial5(length(kneetimetrial5)) ;

for i = 1 : (length(kneetimetrial5))
    stpercenttrial5(i) = kneetimetrial5(i) * (100 / percentztrial5)  ;
end

%Knee Reaction Force Z

figure
plot(stpercenttrial1, krZtrial1)
xlim([0 100])
ylim([-1200 0])
title('IDEO Knee Reaction Force - Z direction')
```

```
xlabel('% of Stance Phase')
ylabel('Force [N]')

hold on

plot(stpercenttrial2, krZtrial2)
plot(stpercenttrial3, krZtrial3)
plot(stpercenttrial4, krZtrial4)
plot(stpercenttrial5, krZtrial5)

%Knee Reaction Force X

figure
plot(stpercenttrial1, krXtrial1)
xlim([0 100])
ylim([-200 150])
title('IDEO Knee Reaction Force - X di-
rection')
xlabel('% of Stance Phase')
ylabel('Force [N]')

hold on

plot(stpercenttrial2, krXtrial2)
plot(stpercenttrial3, krXtrial3)
plot(stpercenttrial4, krXtrial4)
plot(stpercenttrial5, krXtrial5)

%Knee Reaction Force Y

figure
plot(stpercenttrial1, krYtrial1)
xlim([0 100])
ylim([-50 100])
title('IDEO Knee Reaction Force - Y di-
rection')
xlabel('% of Stance Phase')
ylabel('Force [N]')

hold on

plot(stpercenttrial2, krYtrial2)
plot(stpercenttrial3, krYtrial3)
plot(stpercenttrial4, krYtrial4)
plot(stpercenttrial5, krYtrial5)

%Ground Reaction Force

% GRF x

figure
plot(stpercenttrial1, normalxforce-
trial1)
xlim([0 100])
ylim([-150 200])
title("Ground Reaction Force - IDEO - X
Direction")
xlabel("% of Stance Phase")
ylabel("Force [N]")

hold on

plot(stpercenttrial2, normalxforce-
trial2)
plot(stpercenttrial3, normalxforce-
trial3)
plot(stpercenttrial4, normalxforce-
trial4)
plot(stpercenttrial5, normalxforce-
trial5)

% GRF y

figure
plot(stpercenttrial1, normalyforce-
trial1)
xlim([0 100])
```

```
ylim([-100 50])

title("Ground Reaction Force - IDEO - Y Direction")

xlabel("% of Stance Phase")

ylabel("Force [N]")


hold on


plot(stpercenttrial2, normalyforce-trial2)

plot(stpercenttrial3, normalyforce-trial3)

plot(stpercenttrial4, normalyforce-trial4)

plot(stpercenttrial5, normalyforce-trial5)


% GRF z


figure

plot(stpercenttrial1, normalzforce-trial1)

xlim([0 100])

ylim([0 1000])

title("Ground Reaction Force - IDEO - Z Direction")

xlabel("% of Stance Phase")

ylabel("Force [N]")


hold on


plot(stpercenttrial2, normalzforce-trial2)

plot(stpercenttrial3, normalzforce-trial3)

plot(stpercenttrial4, normalzforce-trial4)

plot(stpercenttrial5, normalzforce-trial5)


%% Energy Stored


% input data file for force over extension


energyfname = "Specimen_RawData_2_new.csv" ; %file format is time (s), extension (mm), load (N)

fdfile = csvread(energyfname) ;


% extract columns


eforce = fdfile(:,3) ;     % [N]

extension = fdfile(:,2) ; % [mm]


fixedsample = find(eforce == max(eforce)) ;

fixedsample = find(eforce == max(eforce)) - 0.02*fixedsample ;


fixedeforce = eforce(1:fixedsample) ; % [N]

fixedextension = extension(1:fixedsample) ./ 1000; %[mm] to [m]


maxextension = max(fixedextension) ;

maxeforce = max(fixedeforce) ;


pextension = extension(1:203) ;

peforce = eforce(1:203) ;


figure

plot(pextension, peforce)

title("Force vs Deflection for Carbon Fiber")

xlabel("Displacement [mm]")

ylabel("Applied Force [N]")


% calculate stress vs strain & plot
```

```
% the area is calculated from actual dimensions of IDEO struts

rodlength = 0.120 ;   %[m] support span
densrod = 1550 ;      %[kg/m^3]
drod = 0.005 ;        %[m]
rrod = 0.5*drod ;     %[m]

vrod = pi*rodlength*(rrod^2) ; %[m^3]
mrod = vrod * densrod ; %[kg]

Irod = (pi/4) * (rrod^4) ;
modulus = (( maxeforce * (rodlength^3)) / (48*maxextension*Irod)) ; %[Pa]

strain = 0:0.0005:0.02 ;
stress = modulus .* strain ;

% IDEO dimensions

strutR = 0.0055 ; %[m]
strutAx = (strutR^2) * pi ; %[m^2]
strutL = 0.143 ; %[m]

% Moment calculations for IDEO

yforcemomenttrial1 = rawforcedatatrial1(:,5) ;
yforcemomenttrial2 = rawforcedatatrial2(:,5) ;
yforcemomenttrial3 = rawforcedatatrial3(:,5) ;
yforcemomenttrial4 = rawforcedatatrial4(:,5) ;
yforcemomenttrial5 = rawforcedatatrial5(:,5) ;

normalyforcemomenttrial1 = yforcemomenttrial1((HSsampletrial1+2):TOsampletrial1) ;
normalyforcemomenttrial2 = yforcemomenttrial2((HSsampletrial2+2):TOsampletrial2) ;
normalyforcemomenttrial3 = yforcemomenttrial3((HSsampletrial3+2):TOsampletrial3) ;
normalyforcemomenttrial4 = yforcemomenttrial4((HSsampletrial4+2):TOsampletrial4) ;
normalyforcemomenttrial5 = yforcemomenttrial5((HSsampletrial5+2):TOsampletrial5) ;

figure
plot(acceltimetrial1, normalyforcemomenttrial1)
title("Ground Reaction Moments about Y axis")
xlabel("Time [s]")
ylabel("Moment [N*m]")

hold on

plot(acceltimetrial2, normalyforcemomenttrial2)
plot(acceltimetrial3, normalyforcemomenttrial3)
plot(acceltimetrial4, normalyforcemomenttrial4)
plot(acceltimetrial5, normalyforcemomenttrial5)

Mgrftrial1 = normalyforcemomenttrial1 ;
Mgrftrial2 = normalyforcemomenttrial2 ;
Mgrftrial3 = normalyforcemomenttrial3 ;
Mgrftrial4 = normalyforcemomenttrial4 ;
Mgrftrial5 = normalyforcemomenttrial5 ;

Mgrftrial1sample = find(normalyforcemomenttrial1 == max(Mgrftrial1)) ;
Mgrftrial2sample = find(normalyforcemomenttrial2 == max
```

```
(Mgrftrial2)) ;
Mgrftrial3sample = find
(normalyforcemomenttrial3 == max
(Mgrftrial3)) ;
Mgrftrial4sample = find
(normalyforcemomenttrial4 == max
(Mgrftrial4)) ;
Mgrftrial5sample = find
(normalyforcemomenttrial5 == max
(Mgrftrial5)) ;


% Strut orientation


strutzrottrial1 = hub2sensor1trial1
(:,6) ;
strutzrottrial2 = hub2sensor1trial2
(:,6) ;
strutzrottrial3 = hub2sensor1trial3
(:,6) ;
strutzrottrial4 = hub2sensor1trial4
(:,6) ;
strutzrottrial5 = hub2sensor1trial5
(:,6) ;


normalstrutzrottrial1 = strutzrottrial1
((HSsampletrial1+2):TOsampletrial1) ;
normalstrutzrottrial2 = strutzrottrial2
((HSsampletrial2+2):TOsampletrial2) ;
normalstrutzrottrial3 = strutzrottrial3
((HSsampletrial3+2):TOsampletrial3) ;
normalstrutzrottrial4 = strutzrottrial4
((HSsampletrial4+2):TOsampletrial4) ;
normalstrutzrottrial5 = strutzrottrial5
((HSsampletrial5+2):TOsampletrial5) ;


figure
plot(acceltimetrial1, normalstrutzrot-
trial1)
title("Strut Orientation about z axis")
xlabel("Time [s]")
ylabel("Degrees")


hold on


plot(acceltimetrial2, normalstrutzrot-
trial2)
plot(acceltimetrial3, normalstrutzrot-
trial3)
plot(acceltimetrial4, normalstrutzrot-
trial4)
plot(acceltimetrial5, normalstrutzrot-
trial5)


% Force Transformation Matrix


strutthetaxtrial1 = normalstrutzrot-
trial1 +180 ;


lambdaxtrial1 = cosd
(strutthetaxtrial1) ;
lambdaytrial1 = sind
(strutthetaxtrial1) ;


lxtrial1 = zeros(length
(strutthetaxtrial1)) ;
for i = 1:length(strutthetaxtrial1)
    lxtrial1(:,i) = lambdaxtrial1 ;
end


lytrial1 = zeros(length
(strutthetaxtrial1)) ;
for i = 1:length(strutthetaxtrial1)
    lytrial1(:,i) = lambdaytrial1 ;
end


zerotrial1 = zeros(length
(strutthetaxtrial1)) ;
onetrial1 = ones(length
(strutthetaxtrial1)) ;


Transformtrial1 = [lxtrial1 -lytrial1
zerotrial1 zerotrial1 zerotrial1 ze-
rotrial1 ;
```

```
            lytrial1 lxtrial1 ze-
rotrial1 zerotrial1 zerotrial1 ze-
rotrial1 ;
            zerotrial1 zerotrial1 one-
trial1 zerotrial1 zerotrial1 ze-
rotrial1;
            zerotrial1 zerotrial1 ze-
rotrial1 lxtrial1 -lytrial1 ze-
rotrial1 ;
            zerotrial1 zerotrial1 ze-
rotrial1 lytrial1 lxtrial1 zerotrial1 ;
            zerotrial1 zerotrial1 ze-
rotrial1 zerotrial1 zerotrial1 one-
trial1 ] ;


strutthetaxtrial2 = normalstrutzrot-
trial2 +180 ;


lambdaxtrial2 = cosd
(strutthetaxtrial2) ;
lambdaytrial2 = sind
(strutthetaxtrial2) ;


lxtrial2 = zeros(length
(strutthetaxtrial2)) ;
for i = 1:length(strutthetaxtrial2)
    lxtrial2(:,i) = lambdaxtrial2 ;
end
lytrial2 = zeros(length
(strutthetaxtrial2)) ;
for i = 1:length(strutthetaxtrial2)
    lytrial2(:,i) = lambdaytrial2 ;
end


zerotrial2 = zeros(length
(strutthetaxtrial2)) ;
onetrial2 = ones(length
(strutthetaxtrial2)) ;


Transformtrial2 = [lxtrial2 -lytrial2
zerotrial2 zerotrial2 zerotrial2 ze-
rotrial2 ;
            lytrial2 lxtrial2 ze-
rotrial2 zerotrial2 zerotrial2 ze-
rotrial2 ;
            zerotrial2 zerotrial2 one-
trial2 zerotrial2 zerotrial2 ze-
rotrial2;
            zerotrial2 zerotrial2 ze-
rotrial2 lxtrial2 -lytrial2 ze-
rotrial2 ;
            zerotrial2 zerotrial2 ze-
rotrial2 lytrial2 lxtrial2 zerotrial2 ;
            zerotrial2 zerotrial2 ze-
rotrial2 zerotrial2 zerotrial2 one-
trial2 ] ;
strutthetaxtrial3 = normalstrutzrot-
trial3 +180 ;


lambdaxtrial3 = cosd
(strutthetaxtrial3) ;
lambdaytrial3 = sind
(strutthetaxtrial3) ;


lxtrial3 = zeros(length
(strutthetaxtrial3)) ;
for i = 1:length(strutthetaxtrial3)
    lxtrial3(:,i) = lambdaxtrial3 ;
end


lytrial3 = zeros(length
(strutthetaxtrial3)) ;
for i = 1:length(strutthetaxtrial3)
    lytrial3(:,i) = lambdaytrial3 ;
end


zerotrial3 = zeros(length
(strutthetaxtrial3)) ;
onetrial3 = ones(length
(strutthetaxtrial3)) ;


Transformtrial3 = [lxtrial3 -lytrial3
zerotrial3 zerotrial3 zerotrial3 ze-
rotrial3 ;
```

```
             lytrial3 lxtrial3 ze-
rotrial3 zerotrial3 zerotrial3 ze-
rotrial3 ;
             zerotrial3 zerotrial3 one-
trial3 zerotrial3 zerotrial3 ze-
rotrial3;
             zerotrial3 zerotrial3 ze-
rotrial3 lxtrial3 -lytrial3 ze-
rotrial3 ;
             zerotrial3 zerotrial3 ze-
rotrial3 lytrial3 lxtrial3 zerotrial3 ;
             zerotrial3 zerotrial3 ze-
rotrial3 zerotrial3 zerotrial3 one-
trial3 ] ;


strutthetaxtrial4 = normalstrutzrot-
trial4 +180 ;


lambdaxtrial4 = cosd
(strutthetaxtrial4) ;
lambdaytrial4 = sind
(strutthetaxtrial4) ;


lxtrial4 = zeros(length
(strutthetaxtrial4)) ;
for i = 1:length(strutthetaxtrial4)
    lxtrial4(:,i) = lambdaxtrial4 ;
end
lytrial4 = zeros(length
(strutthetaxtrial4)) ;
for i = 1:length(strutthetaxtrial4)
    lytrial4(:,i) = lambdaytrial4 ;
end


zerotrial4 = zeros(length
(strutthetaxtrial4)) ;
onetrial4 = ones(length
(strutthetaxtrial4)) ;


Transformtrial4 = [lxtrial4 -lytrial4
zerotrial4 zerotrial4 zerotrial4 ze-
rotrial4 ;
             lytrial4 lxtrial4 ze-
rotrial4 zerotrial4 zerotrial4 ze-
rotrial4 ;
             zerotrial4 zerotrial4 one-
trial4 zerotrial4 zerotrial4 ze-
rotrial4;
             zerotrial4 zerotrial4 ze-
rotrial4 lxtrial4 -lytrial4 ze-
rotrial4 ;
             zerotrial4 zerotrial4 ze-
rotrial4 lytrial4 lxtrial4 zerotrial4 ;
             zerotrial4 zerotrial4 ze-
rotrial4 zerotrial4 zerotrial4 one-
trial4 ] ;
strutthetaxtrial5 = normalstrutzrot-
trial5 +180 ;


lambdaxtrial5 = cosd
(strutthetaxtrial5) ;
lambdaytrial5 = sind
(strutthetaxtrial5) ;


lxtrial5 = zeros(length
(strutthetaxtrial5)) ;
for i = 1:length(strutthetaxtrial5)
    lxtrial5(:,i) = lambdaxtrial5 ;
end


lytrial5 = zeros(length
(strutthetaxtrial5)) ;
for i = 1:length(strutthetaxtrial5)
    lytrial5(:,i) = lambdaytrial5 ;
end


zerotrial5 = zeros(length
(strutthetaxtrial5)) ;
onetrial5 = ones(length
(strutthetaxtrial5)) ;


Transformtrial5 = [lxtrial5 -lytrial5
zerotrial5 zerotrial5 zerotrial5 ze-
```

```
rotrial5 ;
             lytrial5 lxtrial5 ze-
rotrial5 zerotrial5 zerotrial5 ze-
rotrial5 ;
             zerotrial5 zerotrial5 one-
trial5 zerotrial5 zerotrial5 zerotrial5;
             zerotrial5 zerotrial5 ze-
rotrial5 lxtrial5 -lytrial5 zerotrial5 ;
             zerotrial5 zerotrial5 ze-
rotrial5 lytrial5 lxtrial5 zerotrial5 ;
             zerotrial5 zerotrial5 ze-
rotrial5 zerotrial5 zerotrial5 one-
trial5 ] ;


Fxktrial1 = zeros(length(krXtrial1)) ;
for i = 1:length(krXtrial1)
    Fxktrial1(:,i)= krXtrial1 ;
end


Fxktrial2 = zeros(length(krXtrial2)) ;
for i = 1:length(krXtrial2)
    Fxktrial2(:,i)= krXtrial2 ;
end


Fxktrial3 = zeros(length(krXtrial3)) ;
for i = 1:length(krXtrial3)
    Fxktrial3(:,i)= krXtrial3 ;
end


Fxktrial4 = zeros(length(krXtrial4)) ;
for i = 1:length(krXtrial4)
    Fxktrial4(:,i)= krXtrial4 ;
end


Fxktrial5 = zeros(length(krXtrial5)) ;
for i = 1:length(krXtrial5)
    Fxktrial5(:,i)= krXtrial5 ;
end


oldforcetrial1 = [zerotrial1 ; ze-
rotrial1 ; zerotrial1 ; Fxktrial1 ; ze-
rotrial1 ; zerotrial1] ;
oldforcetrial2 = [zerotrial2 ; ze-
rotrial2 ; zerotrial2 ; Fxktrial2 ; ze-
rotrial2 ; zerotrial2] ;
oldforcetrial3 = [zerotrial3 ; ze-
rotrial3 ; zerotrial3 ; Fxktrial3 ; ze-
rotrial3 ; zerotrial3] ;
oldforcetrial4 = [zerotrial4 ; ze-
rotrial4 ; zerotrial4 ; Fxktrial4 ; ze-
rotrial4 ; zerotrial4] ;
oldforcetrial5 = [zerotrial5 ; ze-
rotrial5 ; zerotrial5 ; Fxktrial5 ; ze-
rotrial5 ; zerotrial5] ;

toldforcetrial1 = transpose
(oldforcetrial1) ;
toldforcetrial2 = transpose
(oldforcetrial2) ;
toldforcetrial3 = transpose
(oldforcetrial3) ;
toldforcetrial4 = transpose
(oldforcetrial4) ;
toldforcetrial5 = transpose
(oldforcetrial5) ;


newforcetrial1 = toldforcetrial1 *
Transformtrial1 ;
newforcetrial2 = toldforcetrial2 *
Transformtrial2 ;
newforcetrial3 = toldforcetrial3 *
Transformtrial3 ;
newforcetrial4 = toldforcetrial4 *
Transformtrial4 ;
newforcetrial5 = toldforcetrial5 *
Transformtrial5 ;


valnewforcetrial1 = find
(newforcetrial1) ;
valnewforcetrial2 = find
(newforcetrial2) ;
valnewforcetrial3 = find
(newforcetrial3) ;
valnewforcetrial4 = find
```

```
(newforcetrial4) ;
valnewforcetrial5 = find
(newforcetrial5) ;


Fxktransformedtrial1 = newforcetrial1
(valnewforcetrial1(1)) ;
Fxktransformedtrial2 = newforcetrial2
(valnewforcetrial2(1)) ;
Fxktransformedtrial3 = newforcetrial3
(valnewforcetrial3(1)) ;
Fxktransformedtrial4 = newforcetrial4
(valnewforcetrial4(1)) ;
Fxktransformedtrial5 = newforcetrial5
(valnewforcetrial5(1)) ;


Fxktransformedtrial1 = krXtrial1.*cos
(strutthetaxtrial1) + krXtrial1.*sin
(strutthetaxtrial1) ;
Fxktransformedtrial2 = krXtrial2.*cos
(strutthetaxtrial2) + krXtrial2.*sin
(strutthetaxtrial2) ;
Fxktransformedtrial3 = krXtrial3.*cos
(strutthetaxtrial3) + krXtrial3.*sin
(strutthetaxtrial3) ;
Fxktransformedtrial4 = krXtrial4.*cos
(strutthetaxtrial4) + krXtrial4.*sin
(strutthetaxtrial4) ;
Fxktransformedtrial5 = krXtrial5.*cos
(strutthetaxtrial5) + krXtrial5.*sin
(strutthetaxtrial5) ;


% Moment FBD - Mkrf


Mkrftrial1 = Mgrftrial1 + (strutL .*
Fxktransformedtrial1) ;
Mkrftrial2 = Mgrftrial2 + (strutL .*
Fxktransformedtrial2) ;
Mkrftrial3 = Mgrftrial3 + (strutL .*
Fxktransformedtrial3) ;
Mkrftrial4 = Mgrftrial4 + (strutL .*
Fxktransformedtrial4) ;
Mkrftrial5 = Mgrftrial5 + (strutL .*
Fxktransformedtrial5) ;


% Total Moment Calculation


I = 0.5 * mrod * (rrod^2) ;
ytotal = strutR ; % strut radius


Mtotaltrial1 = Mgrftrial1 + Mkrftrial1
+ (strutL .* Fxktransformedtrial1)  ; %
[N*m]
Mtotaltrial2 = Mgrftrial2 + Mkrftrial2
+ (strutL .* Fxktransformedtrial2)  ; %
[N*m]
Mtotaltrial3 = Mgrftrial3 + Mkrftrial3
+ (strutL .* Fxktransformedtrial3)  ; %
[N*m]
Mtotaltrial4 = Mgrftrial4 + Mkrftrial4
+ (strutL .* Fxktransformedtrial4)  ; %
[N*m]
Mtotaltrial5 = Mgrftrial5 + Mkrftrial5
+ (strutL .* Fxktransformedtrial5)  ; %
[N*m]


figure
plot(stpercenttrial1, Mtotaltrial1)
title("Bending Moment on Struts During
Stance Phase")
xlabel("Percent of Stance Phase")
ylabel("Moment [N*m]")


hold on


plot(stpercenttrial2, Mtotaltrial2)
plot(stpercenttrial3, Mtotaltrial3)
plot(stpercenttrial4, Mtotaltrial4)
plot(stpercenttrial5, Mtotaltrial5)


bendstresstrial1 = abs(Mtotaltrial1 *
ytotal / I) ; %[Pa]
bendstresstrial2 = abs(Mtotaltrial2 *
ytotal / I) ; %[Pa]
bendstresstrial3 = abs(Mtotaltrial3 *
ytotal / I) ; %[Pa]
```

```
bendstresstrial4 = abs(Mtotaltrial4 * ytotal / I) ; %[Pa]

bendstresstrial5 = abs(Mtotaltrial5 * ytotal / I) ; %[Pa]


MPabendstresstrial1 = bend-stresstrial1 ./ 1000000 ; %[MPa]

MPabendstresstrial2 = bend-stresstrial2 ./ 1000000 ; %[MPa]

MPabendstresstrial3 = bend-stresstrial3 ./ 1000000 ; %[MPa]

MPabendstresstrial4 = bend-stresstrial4 ./ 1000000 ; %[MPa]

MPabendstresstrial5 = bend-stresstrial5 ./ 1000000 ; %[MPa]


figure

plot(strain, stress)

title("Stress vs Strain relationship of Carbon Fiber")

xlabel("Strain [%]")

ylabel("Stress [Pa]")


figure

plot(stpercenttrial1, MPabend-stresstrial1)

title("Bending Stress During Stance Phase")

xlabel("Percent of Stance Phase")

ylabel("Stress [MPa]")


hold on


plot(stpercenttrial2, MPabend-stresstrial2)

plot(stpercenttrial3, MPabend-stresstrial3)

plot(stpercenttrial4, MPabend-stresstrial4)

plot(stpercenttrial5, MPabend-stresstrial5)


strainenergytrial1 = (bendstresstrial1.^2 .* strutL) ./ (2*modulus*Irod) ;

strainenergytrial2 = (bendstresstrial2.^2 .* strutL) ./ (2*modulus*Irod) ;

strainenergytrial3 = (bendstresstrial3.^2 .* strutL) ./ (2*modulus*Irod) ;

strainenergytrial4 = (bendstresstrial4.^2 .* strutL) ./ (2*modulus*Irod) ;

strainenergytrial5 = (bendstresstrial5.^2 .* strutL) ./ (2*modulus*Irod) ;


figure

plot(stpercenttrial1, strainener-gytrial1)

title("Strain Energy in Struts During Stance Phase")

xlabel("Percent of Stance Phase")

ylabel("Strain Energy [J]")


hold on


plot(stpercenttrial2, strainener-gytrial2)

plot(stpercenttrial3, strainener-gytrial3)

plot(stpercenttrial4, strainener-gytrial4)

plot(stpercenttrial5, strainener-gytrial5)
```

Appendix B: Full MatLab Code - Healthy Leg Calculations

```
%% Import files for coincident GRF (txt) and motion (csv) of trial 2**

fnameforcetrial2 = "trial2txt.txt" ;
rawforcedatatrial2 = importdata (fnameforcetrial2) ;

fnamemotiontrial2 = "trial2csv.csv" ;
rawmotiondatatrial2 = csvread (fnamemotiontrial2) ;

% Stance times
% trial 1*: 0, 0
% trial 2: 0.258 , 1.258
% trial 3*: 0.300 , 1.375
% trial 4: 0.358 , 1.392
% trial 5: 0.842 , 1.925
% trial 6: 0.500 , 1.458

% INPUTS for this type of trial

HStimetrial2 = (0.258) ; % input [s]
TOtimetrial2 = (1.258) ; % input [s]

legwidth = (0.1333) ; % input [m] 5.250 in
footwidth = (0.0931) ; % input [m] 3.666 in

mbody = (91.6257) ; % input [kg] 202 pounds

% extracting column data for force *check y force

xforcetrial2 = rawforcedatatrial2(:,1) ;
yforcetrial2 = rawforcedatatrial2(:,2) ; % positive should be right side (medial for left foot)
zforcetrial2 = rawforcedatatrial2(:,3) ;

% set sampling rate and inital time domain for raw data
Fs = 120 ;
forcesamplestrial2 = length (rawforcedatatrial2) ;
forcetimetrial2 = (0:(1/Fs):((forcesamplestrial2/Fs)-(1/Fs))) ;

% plot raw force data over time in three dimensions

% extract sample numbers from motion data to create time domain
% delete the column with 0x0000's before analysis*****

motionsamplestrial2 = rawmotiondatatrial2(:,4);
motiontimetrial2 = (0:(1/Fs):(((length (motionsamplestrial2)/2)/Fs)-(1/Fs))) ;

% organize data by sensor/hub with if statements

hub1sensor1trial2 = zeros((floor(length (motionsamplestrial2)/2)), 3) ;

j = 1;
for i = 1 : (length (rawmotiondatatrial2))
    if (rawmotiondatatrial2(i,2) == 1) && (rawmotiondatatrial2(i,3) == 1)
        hub1sensor1trial2(j, :) = rawmotiondatatrial2(i, 5:7);
        j = j + 1;
    end
end
```

```
end

hub1sensor2trial2 = zeros((floor(length
(motionsamplestrial2)/2)), 3) ;


j = 1;
for i = 1 : (length
(rawmotiondatatrial2))
    if (rawmotiondatatrial2(i,2) == 1)
&& (rawmotiondatatrial2(i,3) == 2)
        hub1sensor2trial2(j, :) =
rawmotiondatatrial2(i, 5:7);
        j = j + 1;
    end
end


hub1sensor3trial2 = zeros((floor(length
(motionsamplestrial2)/2)), 3) ;


j = 1;
for i = 1 : (length
(rawmotiondatatrial2))
    if (rawmotiondatatrial2(i,2) == 1)
&& (rawmotiondatatrial2(i,3) == 3)
        hub1sensor3trial2(j, :) =
rawmotiondatatrial2(i, 5:7);
        j = j + 1;
    end
end


hub2sensor1trial2 = zeros((floor(length
(motionsamplestrial2)/2)), 3) ;


j = 1;
for i = 1 : (length
(rawmotiondatatrial2))
    if (rawmotiondatatrial2(i,2) == 2)
&& (rawmotiondatatrial2(i,3) == 1)
        hub2sensor1trial2(j, :) =
rawmotiondatatrial2(i, 5:7);
        j = j + 1;
    end
end


hub2sensor2trial2 = zeros((floor(length
(motionsamplestrial2)/2)), 3) ;


j = 1;
for i = 1 : (length
(rawmotiondatatrial2))
    if (rawmotiondatatrial2(i,2) == 2)
&& (rawmotiondatatrial2(i,3) == 2)
        hub2sensor2trial2(j, :) =
rawmotiondatatrial2(i, 5:7);
        j = j + 1;
    end
end


% Sensor locations (lateral)
% 1,1 = COM of foot
% 1,2 = COM of leg
% 1,3 = femoral condyles at knee
% 2,1 = heel
% 2,2 = toes - metatarsals


% here we can fix the sign of each set
of data if we want and correct the
% y value of the displacement data for
the leg and foot


fix = -1 ;
intom = 0.0254 ; %[m/in]


footdispXtrial2 =
intom*hub1sensor1trial2(:,1) ;
footdispYtrial2 =
intom*hub1sensor1trial2(:,2) +
(0.5*footwidth) ; %add for left leg
lateral sensors
footdispZtrial2 =
intom*hub1sensor1trial2(:,3) ;
```

```
legdispXtrial2 = intom*hub1sensor2trial2
(:,1) ;

legdispYtrial2 = intom*hub1sensor2trial2
(:,2) + (0.5*legwidth) ; %add for left
leg lateral sensors

legdispZtrial2 = intom*hub1sensor2trial2
(:,3) ;


% plot raw displacement data over time
in three dimensions


%Normalized Data inputs for full step
with force plate


HSsampletrial2 = floor(HStimetrial2*Fs)
- 2 ;

TOsampletrial2 = floor
(TOtimetrial2*Fs) ;

normalsampleswrongtrial2 = (0:
(TOsampletrial2 - HSsampletrial2)) ;

normalsamplestrial2 = transpose
(normalsampleswrongtrial2) ;


%make the displacement data normalized

normalfootdispXtrial2 = footdispXtrial2
(HSsampletrial2:TOsampletrial2) ;

normalfootdispYtrial2 = footdispYtrial2
(HSsampletrial2:TOsampletrial2) ;

normalfootdispZtrial2 = footdispZtrial2
(HSsampletrial2:TOsampletrial2) ;


normallegdispXtrial2 = legdispXtrial2
(HSsampletrial2:TOsampletrial2) ;

normallegdispYtrial2 = legdispYtrial2
(HSsampletrial2:TOsampletrial2) ;

normallegdispZtrial2 = legdispZtrial2
(HSsampletrial2:TOsampletrial2) ;


%FILTER NORMALIZED DISPLACEMENT DATA,
THEN DIFFERENTIATE IT


%design fourth order butterworth filter


Wp = 5/60;

Ws = 20/60;


[n,Wn] = buttord(Wp,Ws,3,60) ;


[z,p,k] = butter(n,Wn);

filter1 = zp2sos(z,p,k);

freqz(filter1,66,120)

title(sprintf('n = %d Butterworth Low-
pass Filter',n))


%filter foot and leg displacement data &
plot


filteredfootdispXtrial2 = filtfilt
(filter1, n, normalfootdispXtrial2) ;

filteredfootdispYtrial2 = filtfilt
(filter1, n, normalfootdispYtrial2) ;

filteredfootdispZtrial2 = filtfilt
(filter1, n, normalfootdispZtrial2) ;


filteredlegdispXtrial2 = filtfilt
(filter1, n, normallegdispXtrial2) ;

filteredlegdispYtrial2 = filtfilt
(filter1, n, normallegdispYtrial2) ;

filteredlegdispZtrial2 = filtfilt
(filter1, n, normallegdispZtrial2) ;


%use normalsamples time domain and plot
filtered displacement data


%differentiate these displacement values
```

```
to get velocity

filteredfootvelXtrial2 = diff
(filteredfootdispXtrial2) ;

filteredfootvelYtrial2 = diff
(filteredfootdispYtrial2) ;

filteredfootvelZtrial2 = diff
(filteredfootdispZtrial2) ;

filteredlegvelXtrial2 = diff
(filteredlegdispXtrial2) ;

filteredlegvelYtrial2 = diff
(filteredlegdispYtrial2) ;

filteredlegvelZtrial2 = diff
(filteredlegdispZtrial2) ;

% make time domain and plot

veltimetrial2 = 0:(1/Fs):((1/Fs)*length
(filteredfootvelXtrial2) - (1/Fs)) ;

%differentiate these velocity values to
get accleration

finalfootaccXtrial2 = diff
(filteredfootvelXtrial2) ;

finalfootaccYtrial2 = diff
(filteredfootvelYtrial2) ;

finalfootaccZtrial2 = diff
(filteredfootvelZtrial2) ;

finallegaccXtrial2 = diff
(filteredlegvelXtrial2) ;

finallegaccYtrial2 = diff
(filteredlegvelYtrial2) ;

finallegaccZtrial2 = diff
(filteredlegvelZtrial2) ;

%create time domain & plot

acceltimetrial2 = (1/Fs) * (0:(length
(finalfootaccXtrial2)-1)) ;

% combine variables into 3D matrices

normalxforcetrial2 = xforcetrial2
((HSsampletrial2+2):TOsampletrial2) ;

normalyforcetrial2 = yforcetrial2
((HSsampletrial2+2):TOsampletrial2) ;

normalzforcetrial2 = zforcetrial2
((HSsampletrial2+2):TOsampletrial2) ;

footacctrial2 = [finalfootaccXtrial2 fi-
nalfootaccYtrial2 finalfootaccZtrial2] ;
%[m/s^2]

legacctrial2 = [finallegaccXtrial2 fi-
nallegaccYtrial2 finallegaccZtrial2] ; %
[m/s^2]

GRFtrial2 = [normalxforcetrial2 nor-
malyforcetrial2 normalzforcetrial2] ; %
[N]

%use footacc with GRF to calculate ankle
reaction force

gravity = [0 0 9.8] ; %[m/s^2] Z direc-
tion!

mfoot = 0.0145*mbody ;

ARF3Dtrial2 = (mfoot.*footacctrial2) -
GRFtrial2 - (mfoot.*gravity) ;

%use legacc with ARF3D to calculate knee
reaction force

mleg = 0.0465*mbody ;
```

```
KRF3Dtrial2 = (mleg.*legacctrial2) +
ARF3Dtrial2 - (mleg.*gravity) ;

%take out components of KRF3D

krXtrial2 = KRF3Dtrial2(:,1) ;
krYtrial2 = KRF3Dtrial2(:,2) ;
krZtrial2 = KRF3Dtrial2(:,3) ;

kneetimetrial2 = acceltimetrial2 ;

%% Import files for coincident GRF (txt)
and motion (csv) of trial 4**

fnameforcetrial4 = "trial4txt.txt" ;
rawforcedatatrial4 = importdata
(fnameforcetrial4) ;

fnamemotiontrial4 = "trial4csv.csv" ;
rawmotiondatatrial4 = csvread
(fnamemotiontrial4) ;

% Stance times
% trial 1*: 0, 0
% trial 2: 0.258 , 1.258
% trial 3*: 0.300 , 1.375
% trial 4: 0.358 , 1.392
% trial 5: 0.842 , 1.925
% trial 6: 0.500 , 1.458

% INPUTS for this type of trial

HStimetrial4 = (0.358) ; % input [s]
TOtimetrial4 = (1.392) ; % input [s]

% extracting column data for force
*check y force

xforcetrial4 = rawforcedatatrial4(:,1) ;
yforcetrial4 = rawforcedatatrial4(:,2) ;
% positive should be right side (medial
for left foot)
zforcetrial4 = rawforcedatatrial4(:,3) ;

% set sampling rate and inital time do-
main for raw data

forcesamplestrial4 = length
(rawforcedatatrial4) ;
forcetimetrial4 = (0:(1/Fs):
((forcesamplestrial4/Fs)-(1/Fs))) ;

% plot raw force data over time in three
dimensions

% extract sample numbers from motion da-
ta to create time domain
% delete the column with 0x0000's before
analysis*****

motionsamplestrial4 = rawmotiondata-
trial4(:,4);
motiontimetrial4 = (0:(1/Fs):(((length
(motionsamplestrial4)/2)/Fs)-(1/Fs))) ;

% organize data by sensor/hub with if
statements

hub1sensor1trial4 = zeros((floor(length
(motionsamplestrial4)/2)), 3) ;

j = 1;
for i = 1 : (length
(rawmotiondatatrial4))
    if (rawmotiondatatrial4(i,2) == 1)
&& (rawmotiondatatrial4(i,3) == 1)
```

```
        hub1sensor1trial4(j, :) = rawmo-
tiondatatrial4(i, 5:7);
        j = j + 1;
    end
end

hub1sensor2trial4 = zeros((floor(length
(motionsamplestrial4)/2)), 3) ;

j = 1;
for i = 1 : (length
(rawmotiondatatrial4))
    if (rawmotiondatatrial4(i,2) == 1)
&& (rawmotiondatatrial4(i,3) == 2)
        hub1sensor2trial4(j, :) = rawmo-
tiondatatrial4(i, 5:7);
        j = j + 1;
    end
end

hub1sensor3trial4 = zeros((floor(length
(motionsamplestrial4)/2)), 3) ;

j = 1;
for i = 1 : (length
(rawmotiondatatrial4))
    if (rawmotiondatatrial4(i,2) == 1)
&& (rawmotiondatatrial4(i,3) == 3)
        hub1sensor3trial4(j, :) = rawmo-
tiondatatrial4(i, 5:7);
        j = j + 1;
    end
end

hub2sensor1trial4 = zeros((floor(length
(motionsamplestrial4)/2)), 3) ;

j = 1;
for i = 1 : (length
(rawmotiondatatrial4))
    if (rawmotiondatatrial4(i,2) == 2)
&& (rawmotiondatatrial4(i,3) == 1)
        hub2sensor1trial4(j, :) = rawmo-
tiondatatrial4(i, 5:7);
        j = j + 1;
    end
end

hub2sensor2trial4 = zeros((floor(length
(motionsamplestrial4)/2)), 3) ;

j = 1;
for i = 1 : (length
(rawmotiondatatrial4))
    if (rawmotiondatatrial4(i,2) == 2)
&& (rawmotiondatatrial4(i,3) == 2)
        hub2sensor2trial4(j, :) = rawmo-
tiondatatrial4(i, 5:7);
        j = j + 1;
    end
end

% Sensor locations (lateral)
% 1,1 = COM of foot
% 1,2 = COM of leg
% 1,3 = femoral condyles at knee
% 2,1 = heel
% 2,2 = toes - metatarsals

% here we can fix the sign of each set
of data if we want and correct the
% y value of the displacement data for
the leg and foot

footdispXtrial4 =
intom*hub1sensor1trial4(:,1) ;
footdispYtrial4 =
intom*hub1sensor1trial4(:,2) +
(0.5*footwidth) ; %add for left leg lat-
eral sensors
footdispZtrial4 =
intom*hub1sensor1trial4(:,3) ;

legdispXtrial4 = intom*hub1sensor2trial4
```

```
(:,1) ;

legdispYtrial4 = intom*hub1sensor2trial4
(:,2) + (0.5*legwidth) ; %add for left
leg lateral sensors

legdispZtrial4 = intom*hub1sensor2trial4
(:,3) ;


% plot raw displacement data over time
in three dimensions




%Normalized Data inputs for full step
with force plate


HSsampletrial4 = floor(HStimetrial4*Fs)
- 2 ;

TOsampletrial4 = floor
(TOtimetrial4*Fs) ;

normalsampleswrongtrial4 = (0:
(TOsampletrial4 - HSsampletrial4)) ;

normalsamplestrial4 = transpose
(normalsampleswrongtrial4) ;


%make the displacement data normalized


normalfootdispXtrial4 = footdispXtrial4
(HSsampletrial4:TOsampletrial4) ;

normalfootdispYtrial4 = footdispYtrial4
(HSsampletrial4:TOsampletrial4) ;

normalfootdispZtrial4 = footdispZtrial4
(HSsampletrial4:TOsampletrial4) ;


normallegdispXtrial4 = legdispXtrial4
(HSsampletrial4:TOsampletrial4) ;

normallegdispYtrial4 = legdispYtrial4
(HSsampletrial4:TOsampletrial4) ;

normallegdispZtrial4 = legdispZtrial4
(HSsampletrial4:TOsampletrial4) ;



%FILTER NORMALIZED DISPLACEMENT DATA,
THEN DIFFERENTIATE IT


%design fourth order butterworth filter




%filter foot and leg displacement data &
plot


filteredfootdispXtrial4 = filtfilt
(filter1, n, normalfootdispXtrial4) ;

filteredfootdispYtrial4 = filtfilt
(filter1, n, normalfootdispYtrial4) ;

filteredfootdispZtrial4 = filtfilt
(filter1, n, normalfootdispZtrial4) ;


filteredlegdispXtrial4 = filtfilt
(filter1, n, normallegdispXtrial4) ;

filteredlegdispYtrial4 = filtfilt
(filter1, n, normallegdispYtrial4) ;

filteredlegdispZtrial4 = filtfilt
(filter1, n, normallegdispZtrial4) ;


%use normalsamples time domain and plot
filtered displacement data


%differentiate these displacement values
to get velocity


filteredfootvelXtrial4 = diff
(filteredfootdispXtrial4) ;

filteredfootvelYtrial4 = diff
(filteredfootdispYtrial4) ;

filteredfootvelZtrial4 = diff
(filteredfootdispZtrial4) ;


filteredlegvelXtrial4 = diff
(filteredlegdispXtrial4) ;

filteredlegvelYtrial4 = diff
(filteredlegdispYtrial4) ;

filteredlegvelZtrial4 = diff
```

```
(filteredlegdispZtrial4) ;

% make time domain and plot

veltimetrial4 = 0:(1/Fs):((1/Fs)*length
(filteredfootvelXtrial4) - (1/Fs)) ;

%differentiate these velocity values to
get accleration

finalfootaccXtrial4 = diff
(filteredfootvelXtrial4) ;

finalfootaccYtrial4 = diff
(filteredfootvelYtrial4) ;

finalfootaccZtrial4 = diff
(filteredfootvelZtrial4) ;

finallegaccXtrial4 = diff
(filteredlegvelXtrial4) ;

finallegaccYtrial4 = diff
(filteredlegvelYtrial4) ;

finallegaccZtrial4 = diff
(filteredlegvelZtrial4) ;

%create time domain & plot

acceltimetrial4 = (1/Fs) * (0:(length
(finalfootaccXtrial4)-1)) ;

% combine variables into 3D matrices

normalxforcetrial4 = xforcetrial4
((HSsampletrial4+2):TOsampletrial4) ;

normalyforcetrial4 = yforcetrial4
((HSsampletrial4+2):TOsampletrial4) ;

normalzforcetrial4 = zforcetrial4
((HSsampletrial4+2):TOsampletrial4) ;

footacctrial4 = [finalfootaccXtrial4 fi-
nalfootaccYtrial4 finalfootaccZtrial4] ;
%[m/s^2]

legacctrial4 = [finallegaccXtrial4 fi-
nallegaccYtrial4 finallegaccZtrial4] ; %
[m/s^2]

GRFtrial4 = [normalxforcetrial4 nor-
malyforcetrial4 normalzforcetrial4] ; %
[N]

%use footacc with GRF to calculate ankle
reaction force

ARF3Dtrial4 = (mfoot.*footacctrial4) -
GRFtrial4 - (mfoot.*gravity) ;

%use legacc with ARF3D to calculate knee
reaction force

KRF3Dtrial4 = (mleg.*legacctrial4) +
ARF3Dtrial4 - (mleg.*gravity) ;

%take out components of KRF3D

krXtrial4 = KRF3Dtrial4(:,1) ;

krYtrial4 = KRF3Dtrial4(:,2) ;

krZtrial4 = KRF3Dtrial4(:,3) ;

kneetimetrial4 = acceltimetrial4 ;

%% Import files for coincident GRF (txt)
and motion (csv) of trial 5**

fnameforcetrial5 = "trial5txt.txt" ;

rawforcedatatrial5 = importdata
(fnameforcetrial5) ;

fnamemotiontrial5 = "trial5csv.csv" ;
```

```
rawmotiondatatrial5 = csvread
(fnamemotiontrial5) ;


% Stance times

% trial 1*: 0, 0

% trial 2: 0.258 , 1.258

% trial 3*: 0.300 , 1.375

% trial 4: 0.358 , 1.392

% trial 5: 0.842 , 1.925

% trial 6: 0.500 , 1.458


% INPUTS for this type of trial


HStimetrial5 = (0.842) ; % input [s]

TOtimetrial5 = (1.925) ; % input [s]


% extracting column data for force
*check y force


xforcetrial5 = rawforcedatatrial5(:,1) ;

yforcetrial5 = rawforcedatatrial5(:,2) ;
% positive should be right side (medial
for left foot)

zforcetrial5 = rawforcedatatrial5(:,3) ;


% set sampling rate and inital time do-
main for raw data


forcesamplestrial5 = length
(rawforcedatatrial5) ;

forcetimetrial5 = (0:(1/Fs):
((forcesamplestrial5/Fs)-(1/Fs))) ;


% plot raw force data over time in three
dimensions


% extract sample numbers from motion da-
ta to create time domain

% delete the column with 0x0000's before
analysis*****


motionsamplestrial5 = rawmotiondata-
trial5(:,4);

motiontimetrial5 = (0:(1/Fs):(((length
(motionsamplestrial5)/2)/Fs)-(1/Fs))) ;


% organize data by sensor/hub with if
statements


hub1sensor1trial5 = zeros((floor(length
(motionsamplestrial5)/2)), 3) ;

j = 1;

for i = 1 : (length
(rawmotiondatatrial5))

    if (rawmotiondatatrial5(i,2) == 1)
&& (rawmotiondatatrial5(i,3) == 1)

        hub1sensor1trial5(j, :) = rawmo-
tiondatatrial5(i, 5:7);

        j = j + 1;

    end

end


hub1sensor2trial5 = zeros((floor(length
(motionsamplestrial5)/2)), 3) ;


j = 1;

for i = 1 : (length
(rawmotiondatatrial5))

    if (rawmotiondatatrial5(i,2) == 1)
&& (rawmotiondatatrial5(i,3) == 2)

        hub1sensor2trial5(j, :) = rawmo-
tiondatatrial5(i, 5:7);

        j = j + 1;

    end

end


hub1sensor3trial5 = zeros((floor(length
(motionsamplestrial5)/2)), 3) ;
```

```
j = 1;
for i = 1 : (length
(rawmotiondatatrial5))
    if (rawmotiondatatrial5(i,2) == 1)
&& (rawmotiondatatrial5(i,3) == 3)
        hub1sensor3trial5(j, :) = rawmo-
tiondatatrial5(i, 5:7);
        j = j + 1;
    end
end

hub2sensor1trial5 = zeros((floor(length
(motionsamplestrial5)/2)), 3) ;

j = 1;
for i = 1 : (length
(rawmotiondatatrial5))
    if (rawmotiondatatrial5(i,2) == 2)
&& (rawmotiondatatrial5(i,3) == 1)
        hub2sensor1trial5(j, :) = rawmo-
tiondatatrial5(i, 5:7);
        j = j + 1;
    end
end

hub2sensor2trial5 = zeros((floor(length
(motionsamplestrial5)/2)), 3) ;

j = 1;
for i = 1 : (length
(rawmotiondatatrial5))
    if (rawmotiondatatrial5(i,2) == 2)
&& (rawmotiondatatrial5(i,3) == 2)
        hub2sensor2trial5(j, :) = rawmo-
tiondatatrial5(i, 5:7);
        j = j + 1;
    end
end

% Sensor locations (lateral)
% 1,1 = COM of foot
% 1,2 = COM of leg
% 1,3 = femoral condyles at knee
% 2,1 = heel
% 2,2 = toes - metatarsals

% here we can fix the sign of each set
of data if we want and correct the
% y value of the displacement data for
the leg and foot

footdispXtrial5 =
intom*hub1sensor1trial5(:,1) ;
footdispYtrial5 =
intom*hub1sensor1trial5(:,2) +
(0.5*footwidth) ; %add for left leg lat-
eral sensors
footdispZtrial5 =
intom*hub1sensor1trial5(:,3) ;

legdispXtrial5 = intom*hub1sensor2trial5
(:,1) ;
legdispYtrial5 = intom*hub1sensor2trial5
(:,2) + (0.5*legwidth) ; %add for left
leg lateral sensors
legdispZtrial5 = intom*hub1sensor2trial5
(:,3) ;

% plot raw displacement data over time
in three dimensions

%Normalized Data inputs for full step
with force plate

HSsampletrial5 = floor(HStimetrial5*Fs)
- 2 ;
TOsampletrial5 = floor
(TOtimetrial5*Fs) ;
normalsampleswrongtrial5 = (0:
(TOsampletrial5 - HSsampletrial5)) ;
normalsamplestrial5 = transpose
(normalsampleswrongtrial5) ;
```

```
%make the displacement data normalized

normalfootdispXtrial5 = footdispXtrial5
(HSsampletrial5:TOsampletrial5) ;
normalfootdispYtrial5 = footdispYtrial5
(HSsampletrial5:TOsampletrial5) ;
normalfootdispZtrial5 = footdispZtrial5
(HSsampletrial5:TOsampletrial5) ;

normallegdispXtrial5 = legdispXtrial5
(HSsampletrial5:TOsampletrial5) ;
normallegdispYtrial5 = legdispYtrial5
(HSsampletrial5:TOsampletrial5) ;
normallegdispZtrial5 = legdispZtrial5
(HSsampletrial5:TOsampletrial5) ;

%FILTER NORMALIZED DISPLACEMENT DATA,
THEN DIFFERENTIATE IT

%design fourth order butterworth filter

%filter foot and leg displacement data
& plot

filteredfootdispXtrial5 = filtfilt
(filter1, n, normalfootdispXtrial5) ;
filteredfootdispYtrial5 = filtfilt
(filter1, n, normalfootdispYtrial5) ;
filteredfootdispZtrial5 = filtfilt
(filter1, n, normalfootdispZtrial5) ;

filteredlegdispXtrial5 = filtfilt
(filter1, n, normallegdispXtrial5) ;
filteredlegdispYtrial5 = filtfilt
(filter1, n, normallegdispYtrial5) ;
filteredlegdispZtrial5 = filtfilt
(filter1, n, normallegdispZtrial5) ;

%use normalsamples time domain and plot
filtered displacement data

%differentiate these displacement val-
ues to get velocity

filteredfootvelXtrial5 = diff
(filteredfootdispXtrial5) ;
filteredfootvelYtrial5 = diff
(filteredfootdispYtrial5) ;
filteredfootvelZtrial5 = diff
(filteredfootdispZtrial5) ;

filteredlegvelXtrial5 = diff
(filteredlegdispXtrial5) ;
filteredlegvelYtrial5 = diff
(filteredlegdispYtrial5) ;
filteredlegvelZtrial5 = diff
(filteredlegdispZtrial5) ;

% make time domain and plot

veltimetrial5 = 0:(1/Fs):((1/Fs)*length
(filteredfootvelXtrial5) - (1/Fs)) ;

%differentiate these velocity values to
get accleration

finalfootaccXtrial5 = diff
(filteredfootvelXtrial5) ;
finalfootaccYtrial5 = diff
(filteredfootvelYtrial5) ;
finalfootaccZtrial5 = diff
(filteredfootvelZtrial5) ;

finallegaccXtrial5 = diff
(filteredlegvelXtrial5) ;
```

```
finallegaccYtrial5 = diff
(filteredlegvelYtrial5) ;

finallegaccZtrial5 = diff
(filteredlegvelZtrial5) ;


%create time domain & plot


acceltimetrial5 = (1/Fs) * (0:(length
(finalfootaccXtrial5)-1)) ;


% combine variables into 3D matrices

normalxforcetrial5 = xforcetrial5
((HSsampletrial5+2):TOsampletrial5) ;

normalyforcetrial5 = yforcetrial5
((HSsampletrial5+2):TOsampletrial5) ;

normalzforcetrial5 = zforcetrial5
((HSsampletrial5+2):TOsampletrial5) ;


footacctrial5 = [finalfootaccXtrial5 fi-
nalfootaccYtrial5 finalfootaccZtrial5] ;
%[m/s^2]

legacctrial5 = [finallegaccXtrial5 fi-
nallegaccYtrial5 finallegaccZtrial5] ; %
[m/s^2]

GRFtrial5 = [normalxforcetrial5 nor-
malyforcetrial5 normalzforcetrial5] ; %
[N]

%use footacc with GRF to calculate ankle
reaction force


ARF3Dtrial5 = (mfoot.*footacctrial5) -
GRFtrial5 - (mfoot.*gravity) ;


%use legacc with ARF3D to calculate knee
reaction force


KRF3Dtrial5 = (mleg.*legacctrial5) +
ARF3Dtrial5 - (mleg.*gravity) ;


%take out components of KRF3D


krXtrial5 = KRF3Dtrial5(:,1) ;

krYtrial5 = KRF3Dtrial5(:,2) ;

krZtrial5 = KRF3Dtrial5(:,3) ;


kneetimetrial5 = acceltimetrial5 ;



%% Import files for coincident GRF (txt)
and motion (csv) of trial 6**


fnameforcetrial6 = "trial6txt.txt" ;

rawforcedatatrial6 = importdata
(fnameforcetrial6) ;


fnamemotiontrial6 = "trial6csv.csv" ;

rawmotiondatatrial6 = csvread
(fnamemotiontrial6) ;


% Stance times

% trial 1*: 0, 0

% trial 2: 0.258 , 1.258

% trial 3*: 0.300 , 1.375

% trial 4: 0.358 , 1.392

% trial 5: 0.842 , 1.925

% trial 6: 0.500 , 1.458


% INPUTS for this type of trial


HStimetrial6 = (0.500) ; % input [s]

TOtimetrial6 = (1.458) ; % input [s]


% extracting column data for force
*check y force


xforcetrial6 = rawforcedatatrial6(:,1) ;

yforcetrial6 = rawforcedatatrial6(:,2) ;
% positive should be right side (medial
```

```
for left foot)

zforcetrial6 = rawforcedatatrial6(:,3) ;


% set sampling rate and inital time do-
main for raw data


forcesamplestrial6 = length
(rawforcedatatrial6) ;

forcetimetrial6 = (0:(1/Fs):
((forcesamplestrial6/Fs)-(1/Fs))) ;


% plot raw force data over time in three
dimensions




% extract sample numbers from motion da-
ta to create time domain

% delete the column with 0x0000's before
analysis*****


motionsamplestrial6 = rawmotiondata-
trial6(:,4);

motiontimetrial6 = (0:(1/Fs):(((length
(motionsamplestrial6)/2)/Fs)-(1/Fs))) ;


% organize data by sensor/hub with if
statements


hub1sensor1trial6 = zeros((floor(length
(motionsamplestrial6)/2)), 3) ;


j = 1;

for i = 1 : (length
(rawmotiondatatrial6))

    if (rawmotiondatatrial6(i,2) == 1)
&& (rawmotiondatatrial6(i,3) == 1)

        hub1sensor1trial6(j, :) = rawmo-
tiondatatrial6(i, 5:7);

        j = j + 1;

    end

end


hub1sensor2trial6 = zeros((floor(length
(motionsamplestrial6)/2)), 3) ;


j = 1;

for i = 1 : (length
(rawmotiondatatrial6))

    if (rawmotiondatatrial6(i,2) == 1)
&& (rawmotiondatatrial6(i,3) == 2)

        hub1sensor2trial6(j, :) = rawmo-
tiondatatrial6(i, 5:7);

        j = j + 1;

    end

end


hub1sensor3trial6 = zeros((floor(length
(motionsamplestrial6)/2)), 3) ;


j = 1;

for i = 1 : (length
(rawmotiondatatrial6))

    if (rawmotiondatatrial6(i,2) == 1)
&& (rawmotiondatatrial6(i,3) == 3)

        hub1sensor3trial6(j, :) = rawmo-
tiondatatrial6(i, 5:7);

        j = j + 1;

    end

end


hub2sensor1trial6 = zeros((floor(length
(motionsamplestrial6)/2)), 3) ;


j = 1;

for i = 1 : (length
(rawmotiondatatrial6))

    if (rawmotiondatatrial6(i,2) == 2)
&& (rawmotiondatatrial6(i,3) == 1)

        hub2sensor1trial6(j, :) = rawmo-
tiondatatrial6(i, 5:7);
```

```
        j = j + 1;
    end
end

hub2sensor2trial6 = zeros((floor(length(motionsamplestrial6)/2)), 3) ;

j = 1;
for i = 1 : (length(rawmotiondatatrial6))
    if (rawmotiondatatrial6(i,2) == 2) && (rawmotiondatatrial6(i,3) == 2)
        hub2sensor2trial6(j, :) = rawmotiondatatrial6(i, 5:7);
        j = j + 1;
    end
end

% Sensor locations (lateral)
% 1,1 = COM of foot
% 1,2 = COM of leg
% 1,3 = femoral condyles at knee
% 2,1 = heel
% 2,2 = toes - metatarsals

% here we can fix the sign of each set of data if we want and correct the
% y value of the displacement data for the leg and foot

footdispXtrial6 = intom*hub1sensor1trial6(:,1) ;
footdispYtrial6 = intom*hub1sensor1trial6(:,2) + (0.5*footwidth) ; %add for left leg lateral sensors
footdispZtrial6 = intom*hub1sensor1trial6(:,3) ;

legdispXtrial6 = intom*hub1sensor2trial6(:,1) ;
legdispYtrial6 = intom*hub1sensor2trial6(:,2) + (0.5*legwidth) ; %add for left leg lateral sensors
legdispZtrial6 = intom*hub1sensor2trial6(:,3) ;

% plot raw displacement data over time in three dimensions

%Normalized Data inputs for full step with force plate

HSsampletrial6 = floor(HStimetrial6*Fs) - 2 ;
TOsampletrial6 = floor(TOtimetrial6*Fs) ;
normalsampleswrongtrial6 = (0:(TOsampletrial6 - HSsampletrial6)) ;
normalsamplestrial6 = transpose(normalsampleswrongtrial6) ;

%make the displacement data normalized

normalfootdispXtrial6 = footdispXtrial6(HSsampletrial6:TOsampletrial6) ;
normalfootdispYtrial6 = footdispYtrial6(HSsampletrial6:TOsampletrial6) ;
normalfootdispZtrial6 = footdispZtrial6(HSsampletrial6:TOsampletrial6) ;

normallegdispXtrial6 = legdispXtrial6(HSsampletrial6:TOsampletrial6) ;
normallegdispYtrial6 = legdispYtrial6(HSsampletrial6:TOsampletrial6) ;
normallegdispZtrial6 = legdispZtrial6(HSsampletrial6:TOsampletrial6) ;

%FILTER NORMALIZED DISPLACEMENT DATA, THEN DIFFERENTIATE IT
```

```
%design fourth order butterworth filter

%filter foot and leg displacement data &
plot

filteredfootdispXtrial6 = filtfilt
(filter1, n, normalfootdispXtrial6) ;
filteredfootdispYtrial6 = filtfilt
(filter1, n, normalfootdispYtrial6) ;
filteredfootdispZtrial6 = filtfilt
(filter1, n, normalfootdispZtrial6) ;

filteredlegdispXtrial6 = filtfilt
(filter1, n, normallegdispXtrial6) ;
filteredlegdispYtrial6 = filtfilt
(filter1, n, normallegdispYtrial6) ;
filteredlegdispZtrial6 = filtfilt
(filter1, n, normallegdispZtrial6) ;

%use normalsamples time domain and plot
filtered displacement data

%differentiate these displacement values
to get velocity

filteredfootvelXtrial6 = diff
(filteredfootdispXtrial6) ;
filteredfootvelYtrial6 = diff
(filteredfootdispYtrial6) ;
filteredfootvelZtrial6 = diff
(filteredfootdispZtrial6) ;

filteredlegvelXtrial6 = diff
(filteredlegdispXtrial6) ;
filteredlegvelYtrial6 = diff
(filteredlegdispYtrial6) ;
filteredlegvelZtrial6 = diff
(filteredlegdispZtrial6) ;

% make time domain and plot

veltimetrial6 = 0:(1/Fs):((1/Fs)*length
(filteredfootvelXtrial6) - (1/Fs)) ;

%differentiate these velocity values to
get accleration

finalfootaccXtrial6 = diff
(filteredfootvelXtrial6) ;
finalfootaccYtrial6 = diff
(filteredfootvelYtrial6) ;
finalfootaccZtrial6 = diff
(filteredfootvelZtrial6) ;

finallegaccXtrial6 = diff
(filteredlegvelXtrial6) ;
finallegaccYtrial6 = diff
(filteredlegvelYtrial6) ;
finallegaccZtrial6 = diff
(filteredlegvelZtrial6) ;

%create time domain & plot

acceltimetrial6 = (1/Fs) * (0:(length
(finalfootaccXtrial6)-1)) ;

% combine variables into 3D matrices
normalxforcetrial6 = xforcetrial6
((HSsampletrial6+2):TOsampletrial6) ;
normalyforcetrial6 = yforcetrial6
((HSsampletrial6+2):TOsampletrial6) ;
normalzforcetrial6 = zforcetrial6
((HSsampletrial6+2):TOsampletrial6) ;
```

```
footacctrial6 = [finalfootaccXtrial6 finalfootaccYtrial6 finalfootaccZtrial6] ; %[m/s^2]

legacctrial6 = [finallegaccXtrial6 finallegaccYtrial6 finallegaccZtrial6] ; %[m/s^2]

GRFtrial6 = [normalxforcetrial6 normalyforcetrial6 normalzforcetrial6] ; %[N]

%use footacc with GRF to calculate ankle reaction force

ARF3Dtrial6 = (mfoot.*footacctrial6) - GRFtrial6 - (mfoot.*gravity) ;

%use legacc with ARF3D to calculate knee reaction force

KRF3Dtrial6 = (mleg.*legacctrial6) + ARF3Dtrial6 - (mleg.*gravity) ;

%take out components of KRF3D

krXtrial6 = KRF3Dtrial6(:,1) ;
krYtrial6 = KRF3Dtrial6(:,2) ;
krZtrial6 = KRF3Dtrial6(:,3) ;

kneetimetrial6 = acceltimetrial6 ;

%% PLOTS FOR ALL TRIALS COMBINED

%Reaction Force over percent of Stance Phase

%create percent time domain for every trial

percentztrial2 = kneetimetrial2(length(kneetimetrial2)) ;

for i = 1 : (length(kneetimetrial2))
    stpercenttrial2(i) = kneetimetrial2(i) * (100 / percentztrial2)  ;
end

percentztrial4 = kneetimetrial4(length(kneetimetrial4)) ;

for i = 1 : (length(kneetimetrial4))
    stpercenttrial4(i) = kneetimetrial4(i) * (100 / percentztrial4)  ;
end

percentztrial5 = kneetimetrial5(length(kneetimetrial5)) ;

for i = 1 : (length(kneetimetrial5))
    stpercenttrial5(i) = kneetimetrial5(i) * (100 / percentztrial5)  ;
end

percentztrial6 = kneetimetrial6(length(kneetimetrial6)) ;

for i = 1 : (length(kneetimetrial6))
    stpercenttrial6(i) = kneetimetrial6(i) * (100 / percentztrial6)  ;
end

%Knee Reaction Force Z

figure
plot(stpercenttrial2, krZtrial2)
xlim([0 100])
ylim([-1200 0])
title('Healthy Knee Reaction Force - Z direction')
xlabel('% of Stance Phase')
```

```
ylabel('Force [N]')

hold on

plot(stpercenttrial4, krZtrial4)
plot(stpercenttrial5, krZtrial5)
plot(stpercenttrial6, krZtrial6)

%Knee Reaction Force X

figure
plot(stpercenttrial2, krXtrial2)
xlim([0 100])
ylim([-200 150])
title('Healthy Knee Reaction Force - X direction')
xlabel('% of Stance Phase')
ylabel('Force [N]')

hold on

plot(stpercenttrial4, krXtrial4)
plot(stpercenttrial5, krXtrial5)
plot(stpercenttrial6, krXtrial6)

%Knee Reaction Force Y

figure
plot(stpercenttrial2, krYtrial2)
xlim([0 100])
ylim([-100 50])
title('Healthy Knee Reaction Force - Y direction')
xlabel('% of Stance Phase')
ylabel('Force [N]')

hold on

plot(stpercenttrial4, krYtrial4)
plot(stpercenttrial5, krYtrial5)
plot(stpercenttrial6, krYtrial6)

%Ground Reaction Force

% GRF x

figure
plot(stpercenttrial2, normalxforce-trial2)
xlim([0 100])
ylim([-150 200])
title("Ground Reaction Force - Healthy - X Direction")
xlabel("% of Stance Phase")
ylabel("Force [N]")

hold on

plot(stpercenttrial4, normalxforce-trial4)
plot(stpercenttrial5, normalxforce-trial5)
plot(stpercenttrial6, normalxforce-trial6)

% GRF y

figure
plot(stpercenttrial2, normalyforce-trial2)
xlim([0 100])
ylim([-50 100])
title("Ground Reaction Force - Healthy - Y Direction")
xlabel("% of Stance Phase")
ylabel("Force [N]")
```

```
hold on

plot(stpercenttrial4, normalyforce-
trial4)
plot(stpercenttrial5, normalyforce-
trial5)
plot(stpercenttrial6, normalyforce-
trial6)

% GRF z

figure
plot(stpercenttrial2, normalzforce-
trial2)
xlim([0 100])
ylim([0 1000])
title("Ground Reaction Force - Healthy -
Z Direction")
xlabel("% of Stance Phase")
ylabel("Force [N]")

hold on

plot(stpercenttrial4, normalzforce-
trial4)
plot(stpercenttrial5, normalzforce-
trial5)
plot(stpercenttrial6, normalzforce-
trial6)
```