Motion Generation of a Wearable Hip Exoskeleton Robot Using Machine Learning-Based Estimation of Ground Reaction Forces and Moments

by

Mohammad Mahdavian

B.Sc., University of Tehran, 2015

Thesis Submitted in Partial Fulfillment of the
Requirements for the Degree of
Master of Applied Science

in the
School of Mechatronic Systems Engineering
Faculty of Applied Science

Approval

Name:	**Mohammad Mahdavian**
Degree:	**Master of Applied Science**
Title:	**Motion Generation of a Wearable Hip Exoskeleton Robot Using Machine Learning-Based Estimation of Ground Reaction Forces and Moments**
Examining Committee:	**Chair:** Mohammad Narimani Lecturer
	Siamak Arzanpour Senior Supervisor Associate Professor
	Edward Jung Wook Park Co-Supervisor Professor
	Faranak Farzan Internal Examiner Assistant Professor
Date Defended/Approved:	September 11th, 2019

Ethics Statement

Abstract

Statistical data acquired from US citizens in 2013 showed that the overall percentage of all disabilities for all ages in this country was around 12.6%, in which the “ambulatory disabilities” had the highest prevalence rate (7.1 %) [1]. This amount is estimated around 7.2% for all Canadian adults, which corresponds to more than 2.5 million people [2]. In order to improve the quality of life of those with ambulatory disabilities (e.g., paraplegic people), wearable robotic exoskeleton is being developed in our lab.

In this project, Ground Reaction Forces and Moments (GRF/M), which are important data for closed-loop control of an exoskeleton, is estimated based on lower limb motion of a wearable hip exoskeleton user. This method can reduce manufacturing cost and design complications of these types of robots. In order to model GRF/M, Neural Network, Random Forest and Support Vector Machine algorithms are utilized. Afterward, the achieved results from the three algorithms are compared with each other and some of the most recent similar studies. In the next step, the trained models are employed in an online control loop for assisting a healthy exoskeleton user to walk easier. The device applies forces on the user’s upper thigh, which reduces the required torque of the hip flexion-extension joint for the user. Finally, the exoskeleton’s performance is compared experimentally with the cases when the device is not powered or it is simply following the user’s motion based on the inverse kinematics. The results showed the presented algorithm can help the exoskeleton user to walk easier.

Keywords: Ground Reaction Forces and Moments Estimation, Machine Learning, Neural Network, Random Forest, Support Vector Machine, Assistive Hip Exoskeleton

Dedication

I dedicate this thesis to all those who supported me in continuing my education and helped me to become successful in different steps of my life. Especially, I dedicate it to my mother, father, and brother, as well as to my wife who made me stronger with her mental support and enabled me to continue my path toward success.

Acknowledgments

First, I would like to thank my supervisors, Dr. Arzanpour and Dr. Park, for all their support and guidance during my Master's studies. Also, I would like to thank my friends in the Assistive Robotic Systems Laboratory for all their help.

Furthermore, I would like to thank my mother and father for years and years of effort for providing a suitable condition for my success, as well as my brother who always supported me in my education. Also, I need to thank my loving wife who supported me mentally during my studies and helped me in achieving my goals.

Finally, I need to thank all my current and previous friends, family, and supervisors who helped me to achieve my goals and supported me through hard conditions, especially my undergraduate supervisor, Dr. Aghil Yousefi-Koma.

Table of Contents

List of Tables

List of Figures

Chapter 1.

Introduction

Powered lower limb exoskeletons, also referred to as wearable robots, are an emerging technology that assist individuals with mobility disabilities. They can completely or partially carry the weight of the user and help them to stand and walk. People with mobility disabilities often are at a greater risk of secondary health conditions due to their inability to stand and walk. Some of these conditions may include: pressure sores, bowel or bladder problems, depression, obesity, fatigue, and pain. The users of mobility aids such as wheelchairs and scooters that substitute walking by providing a wheeled device on which the users sit are still at risk since they are still confined to seating, instead of standing and walking.

Another purpose of using an exoskeleton robot is for augmenting the strength of healthy adults in military or industry applications. For example, exoskeletons can be used to reduce energy consumption while walking which can help soldiers to walk longer distances. They can also assist factory workers in carrying heavy loads and reducing injuries and fatigue. In Sections 1.1 and 1.2 below, some of the most famous commercial and research exoskeleton robots are discussed.

1.1. Lower Limb Exoskeletons for Gait Rehabilitation and Locomotion Assistance:

Strokes, aging, accidents, etc. are some of the main reasons for having minor or major walking disabilities. Spinal cord injuries (SCI) caused by accidents can cause full lower-limb disabilities in children or adults. Powered lower limb exoskeleton robots can bring back the walking abilities to the paralyzed people. In these cases, the robot needs to carry the load of their user and have a predefined motion to help the user walk naturally again. ReWalk, ExoAtlet, HAL, etc. are some of the commercial or research exoskeleton robots with these capabilities. Some of the most famous and advanced lower-limb exoskeletons are presented in the following sections.

1.1.1. ReWalk

ReWalk is an Israeli commercial wearable lower-limb exoskeleton that utilizes powered actuators to enable paraplegics to regain standing and walking abilities. This robot which is developed by ReWalk Robotics was commercialized in 2011. The 4 degrees-of-freedom (DoF) mechanism of this robot helps the user to stand and walk using a solid load-bearing structure. A tilt sensor is implemented in the robot which signals the onboard computer when to take the next step [3]. Also, a wrist unit is provided in order to select settings for the robot’s functions. As can be seen in Figure 1.1 it is recommended to use walking crutches while wearing the device for balancing purposes.

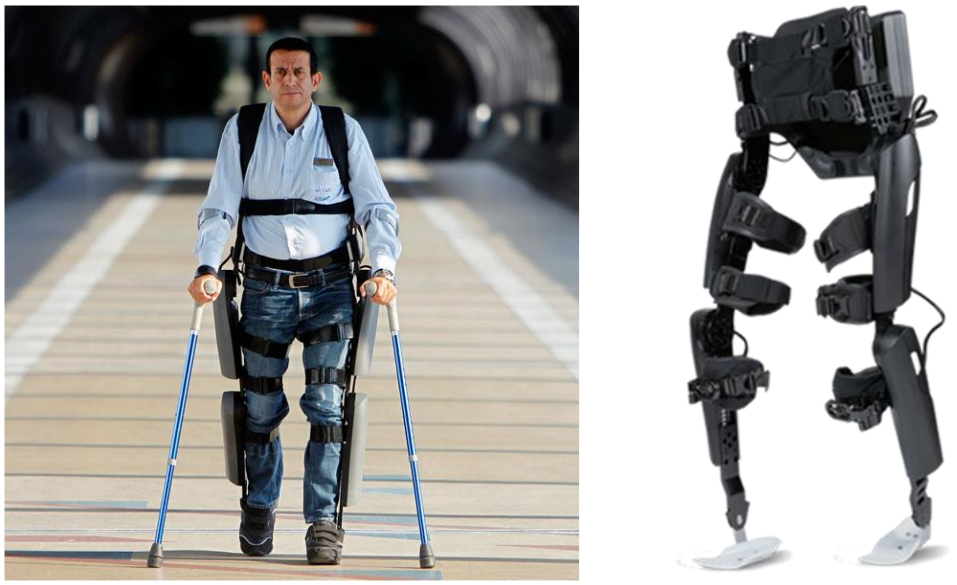


Figure 1.1: ReWalk lower limb exoskeleton

In 2012, Esquenazi et al. studied the safety and performance of ReWalk on 12 subjects with paraplegia due to SCI to carry out routine ambulatory functions [3]. The results showed that the subjects were able to safely participate in training sessions up to three times a week with no falls or occurrences of autonomic dysreflexia. Also, most of the subjects had improvement in their level of walking proficiency.

Moreover, Talayi et al. investigated the effectiveness of using ReWalk on 12 adults with chronic motor complete cervical and thoracic (C7-T12) SCI. It was a preliminary analysis on the difference of walking kinematics between subjects in order to understand possible improvement to the walking velocity range and walking ability by mimicking the better walkers [4]. Their results showed that the range of walking speed between the paraplegic users was mostly between 0.1 m/s to 0.2 m/s which is in the range of slow motion. Also, the lateral motion of the body which is required for robot walking can cause discomfort for a paralyzed user while walking with ReWalk robot.

Another study by Raab et al. in 2016 on an incomplete SCI case showed progress in walking ability using ReWalk. Also, QoL (Quality of Life), mobility, the risk of falling, motor skills and control of bladder and bowel functions were improved as well [5].

1.1.2. MINDWALKER

MindWalker is a full wearable lower-limb exoskeleton developed in the University of Twente which can empower paraplegics to stand and walk. Design, control and preliminary evaluation of this exoskeleton robot are presented in [6]. MindWalker utilizes series elastic actuators (SEA) in each joint which can deliver up to 100 Nm torque and 1 kW power. Also, a finite-state machine-based controller provides balance and gait assistance for the robot in both sagittal and frontal planes. In this robot, walking is triggered by displacement of the Center of Mass (CoM). Moreover, in [7], the series elastic actuator joint of MindWalker is analyzed in detail. This type of joint is normally used for torque control of an actuator by controlling the actuator’s position [8]. The MindWalker robot structure can be seen in Figure 1.2.

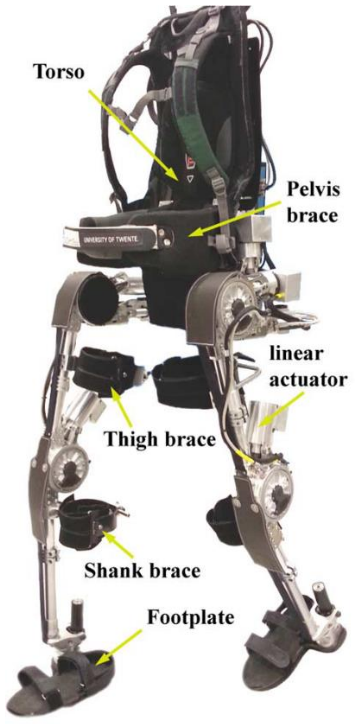


Figure 1.2: MindWalker lower limb exosekeleton

1.1.3. HAL

The Japanese Hybrid Assistive Limb (HAL) exoskeleton was developed in Tsukuba University together with Cyberdyne, which is a robotic company. The first prototype of this robot was presented in 1997 and in 2002 HAL-3 was developed only for leg function. Lee and Sankai in [9], described power assist control for walking aid with HAL-3 robot. They used EMG sensors on flexor and extensor muscles for acquiring the intention of the robot user in order to adjust impedance around the knee joint. This method was able to reduce amplitudes of EMG sensor data and the user was able to swing the leg lighter and easier. Also, knee strain could get alleviated by adding stiffness to the joints. Afterward, Kawamoto et al. proposed a method for motion and torque assistants to realize required power assist corresponding to the operator’s intention. They used phase sequencing control and a feedback controller for motion and torque assist respectively which resulted in effective power assist [10]. An extended version of this work was presented by Kawamoto and Sankai in [11].

HAL-5 robot which is a full body exoskeleton robot was presented as a commercialized exoskeleton robot in 2012. Modifications on this version with respect to previous versions include adding upper-body limbs, lighter and more compact power units and longer battery life [12]. Figure 1.3 shows pictures of this exoskeleton robot.

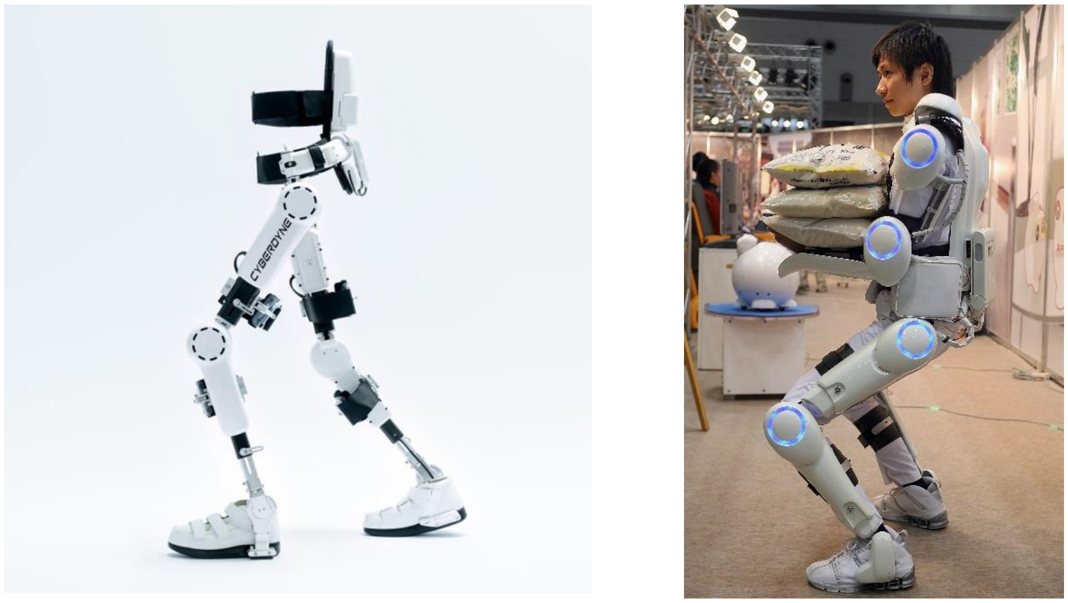


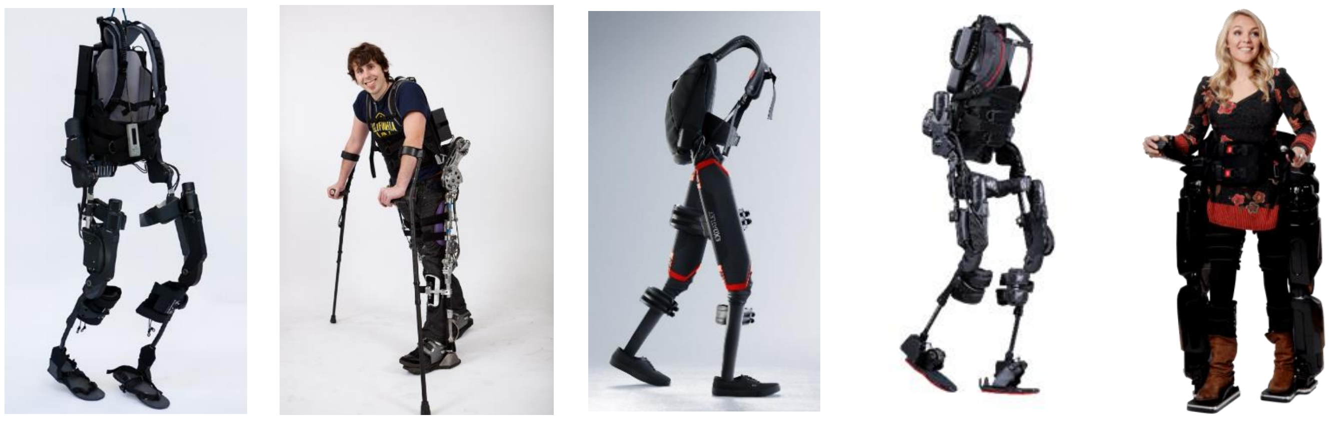
Figure 1.3: HAL exoskeleton robot

A case study from Crucige et al. on the impact of locomotion training with HAL exoskeleton robot on two subjects with severe chronic and therapy-resistant neuropathic pain due to chronic SCI showed beneficial impact of the neurologic controlled exoskeletal intervention on pain severity and health-related quality of life (HRQoL). They both had improvements in motor functions and walking abilities alongside with significant reduction in pain severity and improvements in all HRQoL domains [13].

It is required to use walking crutches while using most of the above-mentioned exoskeleton robots, except REX Bionics [14] and MindWalker. Their structural design and control algorithm allow their users to walk balanced without using crutches.

There are other exoskeleton robots in this category such as eLEGS [15], Austin [16], ExoAtlet [17], and Ekso GT [18]. The eLEGS and Austin exoskeletons are developed by Berkeley Robotics. ExoAtlet and Ekso GT are developed by ExoAtlet Company and Ekso

Bionics, respectively. All of these exoskeleton robots are claiming to have a lightweight structure with one or two DoF in their hip joint which is not enough to assist paraplegic users with all natural 3-DoF capability in their hip joint. Figure 1.4 shows some of the most famous exoskeleton robots that are designed for paraplegic users.



(a) (b) (c) (d) (e)

Figure 1.4: a) eLEGS, b) Austin, c) ExoAtlet, d) Ekso GT, e)REX bionics

1.2. Lower Limb Exoskeletons for Human Strength Augmentation

There is another type of wearable exoskeletons that are being used for augmenting healthy users for performing activities that need higher than normal body strength. Most of these robots are being used for carrying heavy loads in the areas that are too rugged or enclosed for vehicles to access, especially in military applications. The most famous exoskeleton robot in this category is the BLEEX exoskeleton developed at Berkley University. BLEEX, which was presented in 2003, has the capability of carrying the user’s weight and a load up to 34 kg and can walk at the average speed of 1.3 m/s. The structural design of this wearable robot while using hydraulic actuators is presented in Chu et al. [19], [20]. They used clinical gait data of human subjects in order to find suitable actuators for using in this robot. Also, the electrically actuated version of this robot is presented in Zoss and Kazerooni [21]. Figure 1.5, demonstrates this robot while being used to carry heavy loads.

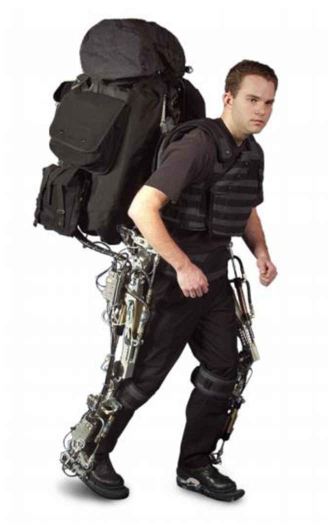


Figure 1.5: BLEEX exoskeleton robot

Control structure of BLEEX robot is presented and discussed in [22], [23] and [24]. The control algorithm developed for this robot increases the closed-loop system sensitivity to its wearer's applying forces and torques without doing any measurement from the wearer [22]. The robot controller uses the inverse dynamics of the exoskeleton as a positive feedback controller so that the loop gain for the exoskeleton approaches almost unity [23]. The dynamic model of the system needs to be relatively good because the control method has little robustness to parameter variations. Therefore, GHAN et al. designed a series of system identification experiments as well as determining the mass and inertia properties of the segments of the legs and various non-ideal elements, such as friction, stiffness and damping forces which resulted in a dynamic model significantly more accurate than the original model predicted from the designs of the robot [24]. Also, the improved control algorithm of this robot called "hybrid BLEEX controller" was presented by Kazerooni et al. in 2006 which added robustness to changing BLEEX backpack payload [25].

This team also developed another exoskeleton robot for military purposes called the ExoHiker. The advantages of this robot with respect to BLEEX were the substantial reduction in system weight, simplification in control, and increased load support capability [26].

ExoClimber is the name of another robot developed by UC Berkley team which has the capability of rapid vertical ascent while carrying heavy loads. Combination of ExoHiker and ExoClimber resulted in an advanced exoskeleton robot named Human Universal Load Carrier (HULC) which has the ability to carry 200lbs. Figure 1.6, shows ExoHiker, ExoClimber and HULC robots, respectively.

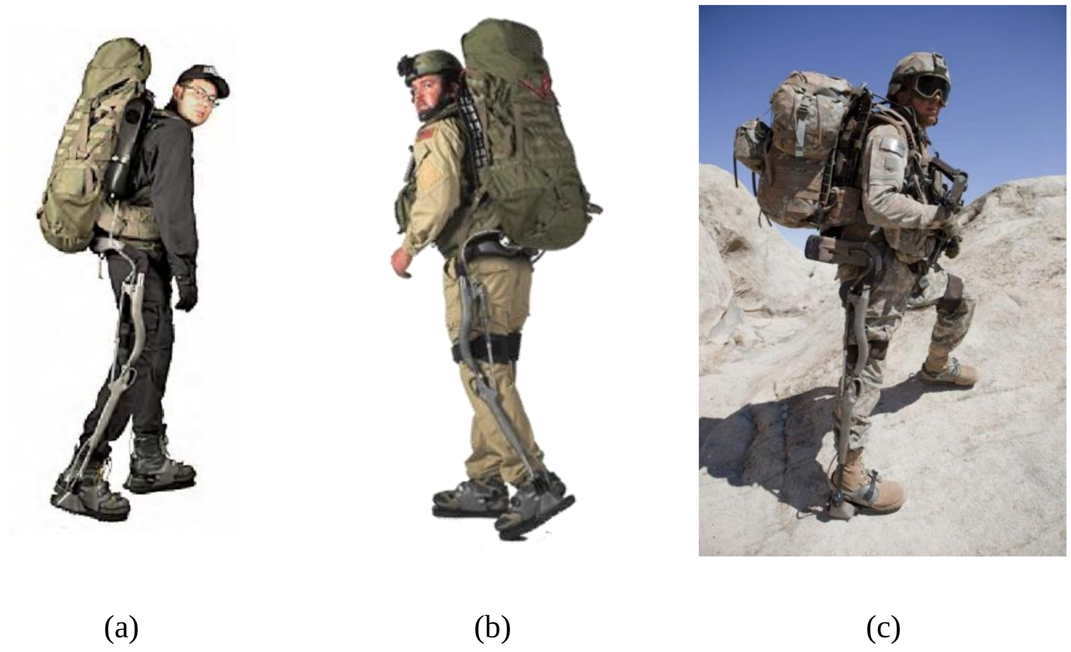


Figure 1.6: a) ExoHiker, b) ExoClimber, c) HULC

There are other exoskeleton robots in this category such as XOS, which has two versions. XOS 1 was developed by SARCOS Robotics at Utah University under DARPA funding. In 2007, SARCOS Research was acquired by Raytheon and the first generation XOS 1 system was publically announced in 2008. The second version of this robot called XOS 2 that can be seen in Figure 1.7, was unveiled and publically demonstrated in 2010 [27]. This robot that can carry loads up to 200lbs utilizes hydraulic actuators similar to BLEEX robot. Although, it has a lighter-weight structure with respect to previous exoskeleton robots in this category.

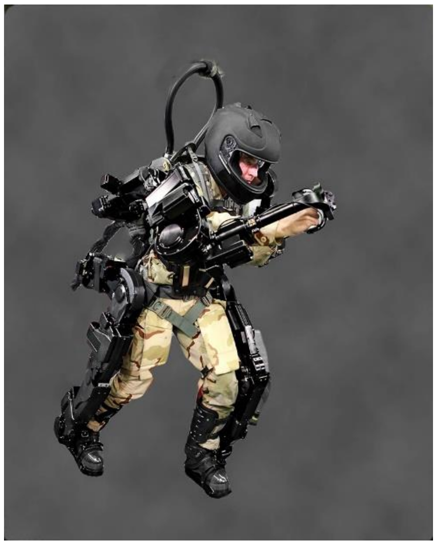


Figure 1.7: XOS 2 exoskeleton robot

1.2.1. Honda

During the past decade, Honda Company established a number of exoskeleton robots that can be used for healthy or elderly people. Their target was to use these robots in daily life in order to compensate minor walking problems or reducing the energy consumption of their users. The walking assist devices developed by Honda are “Bodyweight Support Assist” and “Stride Management Assist”.

The Bodyweight Support Assist robot which is indicated in Figure 1.8, is a user-friendly walking assist device that is designed for elderlies, production operation employees, etc. The robot reduces the ground reaction forces applied to the users’ sole in order to reduce muscle activities and consumed energy by the user [28].

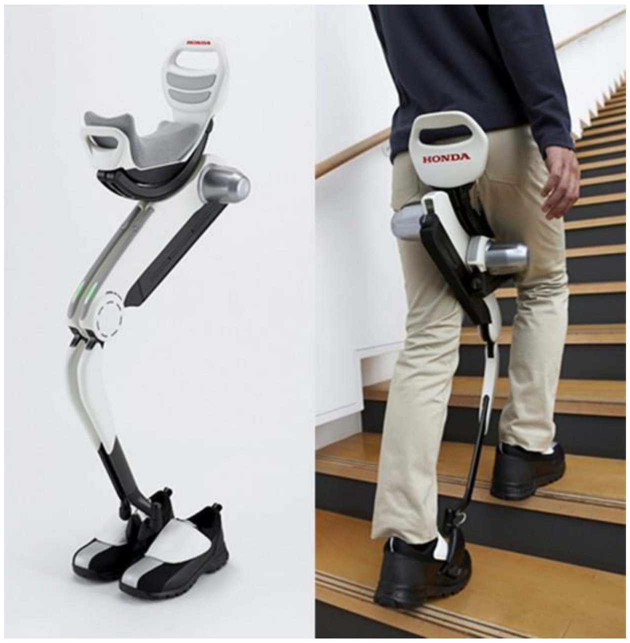


Figure 1.8: Honda’s bodyweight support assist exoskeleton robot

Another Honda product for walking assistance is “Stride Management Assis” which is shown in Figure 1.9. It utilizes a rotary motor in each hip joint to help those with weakened leg muscles walk easier and achieve longer strides while walking [29]. The main control loop actuates the motors based on hip encoder data during walking. It improves symmetry of user and robot motion during natural walking while each leg is lifting from the ground and extending forward.

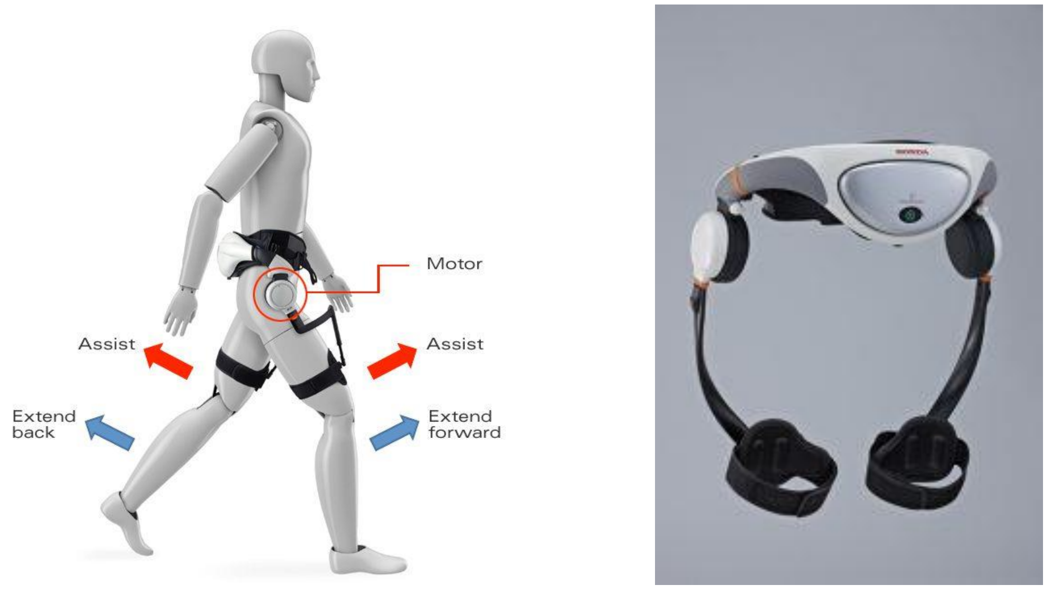


Figure 1.9: Honda’s stride management assist exoskeleton robot

In 2014, Kitatani et al. investigated the effects of Honda’s Stride Management Assist robot on energy expenditure during walking in healthy young adults [30]. The results showed 7.06% reduction in energy consumption while walking with comfortable walking speed and 10.52% while walking with maximum speed. Also, Buesing et al. investigated the effects of the stride management assist system on spatiotemporal gait characteristics in individuals after stroke [31]. Their results showed this robot can be a useful therapeutic tool to improve spatiotemporal parameters and contribute to improved functional mobility in stroke survivors.

There are several other powered or unpowered wearable robots which can be used for reducing energy consumption. As an example, Collins et al. presented an unpowered ankle exoskeleton that reduced around 7.2% of metabolic rate of human walking. They built a light weighted elastic device that worked in parallel with the user’s calf muscles, off-loading muscle force that reduced the metabolic energy consumed in contractions [32]. Also, in 2018 Nasiri et al. presented an unpowered exoskeleton which can reduce approximately 8% of metabolic rate of the robot user [33]. They used a torsional spring that applied torque as a linear function of the difference between two hips angles. They showed it can have a better effect than using a local spring that applies torque as a function of hip angle. A similar perspective is defined for our robot, but the criterion in this study is robot and user interaction forces instead of energy consumption.

1.2.2. Hip Exoskeleton Robot with Agile Eye Mechanism

In the Assistive Robotic Systems Lab at Simon Fraser University, a hip exoskeleton robot has been developed [34, 35]. It has the advantage of using a parallel mechanism called "Agile Eye" [36] in its structure which has the benefit of providing a full range of motion. This capability has a significant impact on users' motion and helps them to walk naturally, especially in circular paths. Figure 1.10 shows pictures of the device worn by a healthy adult.

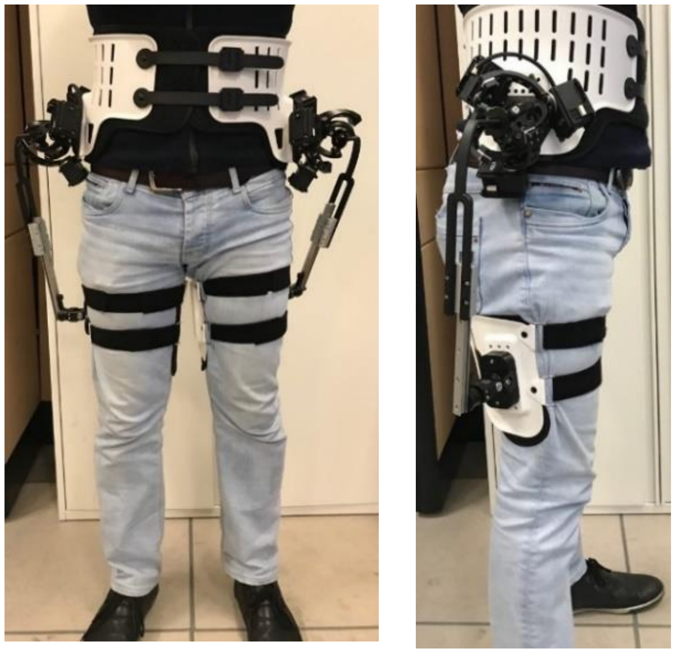


Figure 1.10: Hip exoskeleton robot with agile eye mechanism

One of the goals of this project is to design a preliminary control structure for this robot to be used by healthy users or elderlies. The robot is required to assist the user to walk easier for longer distances similar to Honda's Stride Management Assist by applying force on the upper thigh.

One of the most important required data for designing control loop of most exoskeleton or biped robots is Ground Reaction Forces and Moments (GRF/M) which can be acquired using force sensors implemented under user's sole. However, the attachment of this type of sensors to each user's sole is impractical due to wiring or mounting problems. Also, the accurate and advanced versions of this type of sensors are expensive, which leads to the increased cost of the exoskeleton. For example, the attachment of the sensors such as a 6-axis force/moment sensor can cause unnatural walking due to the sensor's thickness. An alternative is the use of pressure sensors such as in-shoe vertical pressure sensors [37]. However, they can only provide data in the vertical direction (F_z) which may not be enough for a full lower-limb exoskeletons control structure. Another method is using machine learning algorithms for acquiring GRF/M data based on human body motion. In [38], Joo et al. predicted ground reaction forces and moments based on plantar pressure (PP) data obtained from insole type measurement devices. Moreover, in 2013, Oh et al. estimated GRFs in single support phase based on traditional Newtonian mechanics and used artificial neural network for double support phase [39]. Also, Sim et al. used a wavelet neural network to construct a model between GRF/M and insole plantar pressure sensors [40]. Using this method for GRF/M data acquisition eliminates an expensive force sensor from robot structure which can reduce manufacturing costs of exoskeleton robots. Also, it reduces wiring complexity and discomforts of using force sensors under the users' soles.

1.3. Objectives and Contributions

1. In this project, the main objective is to predict GRF/M for the hip exoskeleton users walking in varying speeds using inertial sensors instead of force/torque sensors. The outcome of this work can be used for acquiring all components of GRF/M or for gait phase estimation based on inertial data acquired from the exoskeleton user's lower limb motion. For this purpose, different regression methods such as Neural Network, Random Forest and Support Vector Machine were employed. Currently, there is no exoskeleton available that utilizes machine learning algorithms for estimating GRF/M without any force sensors attached to the foot. Hence, the main advantage and contribution of the proposed approach is the elimination of the force/torque sensors from design and control structure of lower limb exoskeletons,

ultimately leading to reduced production costs and a simplified control loop structure.

2. Another objective and contribution of this thesis are a feasibility study and successful experimental demonstration of using the estimated GRF/M in an online control loop structure of a wearable hip exoskeleton. The trained models with machine learning algorithms were applied to the online generation of motion for the hip exoskeleton in order to push the user's upper-thigh in assisting with walking. The effectiveness of the generated motion was compared with two other cases: when device is off and when the device is following user's motion based on Inverse Kinematics (IK).

1.4. Thesis Structure

The remainder of the thesis is structured as follows:

- In Chapter 2, an overview of different possible methods for GRF/M estimation is presented. Also, a summarized description of machine learning algorithms used in this research study is provided. Afterward, similar studies in the literature which used these methods for estimation of GRF/M are discussed.
- In Chapter 3, the experimental setup and protocols, as well as data collection devices and software, are presented.
- Chapter 4 presents the results of the trained models with different machine learning algorithms and compares their accuracy with each other. Also, the resulted accuracies from the trained models are compared with similar studies in the literature.
- Implementation of trained models in the control structure of the hip exoskeleton robot for motion generation is described in Chapter 5.
- Chapter 6, summarizes outcomes of this study and recommendations for future studies of this research are presented.

Chapter 2. Background

In this chapter, first, we talk about methods for ground reaction forces and moments estimation for a walking person. Afterward, the machine learning methods used in this project are introduced and discussed. At the end, advantage and disadvantage of these methods are explained.

2.1. Ground Reaction Forces and Moments Estimation

In several biped or exoskeleton robots, ground reaction forces and moments data are necessary information for designing the control structure of the robot. Also, they can be used for balancing the robot while walking, sitting and standing, etc. In some cases, it would be beneficial not to use any type of force sensors under the sole, because they can cause complexity for the mechanical structure of the robot or may irritate the user while wearing the robot. Also, these type of sensors are usually expensive (especially 6 axis force sensors). Therefore, it would be beneficial if we could estimate GRF/M without using any kind of sensors under the sole of the robot's users.

There are different methods available in the literature for estimating the GRF/M online during walking [41]. In summary, the GRF/M estimation can be achieved by: (i) kinematic information of the whole body using wearable inertial measurement units (IMU) [42, 43], (ii) camera-based motion measurement systems [44] or (iii) plantar pressure data provided by force plates [45], insole force sensors [46] or pressure sensors [47, 38]
In addition, Jung et al. developed an adjustable foot-ground contact model to estimate the GRF/M [48]. The purpose of their study was to estimate the GRF during gait by utilizing distance and velocity-dependent force models between the foot and ground in an inverse-dynamics-based optimization. Also, in 2014, Wille et al. used sagittal kinematic variables to estimate GRF and joint kinetics [44]. However, this method is not applicable for our purpose as we need all GRF/M components for closed-loop control of the proposed hip exoskeleton.

Another method for estimating the GRF/M is using cost-effective sensors such as Force Sensitive Resistors (FSR) implanted under the user’s soles [49]. However, the exoskeletal application of this method is limited due to the fact that this type of sensors are highly noisy and inaccurate.

More recently, Rosquist et al. used a tenfold cross-validation process to calibrate a separate mathematical model for each component to estimate GRF/M based on data collected from nanocomposite piezo-responsive foam while walking [50]. Also, there are other studies that only estimated vertical GRF which may not be enough information for closed-loop control system design of many exoskeletons. Guo et al. used a new proxy measurement algorithm to estimate the vertical GRF using wearable sensors [51].

A different method for finding GRF/M is using machine learning algorithms to estimate all the components with high accuracy. The machine learning algorithm can be applied on data acquired from other types of sensors and using them GRF/M data can be estimated.In [52], Joo et al. predicted 6-axis ground reaction forces and moments based on plantar pressure (PP) data obtained from insole type measurement devices. Moreover, in 2013, Oh et al. predicted GRFs in single support phase based on traditional Newtonian mechanics and used artificial neural network for predicting GRFs in double support phase [53]. Also, Sim et al. used a wavelet neural network to construct a model between GRF/M and insole plantar pressure sensors [46]. Most of the mentioned methods provide accurate enough results which can be used in control loop structure of an exoskeleton robots. In this project, three machine learning methods are used for estimating GRF/M of a user while wearing a hip exoskeleton robot and walking in a straight line. Based on data provided in literature, artificial neural networks are one of best methods for this purpose. Also, support vector machine(SVM) algorithm is an accurate method for solving regression problems. Random forest is another accurate and fast method that can provide high accuracy in regression problems which has never been used before for estimation of GRF/M. In this project these three methods are being used and compared to find most suitable method that can be used in control structure of exoskeleton robots. In the next section, the estimated forces are going to be used for controlling the robot and helping the user to apply less torque while walking.

2.2. Random Forest

Random forest or random decision forest is a machine learning method for classification or regression that operates by constructing several decision trees while training a model and outputting a class for classification or mean prediction of the individual trees for regression method [54, 55, 56]. Figure 2.1 shows a general RF for regression purpose.

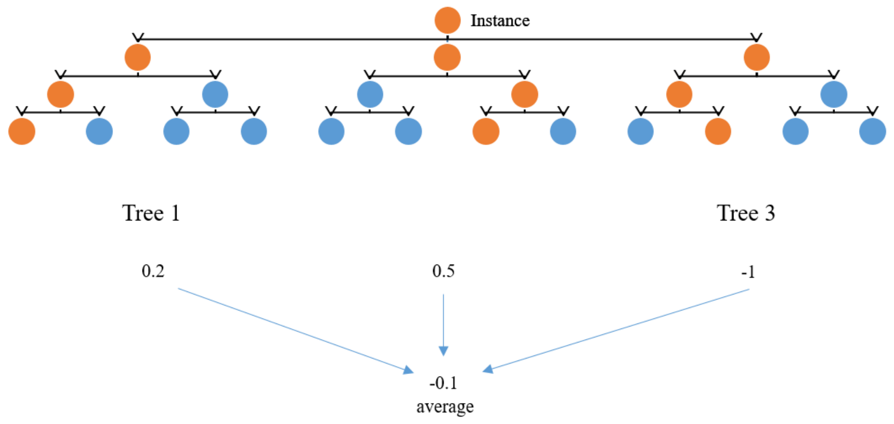


Figure 2.1: General structure of a regression random forest

It has the advantage of correcting overfitting to their training set with respect to decision trees. It only optimizes two parameters which are the number of variables in the random subset at each node to split and the number of trees in the forest while it is not very sensitive to these parameters [57, 58].

Consider a dataset that has N data points, and each one is constructed from M features. By the method developed by Breiman in [57], in order to construct each tree, a random subset of the samples including N' data points is selected. Afterward, each node is divided by the best guess among a random subset of the features. So instead of using all M features to make a decision at each node, M' features are used for making the decision. The majority of the votes for classification and averaging for regression in the whole forest are being used for predicting a new sample [59].

This method has some advantages and disadvantages [60]. Advantages include the following:

- There is no need for feature normalization, so it reduces the complexity of using this method
- Individual decision trees can be trained in parallel
- They reduce overfitting, therefore it can be used for many test subjects

Also, the disadvantages of this method include:

- This method is not easily interpretable
- They’re not a state-of-the-art algorithm

2.3. Support Vector Machine (SVM)

SVMs are supervised learning models that are first identified by Vladimir Vapnik and can be used as a regression method [61]. This method constructs a hyperplane or set of hyperplanes in a high or infinite-dimensional space for both classification and regression problems. In SVM classification, the algorithm is separating different classes of data based on a subset of data. Although, the Regression Support Vector Machine (RSVM) has minor differences with SVM for classification. First of all, the number of possibilities for an answer increases significantly, because the output is a real number. Therefore, a margin of tolerance is needed which will be defined by the user. Also, the algorithm itself is more complicated.

For classification, SVM uses a Kernel function to transfer the input data to a kernel space to find a linear relationship between input and target values. Due to relying on different kernel functions (i.e. linear, polynomial, radial basis function (RBF), or sigmoid), it can be adapted to many different problems [62, 63]. An important advantage of SVMs is that the determination of the model parameters corresponds to a convex optimization problem. Therefore, local solutions are also global optimums. Although, training a model on a dataset with many subsets needs a long time, which makes it impossible to use for some applications.

2.4. Neural Network

One of the most useful methods for solving a regression or classification problem is neural network method. It is mainly because this method can be trained easily and usually provides high accuracy. The neural network models consist of an input and an output layer and one or more hidden layers of nodes between the input and output which construct the relation between them. The general structure of a fully connected neural network model is presented in Figure 2.2.

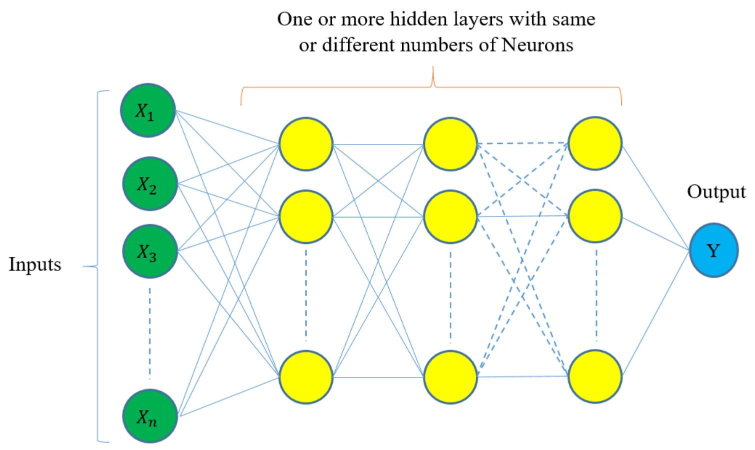
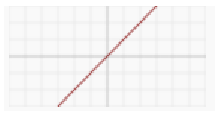
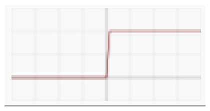
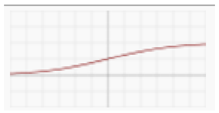
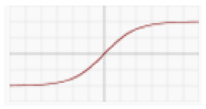
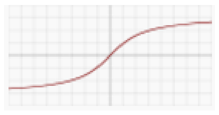
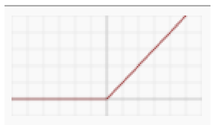
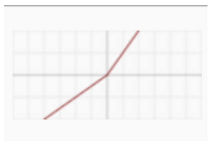
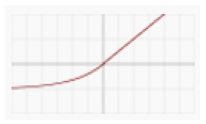
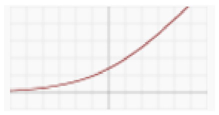

One or more hidden layers with same or different numbers of Neurons
Inputs
X_1
X_2
X_3
X_n
Output
Y

Figure 2.2: General structure of a fully-connected neural network model with hidden layers

The nodes of the input layer are features which construct each sample and the outputs are the target of the model. The output of each layer multiplied by weight vectors construct inputs of the next layer. Each layer has a specific activation function which can be similar to other layer's activation functions. The activation function of a node defines the output of that node given an input or set of inputs. These functions are a very important feature of the neural network models because they are making the output of each layer which eventually results in the output of the whole models. There are different activation functions available and following table shows the main and most useful activation functions.

Table 2.1: The most common activation function employed in artificial neural networks

Name	Plot	Equation	Derivative
Identity(Linear)		$f(x) = x$	$f'(x) = 1$
Binary Step		$f(x) = \begin{cases} 0 & for\ \ x < 0 \\ 1 & for\ \ x \geq 0 \end{cases}$	$f'(x) = \{0\ for\ x \neq 0$
Logistic(Sigmoid)		$f(x) = \dfrac{1}{1 + e^{-x}}$	$f'(x) = f(x)(1 - f(x))$
Tanh		$f(x) = \tanh(x) = \dfrac{2}{1 + e^{-2x}} - 1$	$f'(x) = 1 - f(x)^2$
ArcTan		$f(x) = \tan^{-1}(x)$	$f'(x) = \dfrac{1}{x^2 + 1}$
Rectified Linear Unit(ReLU)		$f(x) = \begin{cases} 0 & for\ \ x < 0 \\ x & for\ \ x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & for\ \ x < 0 \\ 1 & for\ \ x \geq 0 \end{cases}$
Parametric Rectified Linear Unit(PRELU)		$f(x) = \begin{cases} \alpha x & for\ \ x < 0 \\ x & for\ \ x \geq 0 \end{cases}$	$f'(x) = \begin{cases} \alpha & for\ \ x < 0 \\ 1 & for\ \ x \geq 0 \end{cases}$
Exponential Linear Unit(ELU)		$f(x) = \begin{cases} \alpha(e^x - 1) & for\ \ x < 0 \\ x & for\ \ x \geq 0 \end{cases}$	$f'(x) = \begin{cases} f(x) + \alpha & for\ \ x < 0 \\ 1 & for\ \ x \geq 0 \end{cases}$
SoftPlus		$f(x) = \log_e(1 + e^x)$	$f'(x) = \dfrac{1}{1 + e^{-x}}$

The activation functions in Table 2.1 are the most common and useful ones that have been tested in this project for the GRF/M modeling. Some desirable properties in an activation function include:

- Nonlinearity: If an activation function is non-linear, then a two-layer neural network is a universal function approximator [64]. It gives the network more ability and accuracy with the same amount of training data. The identity (linear) activation function does not satisfy this property. If multiple layers in a neural network model use the identity activation function, the entire network is basically equivalent to a single-layer model.
- Range: If the range of an activation function is finite, neural network gradient-based training algorithms tend to be more stable, because pattern presentations significantly affect only limited weights. But if it is infinite, training is generally more efficient because pattern presentations significantly affect most of the weights.

There are other properties such as “continuously differentiable”, which is desirable for enabling gradient-based optimization methods [65] or being “Monotonic”, which makes the error surface associated with a single-layer model in the form of a convex [66]. The proper activation functions for this study are presented and discussed in section 4.1.

Some advantages of ANN include:
It has the ability to learn non-linear and complex datasets. It is an important feature because in real life, many of the relationships between inputs and outputs are non-linear and complex. This featured is helpful for training models on data used in this study as they are non-linear. Moreover, this method doesn’t make restrictions on the distribution of input data [67].
The most famous disadvantage of this method is that it has a black box nature. It means you can not understand how and why some specific result is provided by the neural network model [68].

Chapter 3. Experimental Setup and Protocol:

This chapter explains experiments for modeling ground reaction forces and moments based on the lower limb of human body motion while wearing the hip exoskeleton robot and walking in a straight path. For this purpose, required material and information about the participants, instruments and protocols are provided. Afterward, data analysis based on the mentioned methods in Chapter 2 is discussed. In the end, the results of the modeling with different methods are provided and compared. In order to validate the accuracy of the resulted models, some of the recent studies' results are compared with them.

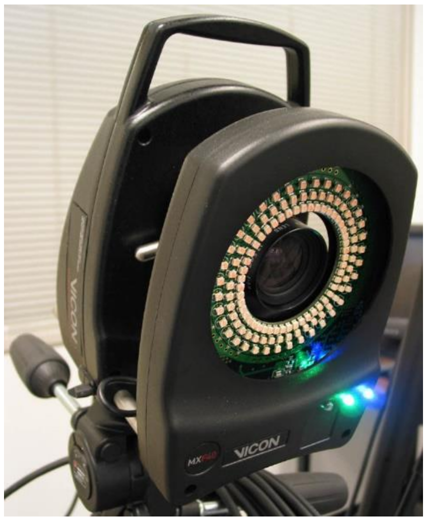
3.1. Material and Methods

3.1.1. Participants

Three females (age: 27 ± 2 years, height: 161.5 ± 4.5 cm, and weight: 58.5 ± 11.5 Kg) and seven males (age: 26.5 ± 2.5 years, height: 180.5 ± 11.5 cm, and weight: 93 ± 23 Kg) were recruited for this study. The experiment protocol was approved by the Research Ethics Board at Simon Fraser University and all participants provided informed written consent. They were all healthy young adults with the ability to walk naturally and without any assistance.

3.1.2. Instrumentation

Walking motion of the right leg of each participant was acquired by using 7 Vicon MX-40+ motion capture cameras (Vicon Motion Systems Ltd., UK). The cameras were positioned around the participant's body and it was checked that it is possible to observe each marker at least with three cameras. Ground reaction forces and moments were collected by force sensors implanted in a Bertec's Fully Instrumented Treadmill (FIT) (Bertec Corporation, USA). Both cameras and force sensors were set to acquire data at 120Hz and a voltage pulse was collected from the camera system at the start of each experiment in order to synchronize the force and motion data for better accuracy. Figure 3.1 shows Vicon MX-40+ and Bertec's FIT treadmill equipped in this experiment.


(a)

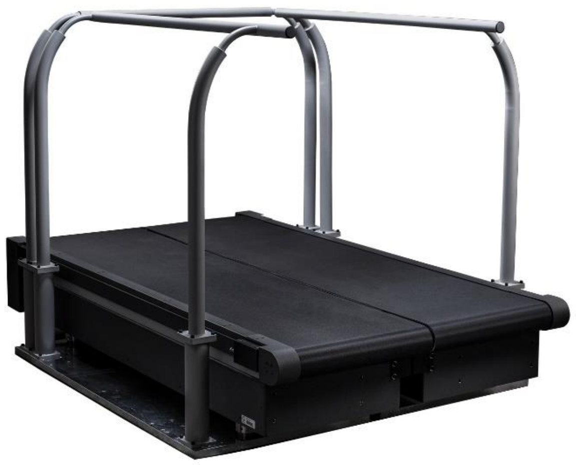

(b)

Figure 3.1: a) A Vicon MX-40+ camera b) Bertec's Fully Instrumented Treadmill (FIT) treadmill

Specification of the Vicon MX-40+ and Bertec's Fully Instrumented (FIT) treadmill are provided in Table 3.1 and Table 3.2.

Table 3.1: Vicon MX-40+ specification

Imager	CMOS
Aspect Ratio	4:3
Pixel Size	7 microns x 7 microns
Photosensitive Pixels	2352 H x 1728 V
Sensor Size	16.46 mm (H) x 12.10 mm (V), 20.43 mm (Diagonal)
Sensor Dynamic Range	59 dB
Digital Responsivity	Monochrome 2500 bits per luxsecond @ 550nm ADC ref @ 1V
Lens Mounts	C- and SLR-mount options
Size (with 20 mm SLR lens)	215 mm (H) x 138 mm (W) x 255 mm (D)
Weight (with 20 mm SLR lens)	2.6 kg
RoHS compliant	Yes (MX40+ camera)

Table 3.2: Bertec's Fully Instrumented (FIT) treadmill specification

Size of Each Belt	1.75 × 0.5 m
Maximum Load Range	$F_X, F_Y: 2500\ N\ \ , F_Z: 5000\ N$
Speed Range	0-24 km/h
Acceleration	0-25 m/s^2

3.1.3. Experimental Protocols

In order to find ground reaction forces and moments, it is necessary to have data from different subjects at different walking speeds. Therefore, the participants were asked to walk on Bertec's fully instrumented treadmill (FIT) (Bertec Corporation, USA) with 0% inclination. The trials included walking at 0.2 m/s, 0.4 m/s, 0.6 m/s, 0.8 m/s, 1 m/s, 1.2 m/s and 1.4 m/s for 4 minutes each. This range of speed helps the models to estimate GRF/M for a wider range of walking speeds. Although, in order to reduce the effect of fatigue in the results, the speed of each test was chosen randomly for each participant. For example, for one of the participants the tests were done in 0.8 m/s, 0.4 m/s, 0.2 m/s, 1 m/s, 1.2 m/s, 0.6 m/s. If this randomness didn't exist while testing participants, then all participants were tired on 1.2 m/s speed and the provided data may have not been realistic.

Figure 3.2 shows the markers placements on a participant as well as the placement of the hip exoskeleton. As you can see markers are attached on different points of the right leg. In order to find lower limb joint angles of each user, three markers are attached on the upper thigh, two on the lower thigh and three others are attached to the foot. Knee joint, ankle flexion-extension and toe joint angles were calculated based on the position of these markers. Also, Figure 3.3 shows motion capture cameras positioning in the lab while recording motion data from markers.

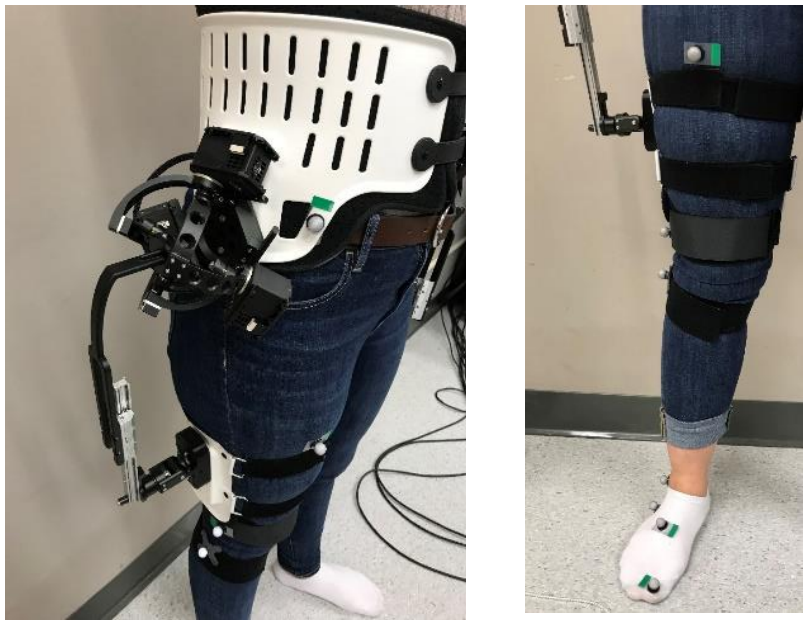


Figure 3.2. Optical markers placements while wearing the hip exoskeleton

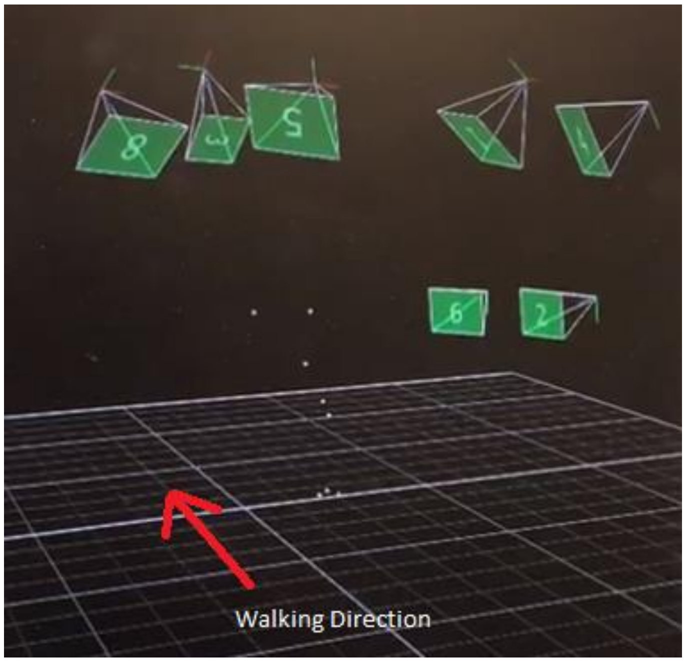


Figure 3.3. Position of motion capture cameras while data acquisition

After data acquisition and data post-processing, it is necessary to find the joints angles based on the 3D position of the markers. Therefore, using simple angle calculation based on the attached markers one the body, knee, ankle flexion-extension, and toe angles can be acquired.

Figure 3.4, shows the obtained angles for the knee, ankle flexion/extension and toe joints over two gait cycles. It should be noted that all toes are considered as one joint and the angle is considered the same for all toes. The obtained results agree with real data available in reference books [69]. Figure 3.5 and Figure 3.6 show the three components (F_x, F_y and F_Z) of the ground reaction forces and the three components (M_x, M_y and M_Z) of the ground reaction moments, respectively, obtained from the Bertec treadmill.

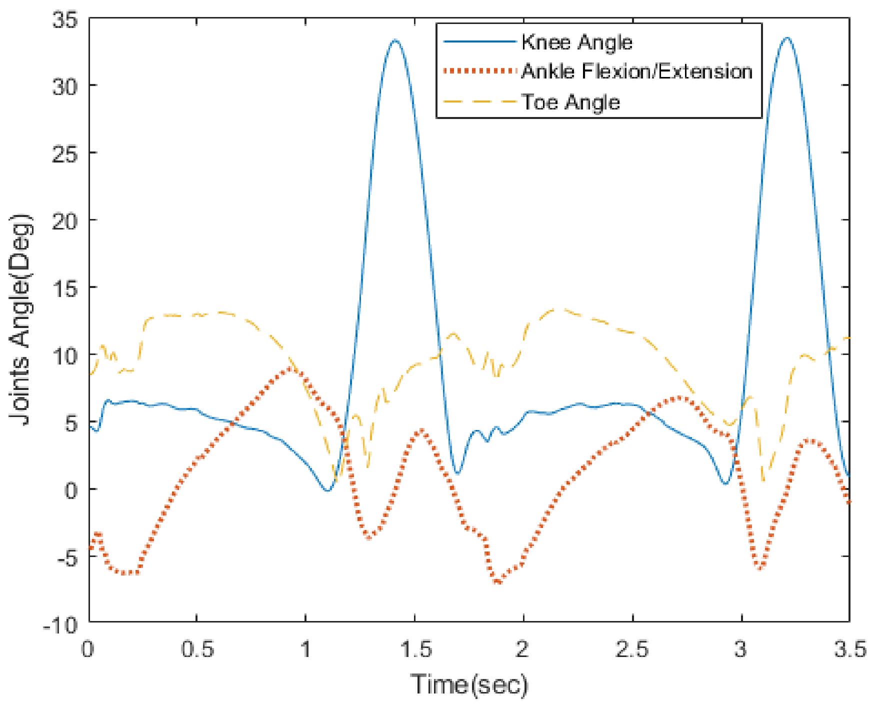


Figure 3.4. Knee rotation, ankle flexion/extension and toe rotation of a random participant walking with 0.4 m/s (slow) speed

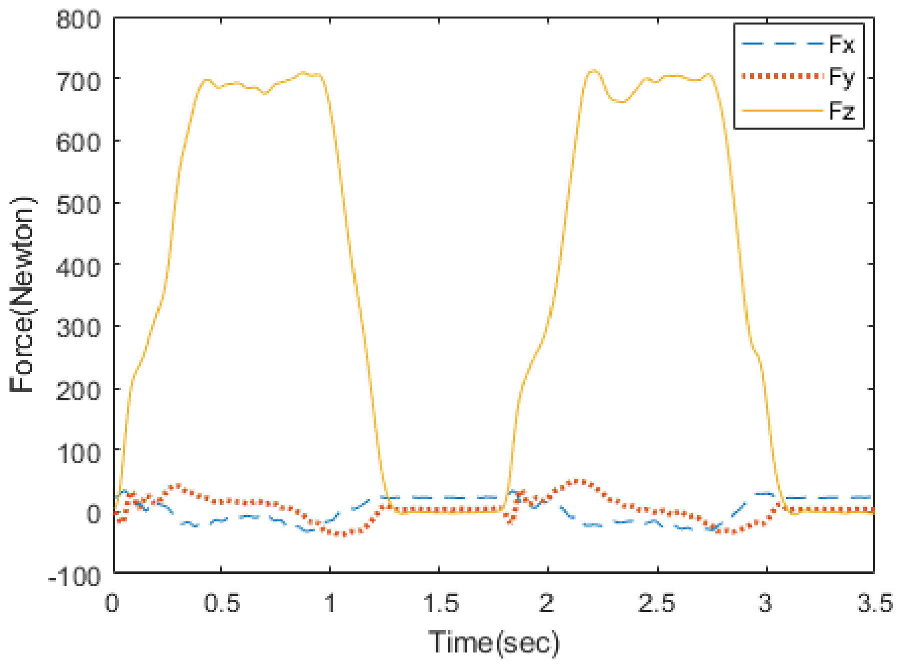


Figure 3.5. Ground reaction forces components acting on a random participant sole while walking with 0.4 m/s (slow) speed

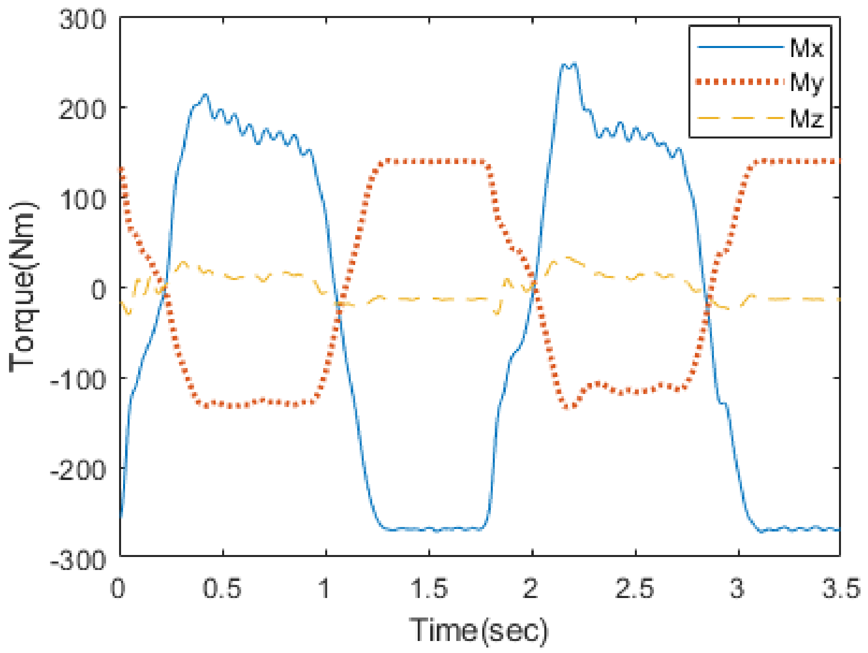


Figure 3.6. Ground reaction moment components acting on a random participant sole while walking with 0.4 m/s (slow) speed

Figure 3.7 shows the increasing maximum value of the F_Z ground reaction forces as the gait speed increases from 0.4 m/s to 1.4 m/s.

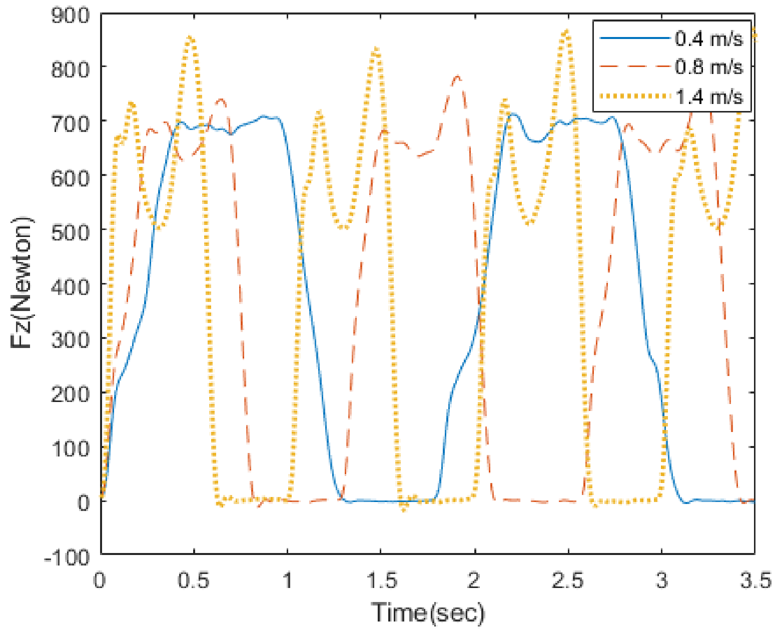


Figure 3.7. Comparison between F_Z in different walking speeds (from slow to high speed)

3.2. Data Analysis

In order to find a suitable model for lower body motion and GRF/M, three different methods are being used. Neural network, random forest and support vector machine methods were tested to find the most accurate method for this study. As can be seen in Figure 3.5 and Figure 3.6 the raw force data is noisy. Therefore, before using the raw data in the model, a fifth-order Butterworth filter with 8 Hz cut-off frequency was used to make the force and moment data less noisy and suitable for modeling. Although this filtering may clear some range of GRF/M data, it will increase the model's accuracy Figure 3.8 and Figure 3.9 show the comparison between ground reaction forces and moments’ raw data and filtered data.

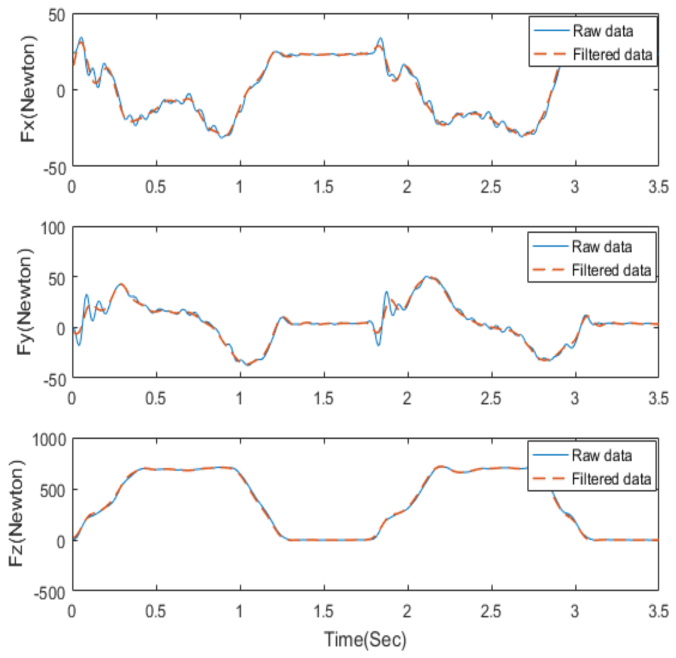


Figure 3.8. Comparison between ground reaction forces' raw data and filtered data

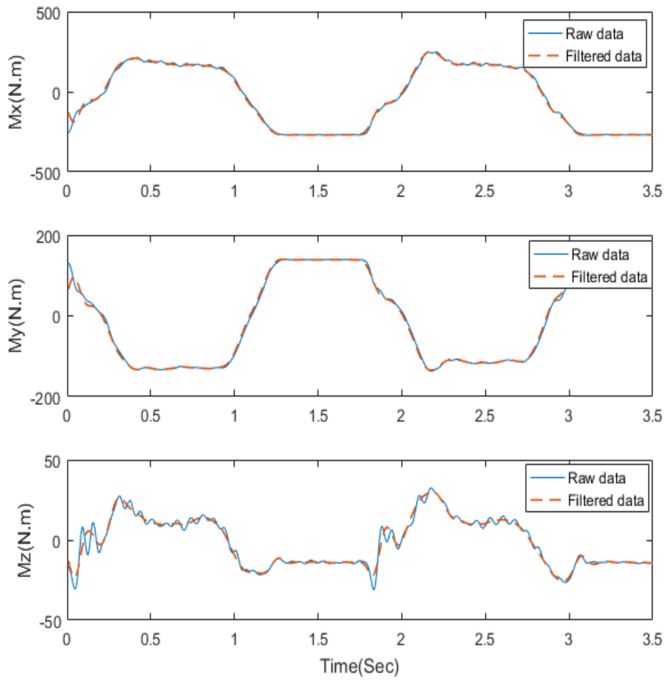


Figure 3.9. Comparison between ground reaction moments' raw data and filtered data

In order to increase the accuracy of these models, it is better to have the effect of dynamics and motion in the input data. Therefore, “Angular Velocity” and “Angular Acceleration” of the knee, ankle flexion/extension and toe joints are considered as input as well. As a result, each model can have 12 inputs and 1 output which is one of the components of GRF/M. You can see the input features in Table 3.3.

Table 3.3: Inputs of machine learning models

Inputs	1-Walking Speed (m/2)
	2-Weigh(lbs)
	3-Height(cm)
	4-Knee Angle (deg)
	5-Knee Angular Velocity (deg/sec)
	6-Knee Angular Acceleration (deg/sec^2)
	7-Ankle Angle (deg)
	8-Ankle Angular Velocity (deg/sec)
	9-Ankle Angular Acceleration (deg/sec^2)
	10-Toe Angle (deg)
	11-Toe Angular Velocity (deg/sec)
	12-Toe Angular Acceleration (deg/sec^2)

Walking speed, weight and height of the user are constant inputs and joint angle, angular velocity and angular acceleration of knee, ankle flexion/extension and toe joints’ recorded data of the user while walking are variable inputs to each model.

In order to better see the effect of dynamics in the model and increase its accuracy, the data is employed in batches for model training. It means each subset of data is combined with 20 previous ones. Therefore, the change in each input can be seen better by models. By combining 21 steps of data together and clearing similar features, such as speed, weight, and height the number of inputs changes from 12 to 192.

Chapter 4. GRF/M Estimation Using Machine Learning Algorithms

In this chapter detailed information about machine learning algorithms used for training models on the data is provided. Also, results of each method are presented and compared with each other. At the end, bests resulted accuracies of this study are compared with some recent similar studies in GRF/M estimation.

4.1. Neural Network

In order to estimate the six components of GRF/M using neural network models, six separate fully connected artificial neural networks (ANN) are used. The modeling is performed in Python 3.6.3 using Keras and TensorFlow libraries. Each neural network model contains one input layer with 192 input features and one output that is one of the GRF/M components. Also, five hidden layers with 300 neurons are used. For modeling F_Z component, the activation functions of all hidden layers are "Rectified Linear Unit" (RELU) due to its higher accuracy and less noise. For modeling the remaining components, "Sigmoid" function is used as the activation function for all layers except the last one. The last layer output needs to have a "linear" activation function to be able to perform accurately and reduce noises on output data. The combination of activation functions and the number of layers and neurons are chosen by try and error. Figure 4.1 shows a schematic of the neural network structure used in this study. In addition, Table 4.1 shows the summarized information for the training models. The models use error backpropagation to improve the accuracy.

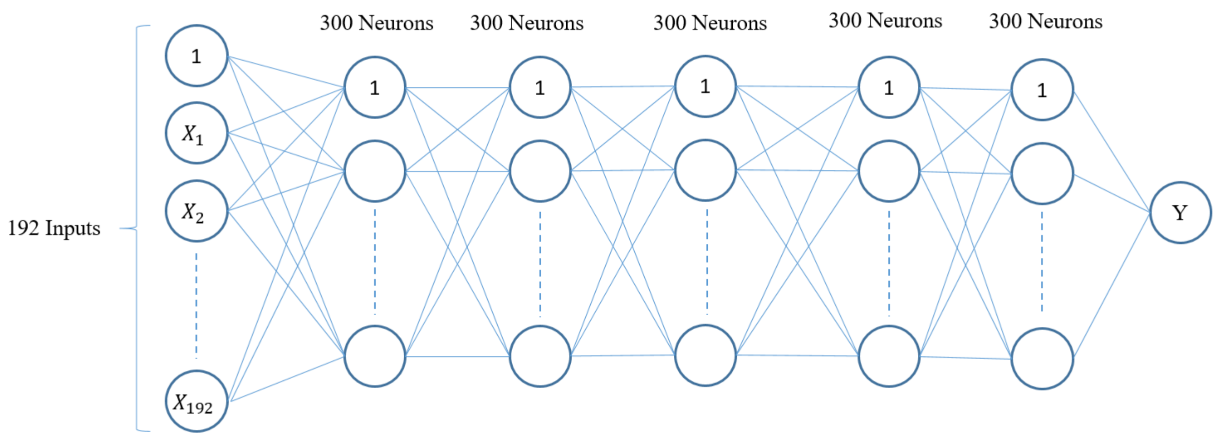


Figure 4.1. Structure of the employed neural network model

Table 4.1: Summarized information of trained artificial neural network models

Items	Detail	
Subjects	10 healthy young adults (7 men, 3 women)	
Training Data	Data from randomly selected 8 subjects: 192 inputs, 6 outputs (F_x, F_y, F_z, M_x, M_y, Mz) for a total of ~200000 entries	
Validation Data	15% of recorded data from each participant for a total of ~ 35250 entries	
Test Data	Data from 2 remaining subjects that are not part of the training data for a total of ~55000 entries	
ANN Structure	1 input layer	192 nodes
	5 hidden layers	1st layer, 300 nodes(Sigmoid Activation Function) 2nd to 5th layer, 300 nodes(Sigmoid Activation Function, with Bias) (Note: Activation Function only for F_z is Rectified Linear Unit)
	1 output layer	1 output (Linear Activation Function)
ANN properties	Using Keras library in Python 3.6, 80 Iterations, Learning Rate=0.001, Optimizer = 'rmsprop'	

4.2. Random Forest (RF)

Another machine learning method that was employed to estimate the GRF/M was RF. The training, validation and testing data used for this method were the same as the data used for training the NN models. In this method, the algorithm constructs a multitude of decision trees and by changing the combination and branch structure of the tree, it tries to output the mean prediction of the individual trees. Similar to NN method, the GRF/M data was filtered with Butterworth filter with 8 Hz cut-off frequency filter. Also, a similar filter was used on the angles data provided by motion capture cameras. In this study, six different RF models were constructed for each output. Each model contained 150 trees that have enough diversity for finding the most accurate model. Further details about the RF regression algorithm can be found in [70]. Table 4.4 shows the summarized information for the RF trained models.

Table 4.2: Summarized information of trained random forest models

Items	Detail
Subjects	10 healthy young adults (7 men, 3 women)
Training Data	Data from randomly selected 8 subjects: 192 inputs, 6 outputs (F_x, F_y, F_z, M_x, M_y, Mz) for a total of ~200000 entries
Test Data	Data from 2 remaining subjects that are not part of the training data for a total of ~55000 entries
RF Properties	150 trees per forest, Using MATLAB 2017b

4.3. Support Vector Machine

This method constructs a hyper-plane or set of hyper-planes in a high or infinite dimensional space for both classification and regression problems. Due to relying on different kernel functions (i.e. linear, polynomial, radial basis function (RBF), or sigmoid), it can be adapted to many different problems. The suitable kernel function for this study is the radial basis function (RBF) that can be seen in equation (4.1) and it is necessary to tune

RBF variables. Therefore, using regression support vector machine toolbox on MATLAB 2016b, ε, Kernel Scale and Box Constraint variables are optimized and a coarse grid-search is used which increases exponentially (i.e. 2^{-5}, 2^{-4}, 2^{-3},…, 2^3,2^4,2^5). Moreover, a 10-fold cross-validation is performed in order to increase the accuracy.

$$K(x,x') = \exp(-\frac{\|x-x'\|^2}{2\sigma^2}) \qquad (4.1)$$

The data used for the SVM method was less than those of the NN and RF algorithms: 70500 entries as training data and 16500 entries as testing data. The reason behind this was that a longer time was required for this method to train accurate models in comparison to the NN and RF algorithms. However, we were still able to obtain high accuracy SVM models despite the lower number of entries. Table 4.3 shows the summarized information for the SVM trained models.

Table 4.3: Summarized information of trained support vector machine models

Items	Detail
Subjects	10 healthy young adults (7 men, 3 women)
Training Data	Data from randomly selected 8 subjects: 192 inputs, 6 outputs (F_x, F_y, F_z, M_x, M_y, *Mz*) for a total of ~70500 entries
Test Data	Data from 2 remaining subjects that are not part of the training data for a total of ~16500 entries
SVM Properties	ε, Kernel Scale and Box Constrain variables are optimized and a coarse grid-search is used which increases exponentially (ε = 2^{-8} , 2^{-7} ,…, 2^3 ,2^4 and Kernel Scale and Box Constraint = 2^{-5} , 2^{-4} ,…, 2^4 ,2^5)

4.4. Results and Discussion

After training the models with NN, RF, and SVM, the test data from the same subjects were used for all methods in order to find the one that has the best accuracy and Figure 4.2

to Figure 4.7 show the results. It should be mentioned that in order to decrease noise on output data a low pass Butterworth filter with 6 Hz cut-off frequency is used.

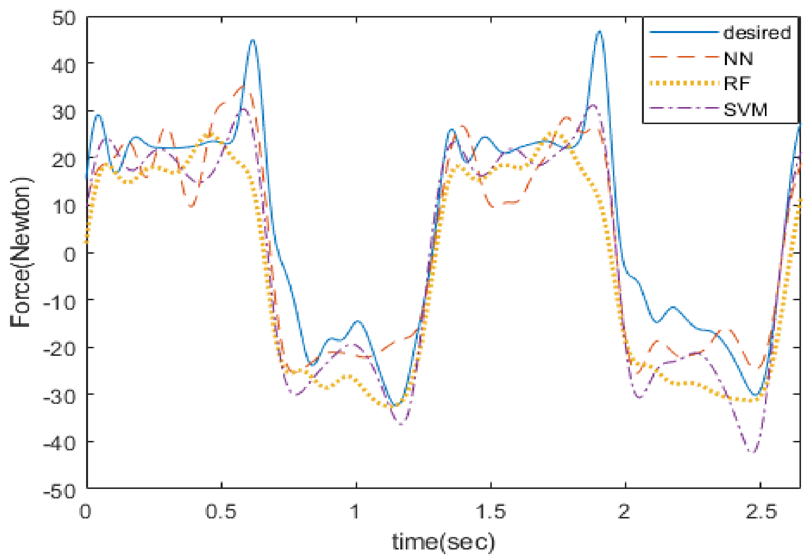


Figure 4.2. Comparison between the F_x resulted from neural network, random forest, support vector machine models and desired value

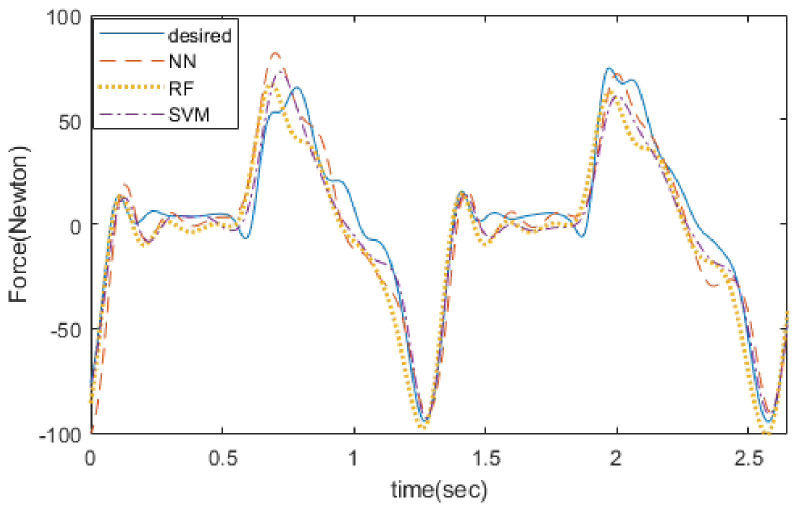


Figure 4.3. Comparison between the F_y resulted from neural network, random forest, support vector machine models and desired value

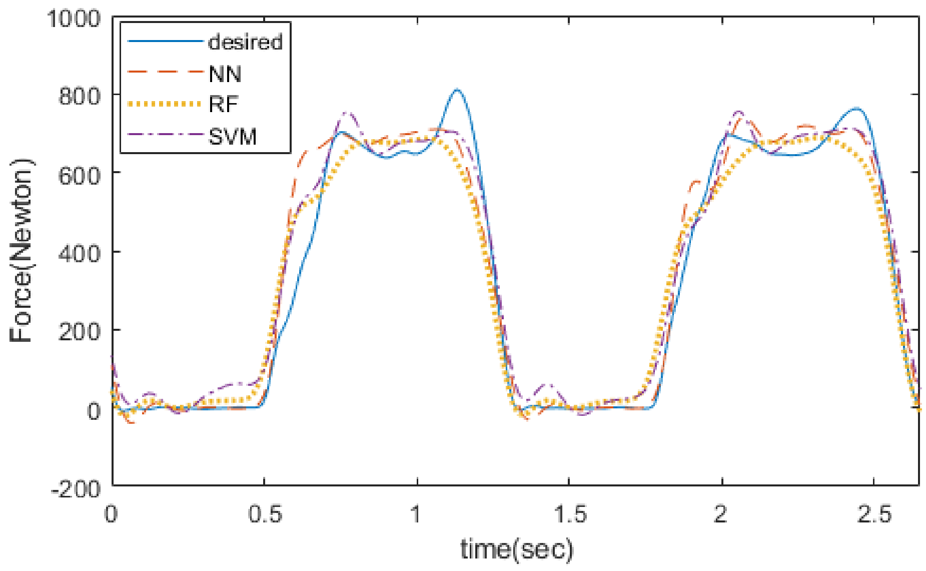


Figure 4.4. Comparison between the F_z resulted from neural network, random forest, support vector machine models and desired value

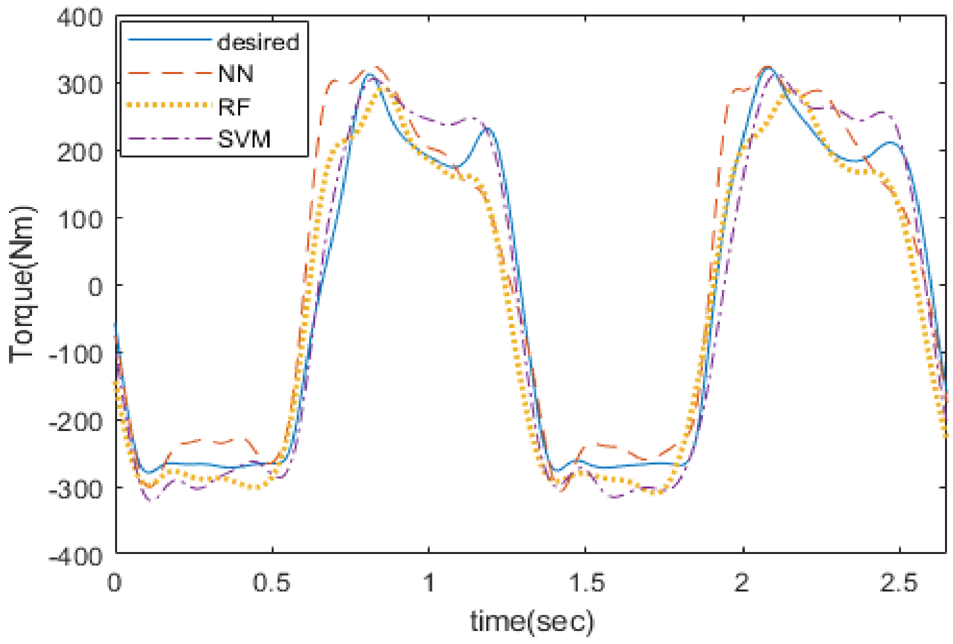


Figure 4.5. Comparison between the M_x resulted from neural network, random forest, support vector machine models and desired value

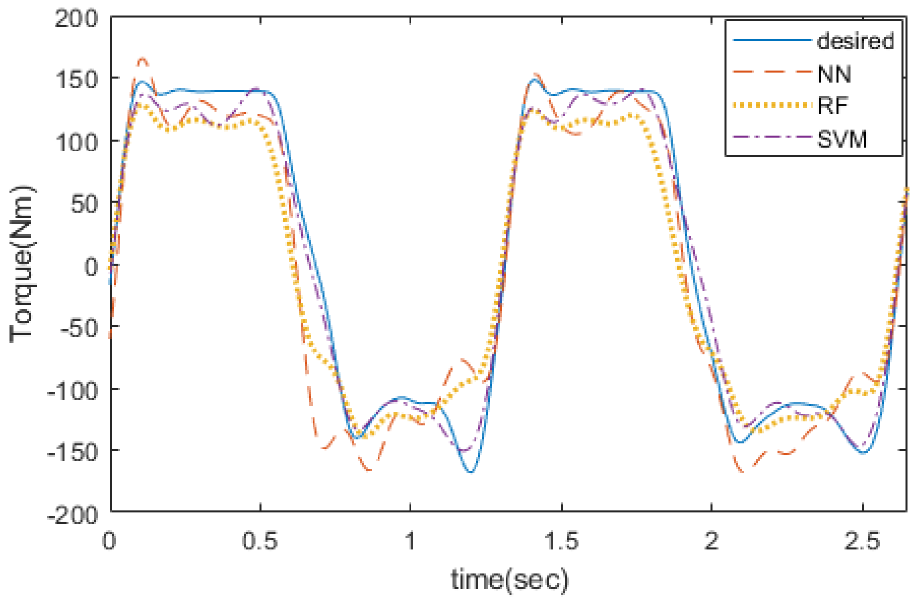


Figure 4.6. Comparison between the M_y resulted from neural network, random forest, support vector machine models and desired value

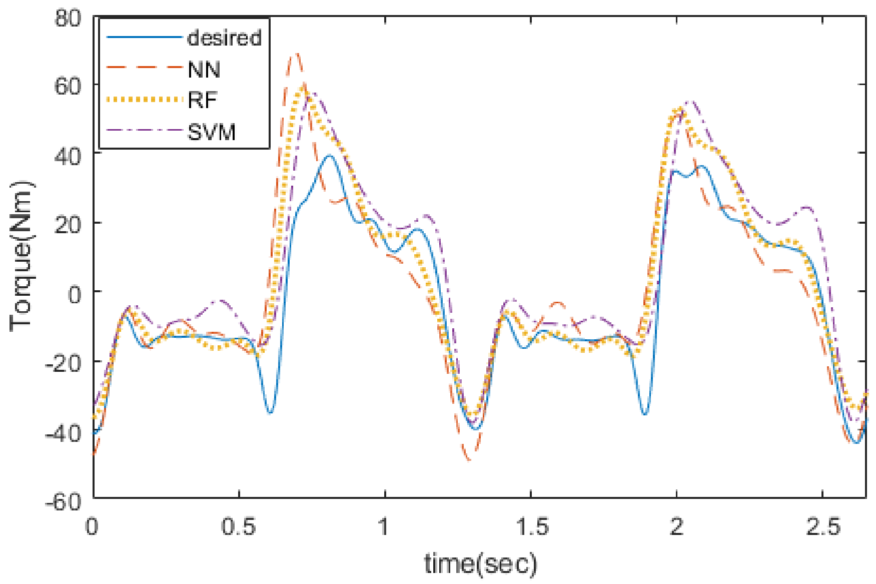


Figure 4.7. Comparison between the M_z resulted from neural network, random forest, support vector machine models and desired value

As you can see in Figure 4.2 to Figure 4.7, all models follow the trend of desired value with high accuracy. However, in some steps, the peak values differ slightly from the measured value from force plates which is normal in machine learning methods. Moreover, Table 4.4 shows the correlation between estimated and measured values with different utilized methods for each component.

Table 4.4: Correlation coefficient and NRMSE% of the GRFs and GRMs trained with neural network, random forest and support vector machine models

	Neural Network		Random Forest		Support Vector Machine	
Component	R	NRMSE%	R	NRMSE%	R	NRMSE%
F_x	0.8802	7.86	**0.8911**	7.52	0.7703	10.56
F_y	0.8879	5.28	**0.9314**	4.18	0.8989	5.03
F_z	**0.9658**	8.03	0.9641	8.22	0.9509	9.58
M_x	**0.9113**	10.22	0.8978	10.94	0.8165	14.33
M_y	0.9318	10.36	**0.9547**	8.49	0.8777	13.68
M_z	0.8247	7.12	**0.8906**	5.73	0.6985	9.01

As you can see in Table 4.4 the differences among the results from the three machine learning methods are not significant. For some components such as F_z and M_x, neural network works slightly better and for other components, random forest model shows better accuracy. The SVM results are also acceptable; however, the problem with this method is that it requires significantly more time to train. Therefore, this method which normally is highly accurate can be used only if it is possible to spend a long time modeling with it. In order to ensure that the trained models are accurate enough, the results of this study are compared with those from Joo et al. [52] and Sim et al. [46]. In [52], Joo et al. predicted 6-axis ground reaction forces and moments based on plantar pressure (PP) data obtained from insole type measurement devices. Also in [46], Sim et al. proposed a prediction model for GRFs and GRMs, which only used plantar pressure information measured from insole pressure sensors with a wavelet neural network (WNN) and principal component analysis-mutual information (PCA-MI).

Table 4.5 Comparison between the correlation coefficient and NRMSE% of the estimated GRF/M components trained using the proposed models versus models from Sim et al. *[52]* and Sim et al. *[46]* trained models

	Sim et al. research results		Joo et al. research results		Presented Method	
Component	R	NRMSE%	R	NRMSE%	R	NRMSE%
F_x	0.84	15.02	0.80	3.89	**0.8911**	7.52
F_y	0.96	12.92	0.94	2.95	**0.9314**	4.18
F_z	0.97	12.95	0.96	2.43	**0.9658**	8.03
M_x	0.85	14.87	0.92	7.25	**0.9113**	10.22
M_y	0.87	18.08	0.94	5.21	**0.9547**	8.49
M_z	0.83	17.88	0.75	5.21	**0.8906**	5.73

Table 4.5 shows that for all components of GRF/M the accuracy of trained models in this study is either almost the same or higher than most recent similar studies. By comparing the results of this study with other available studies, it can be noticed that the random forest method can have high accuracy for this application. Specially, M_z component of GRF/M usually had lower accuracy with respect to other component while modeling with machine learning algorithms. However, random forest method result shows a higher accuracy with respect to famous and common methods such as neural network.

Due to high accuracy of trained models for GRF/M while wearing a hip exoskeleton robot, it can be concluded that it may be possible to use machine learning algorithms instead of gold-standard six-axis force/torque sensor in control algorithm of wearable robots which can reduce costs of commercial wearable robots and also reduces complexity of equipped devices which helps to utilize these robots easier. In the next chapter, feasibility of this idea in a real-time control loop is tested and the results are discussed.

Chapter 5. Real-Time Control Structure Design

In this chapter, a control loop for the hip exoskeleton robot is designed. To implement the controller, motion and GRF/M of the exoskeleton user during walking is required. In order to acquire the motion, Xsens MTw IMUsensors are utilized while,as it is mentioned in section 2.1, machine learning algorithms are deployed to obtain GRF/M data.

This sensor can provide orientation, angular velocity, and angular acceleration, etc. data with sampling rates up to 120 Hz. Therefore, by setting same frequency and adjustment on the body it can be possible to utilize the information provided by these sensors for use in machine learning algorithms. If the frequency of IMU data is not the same as acquired data from motion capture cameras and GRF/M sensors, the model can not predict the GRF/M accurately. Also, if the IMUs are positioned on bad location on the leg which has undesired motions, the trained models will output un-accurate results. The sensors were attached on the thigh, shin and foot of each leg in order to acquire the knee and ankle (dorsiflexion) angles.

As it is discussed in section 2.1, different methods can be utilized to design closed-loop control of the hip exoskeleton robot. our exoskeleton robot (details explained in 1.2.1), is similar to Stride Management Assist exoskeleton [29] and can be controlled such that it helps users during walking by pushing the upper thigh back and forth. In this project, to keep implementation and testing simple, the goal is to design the control algorithm for one degree of freedom (right hip flexion-extension). This work can later on extended to include more degrees of freedom. In order to design an online control loop for the exoskeleton and read IMU data and send necessary commands to motors, a desktop computation system (Intel® Core™ i7 CPU @ 3.60GHz, 16.0 GB RAM, 64-bit Operating System) is used. This controller will generate motion of the flexion-extension DoF of the hip exoskeleton based on the user’s walking motion. Also, the IMU sensor is sending the data using Bluetooth. The used actuators for three degrees of freedom of agile eye mechanism are Maxon (DCX32L GB KL 18V) motor with planetary gearhead (GPX37 LZ 172:1). The Maxon motors are derived using Epos 2 drivers and the commands are sent to motors using serial protocol. The motors require 18 Volts which is provided by a laboratory power supply. Moreover, a miniature load cell is attached in the robot and human body interaction point to monitor exoskeleton's activity. Therefore, the applied force by the robot on the user's body can be measured and compared in different cases and situations. The load cell used in this test is demonstrated in Figure 5.1. Also, the specification of the load cell can be seen in Table 5.1.

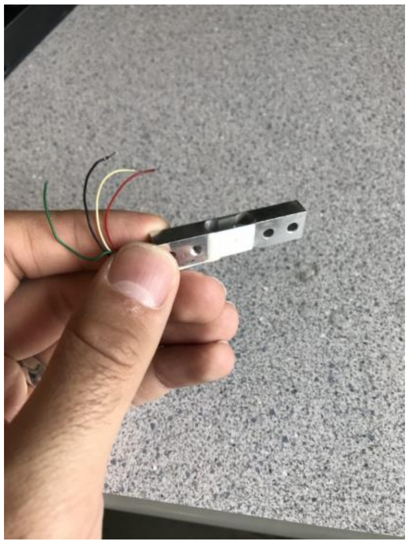


Figure 5.1. Miniature load cell

Table 5.1: Miniature load cell specifications

Load Cell Type	Strain Gauge
Capacity	5 Kg
Size	55.25mm x 12.7mm x 12.7mm
Precision	0.05%

One of the goals of this project is to study the feasibility of acquiring all components of GRF/M for controlling the hip exoskeleton robot. Therefore, a control loop is developed in MATLAB for the robot using the sensors, actuators, and desktop computer mentioned earlier. The libraries for using trained models with Keras library, sample codes for Maxon motors and Xsens IMU sensors and the load cell are available in MATLAB and. As the first step, the six trained neural network models for six components of GRF/M are loaded into the software and the whole loop was tested. The tests showed the estimation of GRF/M using trained models while acquiring data from sensors and driving Maxon motors takes a long time and makes delay for the loop that can reduce the performance of the online control algorithm. Trained models with three different algorithms were tested in the main control loop. The SVM and RF models had long delays which interrupted real-time computing and processing of motion generation. It was not possible to predict even one of the GRF/M components in real-time mode with these methods.. the tests with neural network models, however, showed that while running the online control loop, it is possible to predict up to four components of the GRF/M in the real-time mode which can be used in many studies as well as the current project.

There are a couple of options for solving this problem and use trained models in the control loop structure of the hip exoskeleton robot. First one is using much faster computers or advanced boards such as Field Programmable Gate Arrays (FPGA) for controlling the robot which can be explored in the future Another choice can be training models with fewer layers and neurons which will reduce the accuracy of the training models and therefore they weren't used. The goal of this study was to have accuracies higher than recent similar studies.

Another application for using trained models for GRF/M in control structure of the hip exoskeleton robot is using them for detecting gait phases. Many researches demonstrated that it may be possible to use the trained machine learning models in order to detect phases of walking for the hip exoskeleton robot and move the robot based on the detected phase and assist the user to walk easier. Williamson and Andrews in 2000 used a single cluster of accelerometers attached to the shank and used rule-based detectors to detect main phases of normal gait during walking [71]. Also, in 2001 Pappas et al. implemented FSR sensors and gyroscope sensors on shoes and a used rule-based detection algorithm to do phase detection on a walking person [72]. More studies in this category are reviewed by Rueterbories et al.. [73]. In 2015, Jung et al. used sensors equipped on lower limb exoskeleton robots and neural network algorithm to classify gait phase for the robot users [74]. They utilized eight FSR sensors under the sole to detect walking phases for the robot users. Basically, if the algorithm can detect walking step of the robot user, the hip exoskeleton robot can move forward and backward with respect to the user. The advantage of our method with respect to other mentioned methods is that no force sensor is used in the structure of our robot as it is hard to connect several sensors from user's sole to a hip exoskeleton robot. The best situation for using the hip exoskeleton robot is to be used without any attachments to different user's sole. Also, in most exoskeletons, IMU sensors are being used for detecting motion of users or robots and the same sensor can be used for detecting the phase of the motion. Moreover, since most cell phones have embedded IMU sensors, it may be possible to use those while the user is walking with robot. Therefore, the control algorithm for the hip exoskeleton robot can be structured as Figure 5.2.

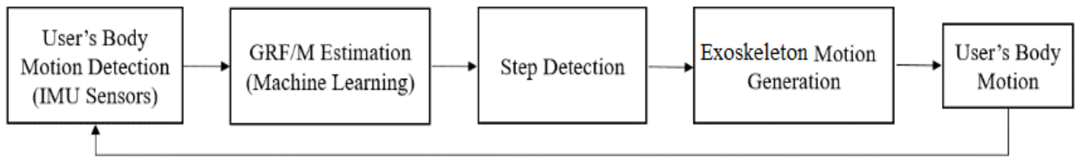

User's Body Motion Detection (IMU Sensors)
GRF/M Estimation (Machine Learning)
Step Detection
Exoskeleton Motion Generation
User's Body Motion

Figure 5.2. Designed control loop for hip exoskeleton robot

5.1. Robot Motion Generation

In this section motion generation for the hip exoskeleton robot is discussed. The robot is designed to be an assistive robot and it is needed to apply force to the upper thigh of the user to assist walking. This robot can be used by healthy adults and elderlies as well. Most elderlies have difficulty while walking especially in swing phase. Because when both feet are on the ground the skeletal structure of the body helps them to maintain balance and requires less torque from hip joints. But in swing phase, hip joints need to apply a higher amount of torque to move the leg which may be difficult for them. Therefore, this robot can help them to walk easier both in swing and stance phase. It is important for the robot to move such that the links move slightly ahead of the user's body to push the upper thigh. Otherwise, the robot would interfere with the legs' natural motion and it would cause discomfort. Using the force data provided by neural network trained models, it is possible to predict the next step of the user. During past decades many researchers studied human walking gait and their reports showed that most people have the same gait cycle when they are walking naturally with normal walking speed. Figure 5.3 demonstrates Generic healthy adult's gait cycle while walking with normal speed. As you can see, a healthy adult's walking is in the swing phase approximately 38% of each gait cycle and the rest (62%) is in the stance phase [75]. In Figure 5.3, DS and SS are double support and single support, respectively.

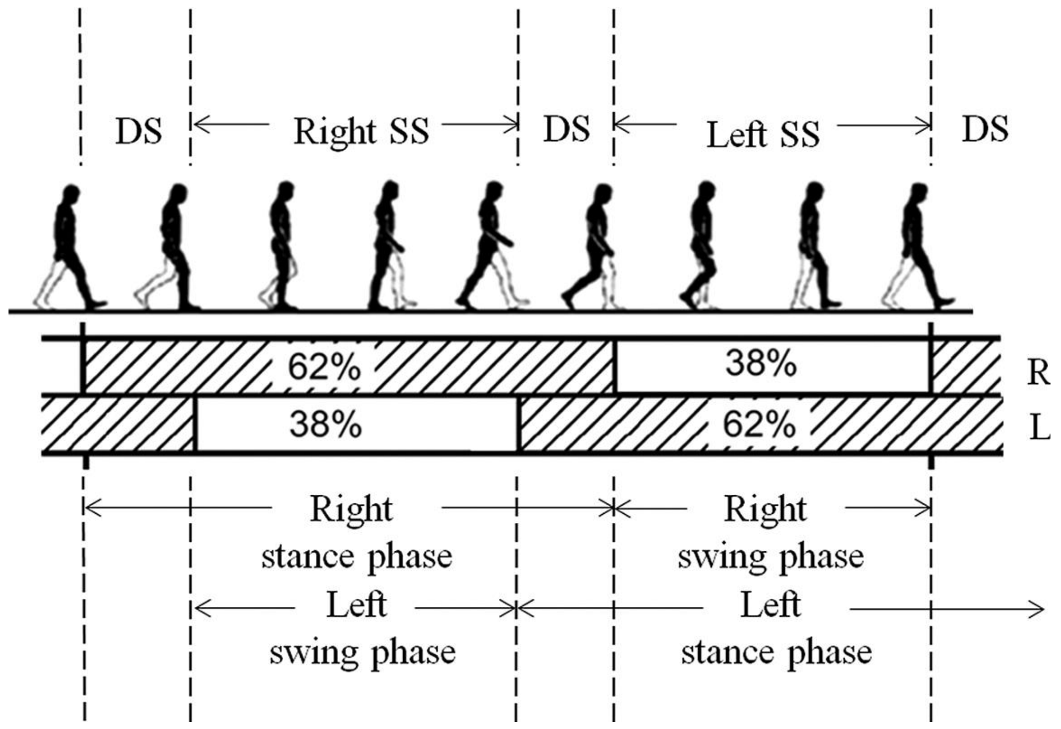


Figure 5.3. Generic healthy adult's gait cycle while walking with normal speed [75]

Therefore, using the normal walking gait of the robot user and detecting walking steps, it is possible to move the robot's joint based on the user's body. In addition, based on the limitation of the SPM's Maxon motors' angular velocity, the walking speed of the user was limited to around 0.4 m/s. Therefore, it is possible to predict the next step of the user and move the robot's link slightly ahead of it in order to push the user's body and assist him while walking. In order to assist the user's walking, the exoskeleton provides additional force to help swinging the legs in each step. In my experiment, the hip exoskeleton was programmed to move the user's thigh at about 5% prior to the start of the swing or the stance phase of the user's gait. This amount was chosen based on trial and error during our human subject testing. The important difference between this method and making an offline predefined motion for the robot is that step detection is resetting each section of walking cycle. Therefore, if the user stops moving or each phase of the cycle takes a different amount of time, it can be detected based on F_Z value. Therefore whenever the force value of the trained model for F_Z of stance foot is almost zero, it shows that the other foot is in swing phase and whenever it is not almost zero it means that the other foot is on the ground. As you can see in section 3.2, the algorithm uses 21 steps of the data which means that algorithm predicts the force value based on the change of the angle and not only based on the real-time amount of angle, angular velocity or angular acceleration which is another benefit of this method for detecting gait phases. Figure 5.4 shows a healthy adult's gait cycle and the exoskeleton's corresponding actuation points (in red). As it can be seen in the figure, when approximately 57% point of the user's walking cycle (from the start of the stance phase) has been reached, the exoskeleton's motion transfer mechanism starts to swing forward which pushes the user's thigh in the flexion direction. Next, the NN model detects the step and then after 33% from the start of the swing phase, the device swings backward which applies force on the user's thigh in the extension direction.

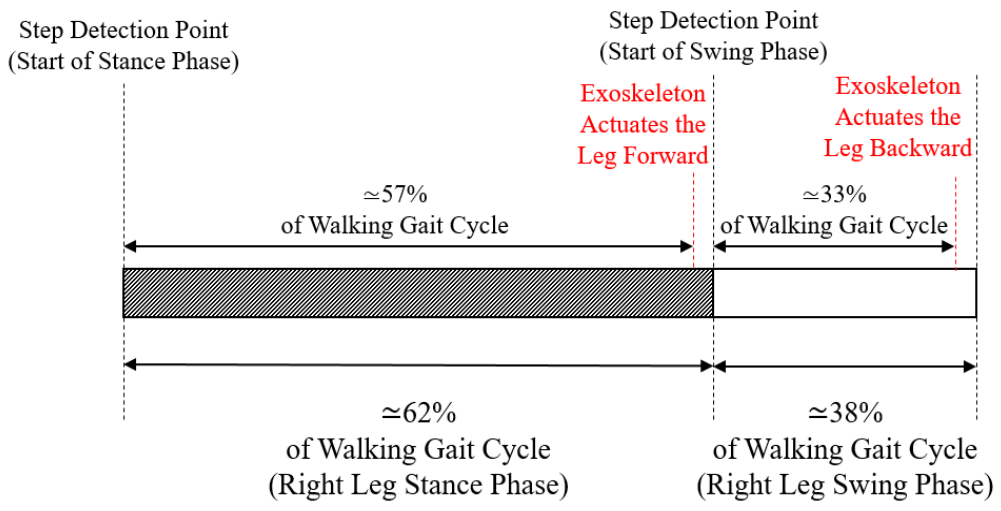

Step Detection Point
(Start of Stance Phase)
Step Detection Point
(Start of Swing Phase)
Exoskeleton Actuates the Leg Forward
Exoskeleton Actuates the Leg Backward
≃57% of Walking Gait Cycle
≃33% of Walking Gait Cycle
≃62% of Walking Gait Cycle (Right Leg Stance Phase)
≃38% of Walking Gait Cycle (Right Leg Swing Phase)

Figure 5.4. Robots gait cycle based on human walking motion

5.2. Control Loop Performance Analysis

In order to analyze performance of the proposed method, two scenarios were tested for comparison purposes. In the first scenario, the exoskeleton was powered off and the user simply walked while donning the hip exoskeleton. In the second scenario, the robot was powered on and tracked the user's thigh based on the Inverse Kinematic (IK) model developed by Sadeqi et al. in [35]. Based on their work, the robot can follow the user's body and imitate the motion. It needs to be mentioned that in this case, the exoskeleton robot is the follower of the user's motion and therefore the user needs to push the robot to be able to move. Therefore, the robot not only doesn't help the user to walk easier, but it also makes the user exhausted. The second case can show the difference between the proposed method and the case that the robot is only imitating the user's motion. Due to similarity between walking steps, the results for one of the steps are provided as follows. Figure 5.5 demonstrates applying interaction force between the user's body and robot in the attachment point recorded by the load-cell with respect to knee angle as a reference.

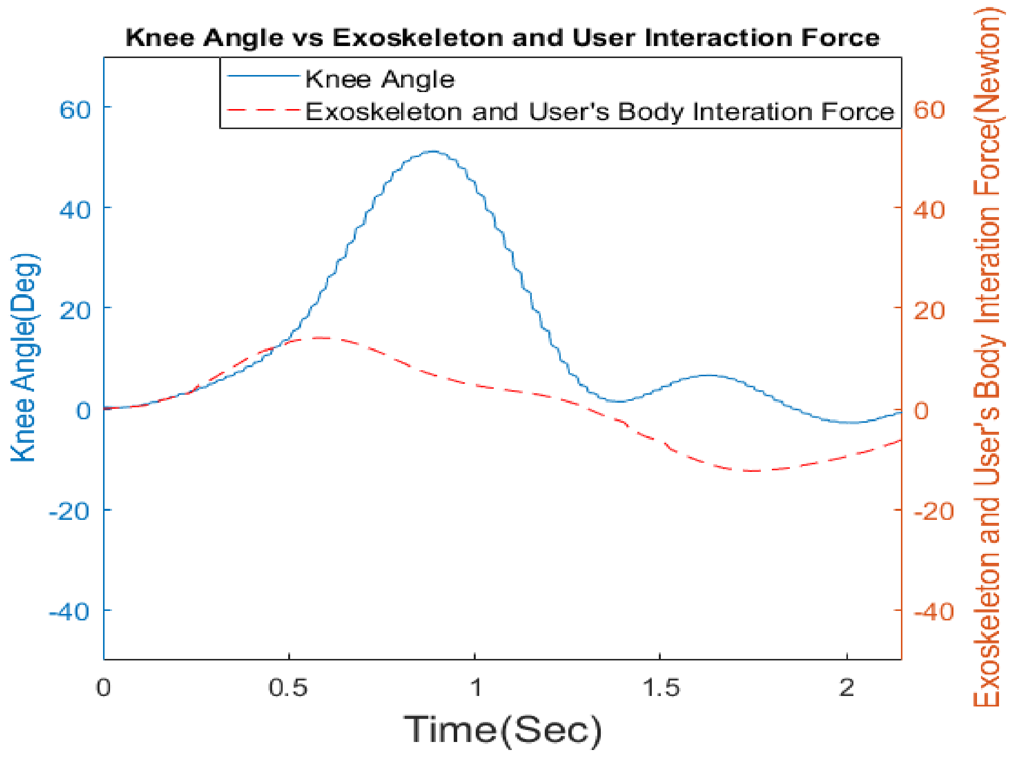


Figure 5.5. The exoskeleton and user's interaction force vs. the knee angle when the exoskeleton is powered off

As you can see in Figure 5.5, in this scenario, the user is pushing the powered-off exoskeleton and the exoskeleton-user interface force value (the dotted line) changes as user walks in the gait cycle.

In the second scenario, the robot is moving based on the user's body and robot link follows user's upper thigh based on the inverse kinematics developed in [35]. One of the randomly chosen walking step's result is provided in Figure 5.6.

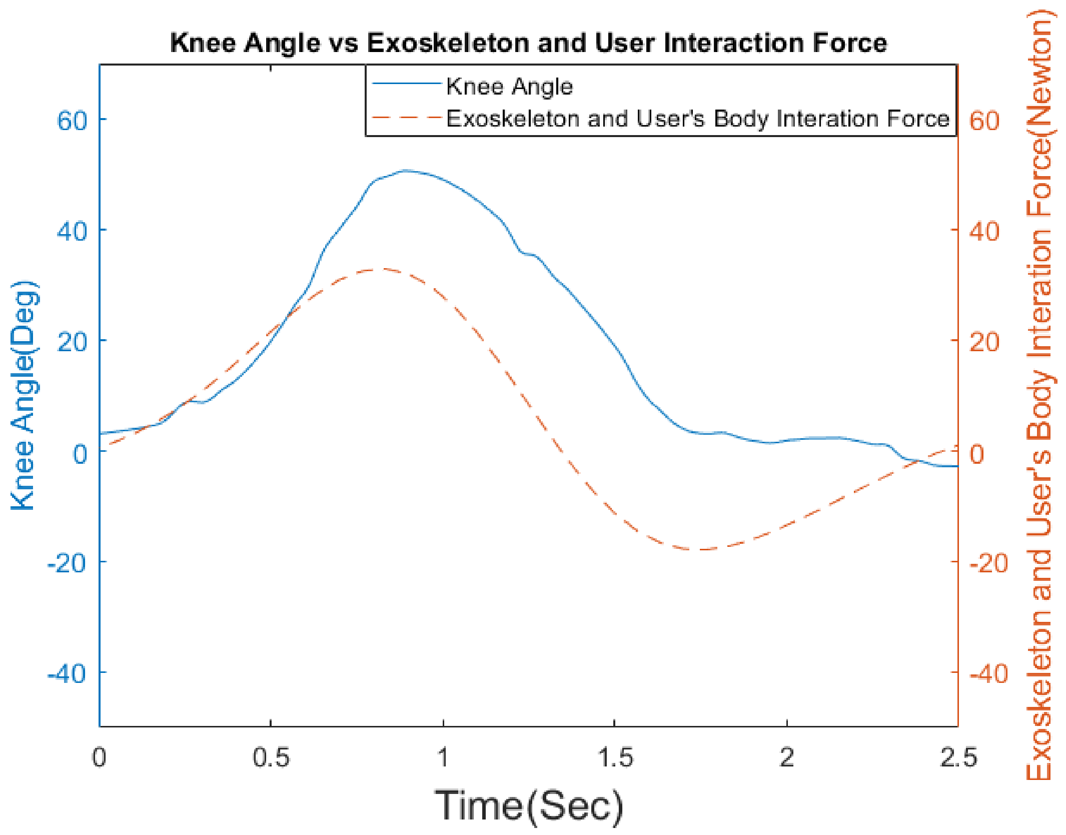


Figure 5.6. The exoskeleton and user’s interaction force vs. the knee angle when the exoskeleton is following the user’s motion

Figure 5.6 demonstrates that in this scenarios robot is moving after body motion and can not apply assistive force on the user’s body. Therefore, in the second scenario robot is avoiding user to move properly and therefore the interaction force is higher than first case. In addition, a direct comparison between the two scenarios shows that the maximum interaction force is around 15N when the exoskeleton is not actuating, while it is around 35N when the device is actuated based on the user’s IK model. This implies that the user exerts more effort during walking in order to work against the exoskeleton when the exoskeleton’s motion is generated based on the user’s IK model.

In the last scenario which is the algorithm defined in this project, robot is assisting the user and pushes the upper thigh by applying torque using agile eye mechanism. Figure 5.7 shows the applied force by the exoskeleton on the user’s thigh when the proposed assistive control with NN-based step detection is used.

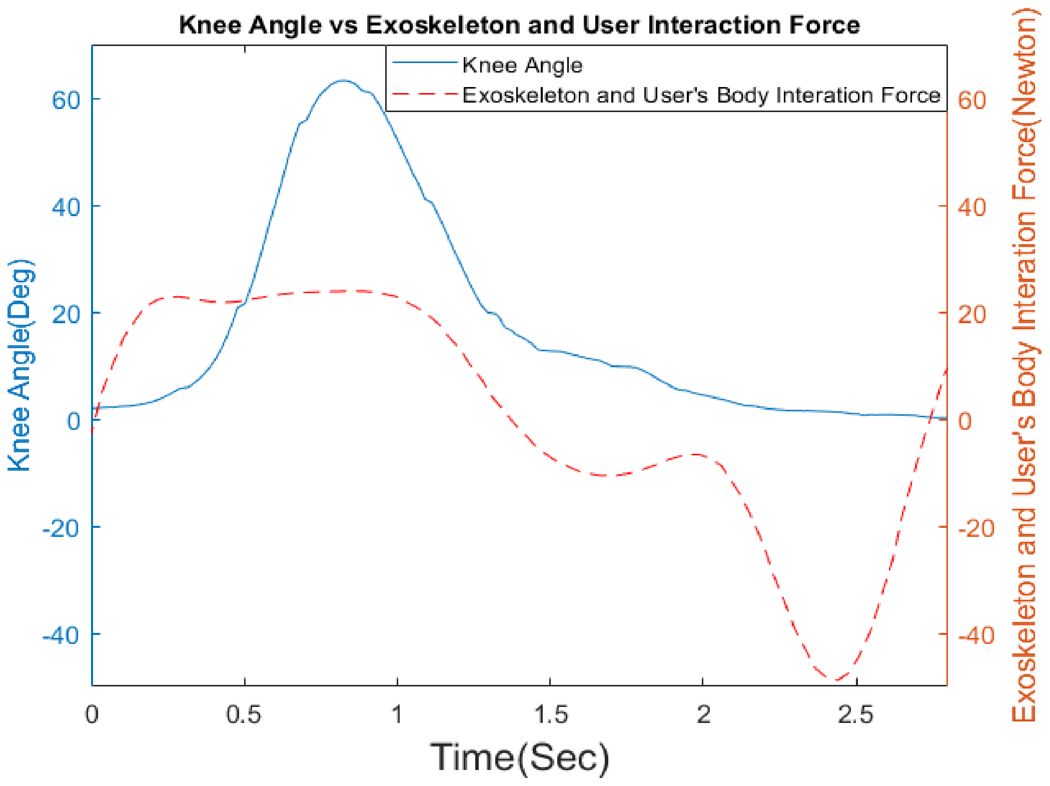


Figure 5.7. The exoskeleton-user interaction force vs. the knee angle with the proposed assistive control with NN-based step detection

By comparing Figure 5.7 against Figure 5.5 and Figure 5.5, it can be seen that the exoskeleton is actuating slightly ahead of the user and pushing on the thigh and assisting with the gait at all times. In the case that exoskeleton is powered off or working based on the IK model, the interaction forces are increasing whenever the user starts walking. However, in the proposed method, the interaction force starts slightly before the user's walking initiation (starting point of the plots in Figure 5.7) which shows that exoskeleton is exerting force on the user's upper thigh. In addition, during the gait cycle, the direction of the force exerted by the exoskeleton on the user's thigh changes when the user starts the double stance phase – this assists the user by pushing back the thigh in this phase.

Chapter 6. Conclusion

This chapter summarizes the results and findings of this project. Afterward, recommendations for future work are presented.

6.1. Conclusion

In this project, motion generation for a wearable hip exoskeleton robot is presented. The main objective of this project was to use mathematical models for ground reaction forces and moments (GRF/M) trained with machine learning algorithms instead of actual force or pressure sensors. For this purpose, neural network, random forest and support vector machine (SVM) algorithms were utilized to find the most suitable and accurate model. In order to acquire the required data for training models, 10 healthy young adults were asked to walk on a treadmill with different speed and GRF/M data was acquired using sensors implemented in the treadmill. Moreover, the motion of the lower limb was recorded using attached markers on the body and motion capture cameras. Data acquired from 8 subjects were used as training data and 2 remaining subjects as test data. The results showed that the neural network and random forest methods were able to estimate GRF/M more accurately. Also, random forest algorithm was cable of estimating M_Z with higher accuracy with respect to available studies.

In the second part of the project, the trained models were used for generating motion of a hip flexion-extension joint of the exoskeleton robot which utilizes agile eye mechanism in order to assist the user to walk easier. The models were supposed to provide GRF/M data in an online control loop. The tests showed that only neural network was capable of estimating GRF/M data while using the exoskeleton. Also, it should be mentioned that using provided PC for this study, at best 4 components of GRF/M were estimated accurately in online mode. But these data still could be used for generating motion of the robot. Also, trained models can be used as a sensor for studying the motion of the user after test is done. Based on the F_Z value from the neural network model, the walking steps of the user were detected. Whenever the force value of a leg was around zero it was counted as moving in swing phase and in other cases it was counted as stance phase. Based on the available studies, most healthy adults have similar walking phases. Generically, healthy adults are in stance phase 62% of their walking gait cycle and 38% in the swing phase. Hence, after detecting each step, the robot estimated the next step of the user's motion and moved the robot's link slightly before the time user intended to move the upper thigh. Therefore, in all steps of walking gait cycle, the exosksleton is pushing the upper thigh and assist the user to walk easier.

The presented method was compared with two other cases. In the first case, the exoskeleton is inactive and the interaction force between robot and user's body was acquired. In the second case, the exoskeleton was following user's upper thigh motion based on developed inverse kinematic models. The interaction forces showed that in the second case, exoskeleton is pushing the body back and makes walking harder. The comparison between the three scenarios demonstrated that in the presented method robot is moving slightly before the user's motion and is assisting the user to apply less torque by the hip joint.

6.2. Future Studies

In order to continue this study, it is necessary to design a lighter robot that can be used for the purpose of reducing the consumed energy of its user. Although, this study helped the exoskeleton robot user to apply less torque by the hip joint, but due to heavy structure of the exoskeleton the consumed energy is still higher than normal. Using lighter material can reduce weight of the robot. Also, the mechanism needs to be redesigned to reduce backlash in each joint. The reason is that in the existence of the backlash, the legs can move slightly in the mechanism and it is harder to design control strategy for them.

Furthermore, the utilized computing system (PC) in this study was capable of estimating four trained models at best. But using more advanced computing systems it can be possible to estimate all components of the GRF/M which can be used for development of better controllers for the robot. Moreover, training models based on data provided by more participants can increase accuracy of the trained models.

Moreover, utilized motors in the robots were only working in position control mode. Using torque controlled motors better control algorithms can be used which can improve performance of the robot.

Different machine learning algorithms can be used on the available data as well to have a better comparison between available machine learning algorithms. Another suggestion is to find models for GRF/M data based on one DoF of the leg (hip flexion-extension as an example). In that case, even IMUs implemented in cell-phones can be used as input to the models and no other sensors are needed to be attached to the exoskeleton user.

In this study, the motion generation for the exoskeleton robot is only for the hip flexion-extension joint. In the next step, motion generation can be done for the other hip joints (hip abduction-adduction and hip rotation).

Bibliography

[1] W. Erickson, C. Lee, S. Schrader, Disability Statistics from the 2013 American Community Survey (ACS), Newyork, 2015.

[2] S. Canada, Disability in Canada: Initial Findings from the Canadian Survey on Disability, Ottawa, 2012.

[3] E.Alberto, M.Talaty, A.Packel, M. Saulino, "The ReWalk Powered Exoskeleton to Restore Ambulatory Function to Individuals with Thoracic-Level Motor-Complete Spinal Cord Injury," *American Journal of Physical Medicine & Rehabilitation,* vol. 91, no. 11, pp. 911-921, 2012.

[4] M.Talaty, A.Esquenazi, J.E.Briceno, "Differentiating Ability in Users of the ReWalk Powered Exoskeleton," in *International Conference on Rehabilitation Robotics*, Seattle, 2013.

[5] K.Raab,K. Krakow,F.Tripp, M.Jung, "Effects of training with the ReWalk exoskeleton on quality of life in incomplete spinal cord injury: a single case study," *Spinal Cord Series and Cases,* vol. 2, no. 15025, 2016.

[6] S.Wang, L.Wang, C.Meijneke, E. van Asseldonk, T.Hoellinger, G.Cheron,Y.Ivanenko, V. La Scaleia, F.Sylos-Labini, M.Molinari, F.Tamburella,I.Pisotta, F.Thorsteinsson, M.Ilzkovitz, "Design and Control of the MINDWALKER Exoskeleton," *IEEE Transactions on Neural Systems and Rehabilitation Engineering,* vol. 23, no. 2, pp. 277-286, 2015.

[7] S.Wang, C.Meijneke,H.van der Kooij, "Modeling, Design, and Optimization of Mindwalker Series Elastic Joint," in *IEEE International Conference on Rehabilitation Robotics*, Seattle, 2013.

[8] G.A.Pratt,M.M.Williamson, "Series Elastic Actuators," *Human Robot Interaction and Cooperative Robots', Proceedings. 1995 IEEE/RSJ International Conference on,* vol. 1, pp. 399-406, 1995.

[9] S. Lee, Y.Sankai, "Power Assist Control for Walking Aid with HAL-3 Based on EMG and Impedance Adjustment Around Knee Joint," in *Internatiuonal Conference on Intelligent Robots and Systems*, Lausanne, Switzerland, 2002.

[10] H.Kawamoto, S. Lee, S.Kanbe,Y.Sankai, "Power Assist Method for HAL-3 using EMG-based Feedback Controller," in *IEEE International Conference on Systems, Man and Cybernetics*, Washington, 2003.

[11] H.KAawamoto,Y.Sankai, "Power Assist Method Based on Phase Sequence and Muscle Force Condition for HAL," *Advanced Robotics,* vol. 19, no. 7, p. 717–734, 2005.

[12] H.Herr, "Exoskeletons and Orthoses:Classification, Design Challenges and Future Directions," *Journal of NeuroEngineering and Rehabilitation,* vol. 6, no. 1, 2009.

[13] O.Cruciger, T. A. Schildhauer, R. C. Meindl, M.Tegenthoff, P.Schwenkreis,M.Citak, M. Aach, "Impact of Locomotion Training with a Neurologic Controlled Hybrid Assistive Limb (HAL) Exoskeleton on Neuropathic Pain and Health Related Quality of Life (HRQoL) in Chronic SCI: a Case Study," *Disability and Rehabilitation: Assistive Technology,* vol. 11, no. 6, pp. 529-534, 2016.

[14] "Rex Bionics - Reimagining Rehabilitation," [Online]. Available: https://www.rexbionics.com/.

[15] "eLEGS™ | Berkeley Robotics &amp; Human Engineering Laboratory," [Online]. Available: https://bleex.me.berkeley.edu/research/exoskeleton/elegs%E2%84%A2/. [Accessed 07 October 2010].

[16] "Austin | Berkeley Robotics &amp; Human Engineering Laboratory," [Online]. Available: https://bleex.me.berkeley.edu/research/exoskeleton/medical-exoskeleton/.

[17] "Главная страница | Drupal," [Online]. Available: https://www.exoatlet.com/en/node/84.

[18] "EksoHealth | Ekso Bionics," [Online]. Available: https://eksobionics.com/eksohealth/.

[19] A.Chu, H. Kazerooni, A.Zoss, "On the Biomimetic Design of the Berkeley Lower Extremity Exoskeleton (BLEEX)," in *International Conference on Robotics and Automation*, Barcelona, 2005.

[20] A.B. Zoss, H. Kazerooni, A.Chu, "Biomechanical Design of the Berkeley Lower Extremity Exoskeleton (BLEEX)," *IEEE/ASME Transactions on Mechatronics,* vol. 11, no. 2, pp. 128-138, 2006.

[21] A.Zoss, H. Kazerooni, "Design of an electrically actuated lower extremity exoskeleton," *Advanced Robotics,* vol. 20, no. 9, p. 967–988, 2006).

[22] H. Kazerooni, J.L.Racine, L. Huang, R. Steger, "On the Control of the Berkeley Lower Extremity Exoskeleton (BLEEX)," in *International Conference on Robotics and Automation*, Barcelona, 2005.

[23] R.Steger, S.H.Kim, and H. Kazerooni, "Control Scheme and Networked Control Architecture for the Berkeley Lower Extremity Exoskeleton (BLEEX)*," in *IEEE International Conference on Robotics and Automation*, Orlando, 2006.

[24] J.Ghan, R.Steger, H. Kazerooni, "Control and system identification for the Berkeley lower extremity exoskeleton (BLEEX)," *Advanced Robotics,* vol. 20, no. 9, p. 989–1014, 2006.

[25] H. Kazerooni, R.Steger, L.Huang, "Hybrid Control of the Berkeley Lower Extremity Exoskeleton (BLEEX)," *The International Journal of Robotics Research,* vol. 25, no. 5-6, pp. 561-573, 2006.

[26] C. J. Walsh, D. Paluska, K. Pasch, W. Grand, A. Valiente, and H. Herr, "Development of a Lightweight, Underactuated Exoskeleton for Load-Carrying Augmentation," in *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, Orlando, 2006.

[27] "“Feature: Wearable Exoskeletons That Give Humans Super Strength,” 2008.," [Online]. Available: http://www.techeblog.com/index.php/tech-gadget/feature-. [Accessed 01 September 2014].

[28] Y.Ikeuchi, J.Ashihara, Y. Hiki, H.Kudoh, T.Noda, "Walking Assist Device with Bodyweight Support System," in *International Conference on Intelligent Robots and Systems*, St. Louis,USA, 2009.

[29] "Honda Worldwide | Walking Assist," [Online]. Available: https://world.honda.com/Walking-Assist/.

[30] R.Kitatani,K.Ohata, H.Takahashi, "Reduction in Energy Expenditure During Walking Using an Automated Stride Assistance Device in Healthy Young Adults," *Archives of Physical Medicine and Rehabilitation,* vol. 95, no. 11, pp. 2128-2133, 2014.

[31] C.Buesing, G.Fisch, M.O’Donnell, I. Shahidi, L.Thomas, C.K. Mummidisetty, K.J. Williams, H.Takahashi, W.Z.Rymer, A.Jayaraman, "Effects of a wearable exoskeleton stride management assist system (SMA®) on spatiotemporal gait characteristics in individuals after stroke: a randomized controlled trial," *Journal of NeuroEngineering and Rehabilitation,* vol. 12, no. 1, p. 69, 2015.

[32] S. H.Collins,M.B.Wiggin, G.S.Sawicki, "Reducing the Energy Cost of Human Walking Using an Unpowered Exoskeleton," *Nature,* vol. 522, no. 7555, p. 212, 2015.

[33] R.Nasiri, A.Ahmadi, M.N.Ahmadabadi, "Reducing the Energy Cost of Human Running Using an Unpowered Exoskeleton," *IEEE Transactions on Neural Systems and Rehabilitation Engineering,* vol. 26, no. 10, pp. 2026-2032, 2018.

[34] S.Sadeqi, "Design of a Hybrid Spherical Manipulator for Lower Limb Exoskeleton Applications," SIMON FRASER UNIVERSITY, Vancouver, 2017.

[35] S.Sadeqi, S.P. Bourgeois, E. J Park, S.Arzanpour, "Design and performance analysis of a 3-RRR spherical parallel manipulator for hip exoskeleton applications," *Journal of Rehabilitation and Assistive Technologies Engineering,* vol. 4, pp. 1-11, 2017.

[36] C.M. Gosselin, E. St Pierre, M. Gagne, "On the Development of the Agile Eye," *IEEE Robotics & Automation Magazine,* vol. 3, no. 4, pp. 29-37, 1996.

[37] "In-Shoe Pressure Measurement | Tekscan," [Online]. Available: https://www.tekscan.com/product-group/medical/in-shoe. [Accessed 08 10 2014auth].

[38] S.B. Joo, S. E. Oh, J. H.Mun, "Improving the Ground Reaction Force Prediction Accuracy using One-Axis Plantar Pressure: Expansion of Input Variable for Neural Network," *Journal of Biomechanics,* vol. 49, no. 14, p. 3153–3161, 2016.

[39] S.E.Oh, A.Choi, J. H.Mun, "Prediction of Ground Reaction Forces and Moments During Various Activities of Daily Living," *Journal of Biomechanics,* vol. 46, no. 14, pp. 2372-2380, 2013.

[40] T.Sim, H.Kwon, S.E.Oh, S.B.Joo, A.Choi, H. M.Heo, K.Kim, J.H.Mun, "Predicting Complete Ground Reaction Forces and Moments During Gait With Insole Plantar Pressure Information Using a Wavelet Neural Network," *Journal of Biomechanical Engineering,* vol. 137, no. 9, p. 091001, 2015.

[41] E.Shahabpoor , A.Pavic, "Measurement of Walking Ground Reactions in Real-Life Environments: A Systematic Review of Techniques and Technologies," *Sensors,* vol. 17, no. 9, p. 2085, 2017.

[42] G.Logar , M.Munih, "Estimation of Joint Forces and Moments for the In-Run and Take-Off in Ski Jumping Based on Measurements with Wearable Inertial Sensors," *sensors,* vol. 15, no. 5, pp. 11258-11276, 2015.

[43] A. Karatsidis , G.Bellusci, H. M.Schepers, M.D.Zee, M. S. Andersen,P.H. Veltink, "Estimation of ground reaction forces and moments during gait using only inertial motion capture," *Sensors,* vol. 17, no. 1, p. 75, 2017.

[44] C.M. Wille , R. L. Lenhart, S.Wang, D.G. Thelen, B.C. Heiderscheit, "Ability of Sagittal Kinematic Variables to Estimate Ground Reaction Forces and Joint Kinetics in Running," *Journal of Orthopaedic & Sports Physical Therapy,* vol. 44, no. 10, p. 825–830, 2014.

[45] J.Kodama, T.Watanabe, "Examination of Inertial Sensor-Based Estimation Methods of Lower Limb Joint Moments and Ground Reaction Force: Results for Squat and Sit-to-Stand Movements in the Sagittal Plane," *Sensors,* vol. 16, no. 8, p. 1209, 2016.

[46] Taeyong Sim, Hyunbin Kwon, Seung Eel Oh, Su-Bin Joo, Ahnryul Choi, Hyun Mu Heo, Kisun Kim , Joung Hwan Mun, "Predicting Complete Ground Reaction Forces and Moments During Gait With Insole Plantar Pressure Information Using a Wavelet Neural Network," *Journal of Biomechanical Engineering,* vol. 137, no. 9, 2015.

[47] H.Rouhani, J.Favrea, X.Revoisier, K.Aminian, "Ambulatory assessment of 3D ground reaction force using plantar pressure distribution," *Gait & Posture,* vol. 32, no. 3, pp. 311-316, 2010.

[48] Y.Jung, M.Jung, J.Ryu, S. Yoon, S.K. Park, S. Koo, "Dynamically adjustable foot-ground contact model to estimate ground reaction force during walking and running," *Gait & Posture,* vol. 62, no. 8, pp. 62-68, 2015.

[49] Y.Jung , M.Jung , K. Lee , S. Koo, "Ground Reaction Force Estimation Using an Insole-Type Pressure Mat and Joint Kinematics During Walking," *Journal of Biomechanics,* vol. 47, no. 11, pp. 2693-2699, 2014.

[50] P.G. Rosquist, G.Collins, A. J.Merrell, N. J. Tuttle, J.B. Tracy, E. T. Bird, M.K. Seeley, D. T. Fullwood, W. F. Christensen, A.E. Bowden, "Estimation of 3D Ground Reaction Force Using Nanocomposite Piezo-Responsive Foam Sensors During Walking," *Annals of Biomedical Engineering,* vol. 46, no. 365, pp. 2122-2134, 2017.

[51] Y.Guo , F.Storm , Y.Zhao , S.A. Billings , A.Pavic, "A New Proxy Measurement Algorithm with Application to the Estimation of Vertical Ground Reaction Forces Using Wearable Sensors," *Sensors,* vol. 17, no. 10, p. 2181, 2017.

[52] Su-Bin Joo, SeungEelOh , JoungHwanMun, "Improving the ground reaction force prediction accuracy using one-axis plantar pressure: Expansion of input variable for neural network," *Journal of Biomechanics,* vol. 49, no. 14, pp. 3153-3161, 2016.

[53] S.E.Oh,A.Choi,J.H.Mun, "Prediction of Ground Reaction Forces and Moments During Various Activities of Daily Living," *Journal of Biomechanics,* vol. 46, no. 14, pp. 2372-2380, 2013.

[54] T.Kam.Ho, "Random Decision Forests," in *Proceedings of the 3rd International Conference on Document Analysis and Recognition*, Montreal, 1995.

[55] T.K.Ho, "The Random Subspace Method for Constructing Decision Forests," *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 20, no. 8, pp. 832-844, 1998.

[56] T.Hastie,R.Tibshirani,J.Friedman, The Elements of Statistical Learning, Springer, 2008.

[57] L.Breiman, "Random Forests," *Machine learning,* vol. 45, no. 1, pp. 5-32, 2001.

[58] A.Liaw, M.Wiener, "Classification and regression by randomforest," *R News,* vol. 2, no. 3, pp. 18-22, 2002.

[59] R.S.Chegani ,C.Menon, "Regressing Grasping Using Force Myography: an Exploratory Study," *Biomedical Engineering Online,* vol. 17, no. 1, p. 159, 2018.

[60] A.C.Bahnsen, "Machine Learning Algorithms: Introduction to Random Forests - DATAVERSITY," [Online]. Available: https://www.dataversity.net/machine-learning-algorithms-introduction-random-forests/. [Accessed 18 12 2017].

[61] V.Vapnik, The Nature of Statistical Learning Theory, 2000.

[62] M.A.Hearst, S.T.Dumais,E.Osman, J.Platt, B.Schölkopf, "Support Vector Machines," *IEEE Intelligent Systems and Their Applications,* vol. 13, no. 4, pp. 18-28, 1998.

[63] C. M.Bishop, Pattern Recognition and Machine Learning, New York: Springer, 2006.

[64] G.Cybenko, "Approximation by superpositions of a sigmoidal function," *Mathematics of Control, Signals and Systems,* vol. 2, no. 4, pp. 303-314, 1989.

[65] J.A. Snyman, D.N. Wilke, Practical Mathematical Optimization, Pretoria: Springer, 2005.

[66] H.Wu, "Global Stability Analysis of a General Class of Discontinuous Neural Networks with Linear Growth Activation Functions," *Information Sciences,* vol. 179, no. 19, pp. 3432-3441, 2009.

[67] J.Mahanta, "Introduction to Neural Networks, Advantages and Applications," [Online]. Available: https://towardsdatascience.com/introduction-to-neural-networks-advantages-and-applications-96851bd1a207. [Accessed 10 7 2017].

[68] N.Donges, "Pros and Cons of Neural Networks – Towards Data Science," [Online]. Available: https://towardsdatascience.com/hype-disadvantages-of-neural-networks-6af04904ba5b. [Accessed 17 4 2018].

[69] M.Whittle, An Introduction to Gait Analysis, Elsevier, 2007.

[70] N.Meinshausen, "Quantile Regression Forests," *Journal of Machine Learning Research,* vol. 7, no. 6, pp. 938-999, 2006.

[71] RWilliamson, BJ. Andrews, "Gait Event Detection for FES Using Accelerometers and Supervised Machine Learning," *IEEE Transactions on Rehabilitation Engineering,* vol. 8, no. 3, pp. 312-319, 2000.

[72] I.P.I. Pappas, M.R. Popovic, T.Keller, V.Dietz, M.Morari, "A Reliable Gait Phase Detection System," *IEEE Transactions on Neural Systems and Rehabilitation Engineering,* vol. 9, no. 2, pp. 113-125, 2001.

[73] J.Rueterbories, E.G. Spaich, B. Larsen, O.K. Andersen, "Methods for Gait Event Detection and Analysis in Ambulatory Systems," *Medical Engineering & Physics,* vol. 32, no. 6, pp. 545-552, 2010.

[74] J.Y. Jung, W.Heo, H.Yang, H.Park, "A Neural Network-Based Gait Phase Classification Method Using Sensors Equipped on Lower Limb Exoskeleton Robots," *Sensors,* vol. 15, no. 11, pp. 27738-27759, 2015.

[75] C.Mummolo, L.Mangialardi, J.H.Kim, "Quantifying Dynamic Characteristics of Human Walking for Comprehensive Gait Cycle," *Journal of Biomechanical Engineering,* vol. 135, no. 9, p. 091006, 2013.

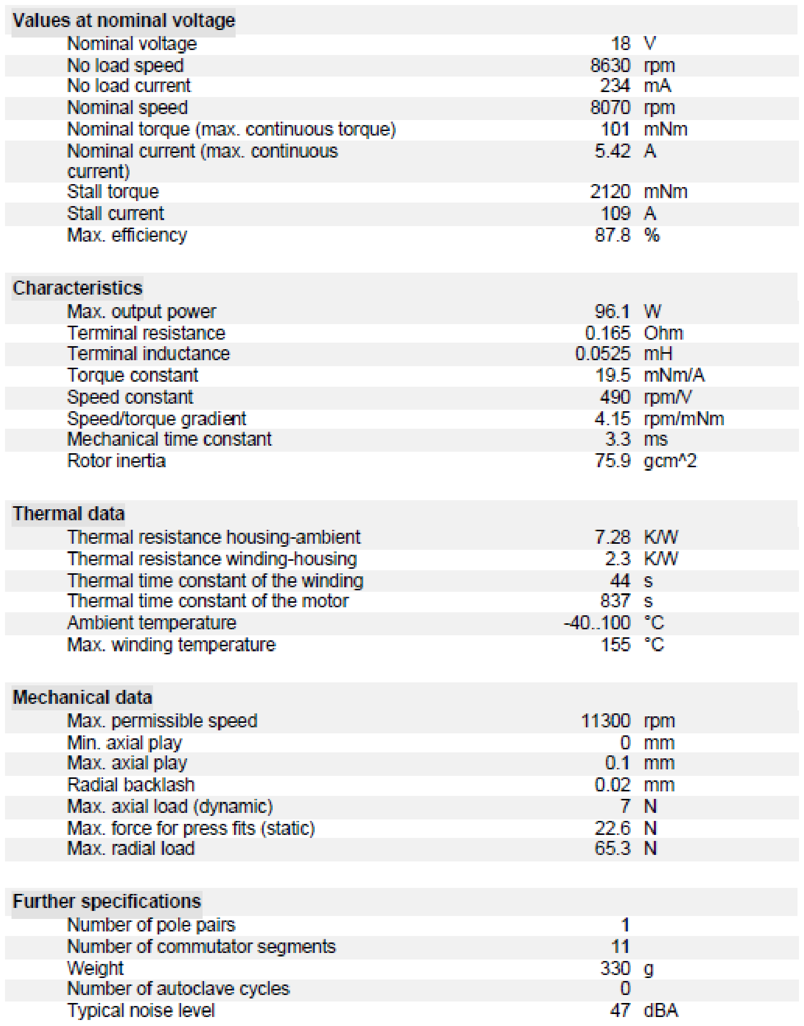
[76] K. W, "Dynamic modeling of robots using recursive Newton-Euler formulations," *Informatics in Control, Automation and Robotic,* no. Springer, pp. 3-20, 2011.

[77] T. Lung-Wen, Robot analysis: the mechanics of serial and parallel manipulators, New York: John Wiley and Sonc Inc, 1999.

[78] R. Fluit, M.S. Andersen, S. Kolk, N. Verdonschot, H.F.J.M. Koopman, "Prediction of Ground Reaction Forces and Moments During Various Activities of Daily Living," *Journal of Biomechanics,* vol. 47, no. 10, pp. 2321-2329, 2014.

Appendix A.

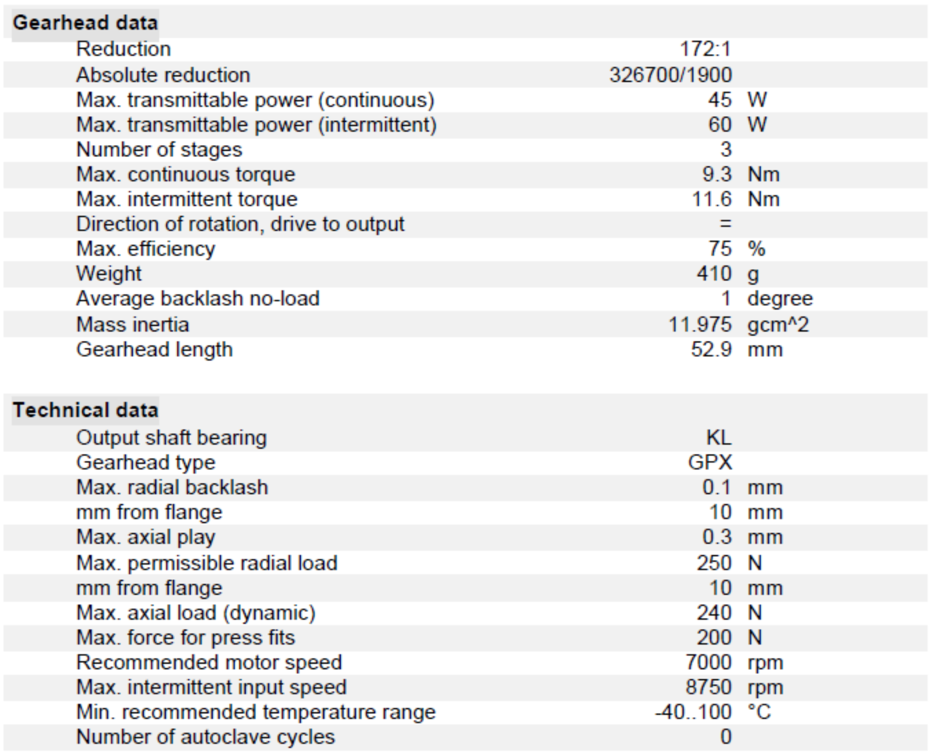
Maxon (DCX32L GB KL 18V) motor datasheet:



Values at nominal voltage		
Nominal voltage	18	V
No load speed	8630	rpm
No load current	234	mA
Nominal speed	8070	rpm
Nominal torque (max. continuous torque)	101	mNm
Nominal current (max. continuous current)	5.42	A
Stall torque	2120	mNm
Stall current	109	A
Max. efficiency	87.8	%
Characteristics		
Max. output power	96.1	W
Terminal resistance	0.165	Ohm
Terminal inductance	0.0525	mH
Torque constant	19.5	mNm/A
Speed constant	490	rpm/V
Speed/torque gradient	4.15	rpm/mNm
Mechanical time constant	3.3	ms
Rotor inertia	75.9	gcm^2
Thermal data		
Thermal resistance housing-ambient	7.28	K/W
Thermal resistance winding-housing	2.3	K/W
Thermal time constant of the winding	44	s
Thermal time constant of the motor	837	s
Ambient temperature	-40..100	°C
Max. winding temperature	155	°C
Mechanical data		
Max. permissible speed	11300	rpm
Min. axial play	0	mm
Max. axial play	0.1	mm
Radial backlash	0.02	mm
Max. axial load (dynamic)	7	N
Max. force for press fits (static)	22.6	N
Max. radial load	65.3	N
Further specifications		
Number of pole pairs	1	
Number of commutator segments	11	
Weight	330	g
Number of autoclave cycles	0	
Typical noise level	47	dBA

Figure 6.1. Datasheet of the maxon motor used in the SPM mechanism of the hip exoskeleton

Planetary gearhead (GPX37 LZ 172:1) datasheet:



Gearhead data		
Reduction	172:1	
Absolute reduction	326700/1900	
Max. transmittable power (continuous)	45	W
Max. transmittable power (intermittent)	60	W
Number of stages	3	
Max. continuous torque	9.3	Nm
Max. intermittent torque	11.6	Nm
Direction of rotation, drive to output	=	
Max. efficiency	75	%
Weight	410	g
Average backlash no-load	1	degree
Mass inertia	11.975	gcm^2
Gearhead length	52.9	mm

Technical data		
Output shaft bearing	KL	
Gearhead type	GPX	
Max. radial backlash	0.1	mm
mm from flange	10	mm
Max. axial play	0.3	mm
Max. permissible radial load	250	N
mm from flange	10	mm
Max. axial load (dynamic)	240	N
Max. force for press fits	200	N
Recommended motor speed	7000	rpm
Max. intermittent input speed	8750	rpm
Min. recommended temperature range	-40..100	°C
Number of autoclave cycles	0	

Figure 6.2. Datasheet of gearhead used for the maxon motor

Appendix B.

As a part of this project, I tried to derive a dynamic model for lower limb of a healthy adult in order to find the applying torque of each degree of freedom of lower body joints while walking. The plan was to compensate a part of the calculated torque of the hip flexion-extension joint using the hip exoskeleton robot. Therefore, lower limb of a person was assumed as a biped with 6 DoF in each leg and utilized Newton-Euler and Lagrange methods to derive dynamic models. The reason that Newton-Euler method was chosen is that this method is analytical and can calculate the joint torques with high speed. Lagrange method was used as a verification method. If the torque results for a sample motion was similar between these two methods, it means each one of these methods are verified. These models needed GRF/M as input to calculate torque of each joint and my first goal was to estimate these forces using machine learning algorithms. But as may have seen in Chapter 5, it was only possible to predict up to four components of the GRF/M which wasn't enough for estimating torques of each joint of lower body. Therefore, another strategy was applied for motion generation of the robot (Section 5.1).

In this appendix, dynamic modeling of lower limb of a healthy adult using Newton-Euler and Lagrange is discussed. Also, a comparison between these two methods will verify each of the derived models. Before that, it is necessary to find Denavit-Hartenberg parameters which are necessary for finding rotation and translation matrices for the dynamic models. The coordinate systems of biped joints are positioned and oriented as you can see in following figure based on the general instruction of Denavit-Hartenberg method. Only Z and X axes are presented for simplification.

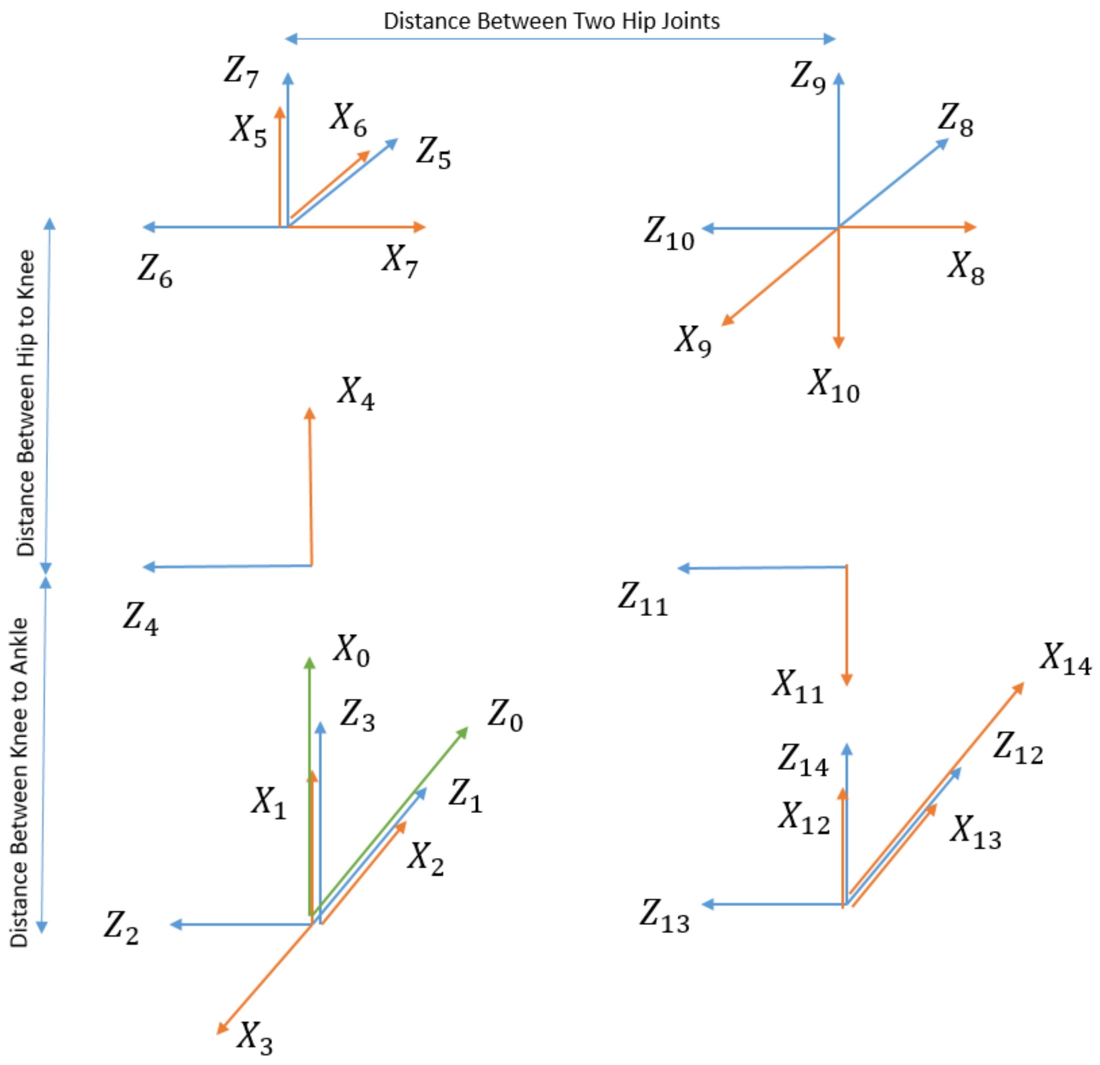


Figure 6.3. Coordinate systems of a biped based on Denavit-Hartenberg's procedure

Based on the coordinates presented in Figure 6.3 and Denavit-Hartenberg procedures, table of parameters for rotation and translation matrices can be seen in Table 6.1.

Table 6.1 Biped parameters for Denavit-Hartenberg procedure

i	a_{i-1}	α_{i-1}	d_i	θ_i
1	0	0	0	θ_1
2	0	90	0	$\theta_2 + 90$
3	0	90	0	$\theta_3 + 180$
4	L_1	90	0	$\theta_4 + 90$
5	L_2	-90	0	θ_5
6	0	90	0	$\theta_6 + 90$
7	0	90	0	$\theta_7 - 90$
8	L_3	-90	0	θ_8
9	0	90	0	$\theta_9 - 90$
10	0	90	0	$\theta_{10} - 90$
11	L_2	0	0	θ_{11}
12	L_1	90	0	$\theta_{12} + 180$
13	0	90	0	$\theta_{13} + 90$
14	0	90	0	θ_{14}

In the Table 6.1, L_1, L_2 and L_3 are distances between ankle and knee, knee and hip and distance between two hips of the user, respectively. The transfer matrices can be found based on the values mentioned in Table 6.1 and following formula.

$$ {}^{i-1}_{\;\;i}T = \begin{bmatrix} \cos\theta_i & -sin\theta_i & 0 & a_{i-1} \\ \sin\theta_i \, cos\alpha_{i-1} & \cos\theta_i \, cos\alpha_{i-1} & -sin\alpha_{i-1} & -sin\alpha_{i-1} d_i \\ \sin\theta_i \, sin\alpha_{i-1} & \cos\theta_i \, sin\alpha_{i-1} & cos\alpha_{i-1} & cos\alpha_{i-1} d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

- Newton-Euler

In order to calculate joints' torque of biped robot using newton-euler method, it is necessary to calculate position, angular velocity and angular acceleration of COM(center of mass) of each limb. These calculations can be done recursively and it should start from first joint toward the last one. Following formulas are needed to be calculated recursively in order to find all required information [76].

Outward Iteration: i=0 to 13

$$
\begin{aligned}
{}^{i+1}\omega_{i+1} &= {}^{i+1}_{i}R\, {}^{i}\omega_i + \dot{\theta}_{i+1}\, {}^{i+1}\hat{Z}_{i+1}, \\
{}^{i+1}\dot{\omega}_{i+1} &= {}^{i+1}_{i}R\, {}^{i}\dot{\omega}_i + {}^{i+1}_{i}R\, {}^{i}\omega_i \times \dot{\theta}_{i+1}\, {}^{i+1}\hat{Z}_{i+1} + \ddot{\theta}_{i+1}\, {}^{i+1}\hat{Z}_{i+1}, \\
{}^{i+1}\dot{v}_{i+1} &= {}^{i+1}_{i}R({}^{i}\dot{\omega}_i \times {}^{i}P_{i+1} + {}^{i}\omega_i \times ({}^{i}\omega_i \times {}^{i}P_{i+1}) + {}^{i}\dot{v}_i), \\
{}^{i+1}\dot{v}_{C_{i+1}} &= {}^{i+1}\dot{\omega}_{i+1} \times {}^{i+1}P_{C_{i+1}} \\
&\quad + {}^{i+1}\omega_{i+1} \times ({}^{i+1}\omega_{i+1} \times {}^{i+1}P_{C_{i+1}}) + {}^{i+1}\dot{v}_{i+1}, \\
{}^{i+1}F_{i+1} &= m_{i+1}\, {}^{i+1}\dot{v}_{C_{i+1}}, \\
{}^{i+1}N_{i+1} &= {}^{C_{i+1}}I_{i+1}\, {}^{i+1}\dot{\omega}_{i+1} + {}^{i+1}\omega_{i+1} \times {}^{C_{i+1}}I_{i+1}\, {}^{i+1}\omega_{i+1}.
\end{aligned}
$$

Inward Iteration: i=14 to 1

$$
\begin{aligned}
{}^{i}f_i &= {}^{i}_{i+1}R\, {}^{i+1}f_{i+1} + {}^{i}F_i, \\
{}^{i}n_i &= {}^{i}N_i + {}^{i}_{i+1}R\, {}^{i+1}n_{i+1} + {}^{i}P_{C_i} \times {}^{i}F_i \\
&\quad + {}^{i}P_{i+1} \times {}^{i}_{i+1}R\, {}^{i+1}f_{i+1}, \\
\tau_i &= {}^{i}n_i^T\, {}^{i}\hat{Z}_i.
\end{aligned}
$$

By having angle (θ), angular velocity and acceleration ($\dot{\theta}, \ddot{\theta}$) of each joint, approximate weight and inertial information (m,I) of each limb and distance between joints(P), it is possible to estimate the applying torque(τ) of each joint of the body while user is walking. Important missing information in the formulas are GRF/M which could be estimated from machine learning models. However, in order to find θ, $\dot{\theta}, \ddot{\theta}$ of all body joints, several IMUs need to be attached to the body and in the online closed-loop, they could not perform properly and machine learning models could not estimate GRF/M correctly.

- Lagrange

In order to verify the Newton-Euler method, the Lagrange method is developed. The equations of motion for a mechanical system with generalized coordinates q and Lagrangian L are [77]:

$$L(q,\dot{q}) = T(q,\dot{q}) - V(q)$$

$$\frac{d}{dt}\frac{\partial L}{\partial \dot{q}} - \frac{\partial L}{\partial q} = \tau$$

where F,T and V are external force acting on the whole body, kinetic energy and potential energy, respectively. In order to use these equations, center of mass of each joint is considered as a pendulum and kinetic and potential energy formulation for all biped limb is derived. In the next step, using lagrangian formulas, torque of each joint is calculated.

- Verification of Two models

In order to compare these two models, one random and similar motion for the end-effector of the biped (right leg as an example) is considered and torque of each joint calculated from two methods are compared. For this purpose, the biped limb lengths and weights are considered as can be seen in Figure 6.4.

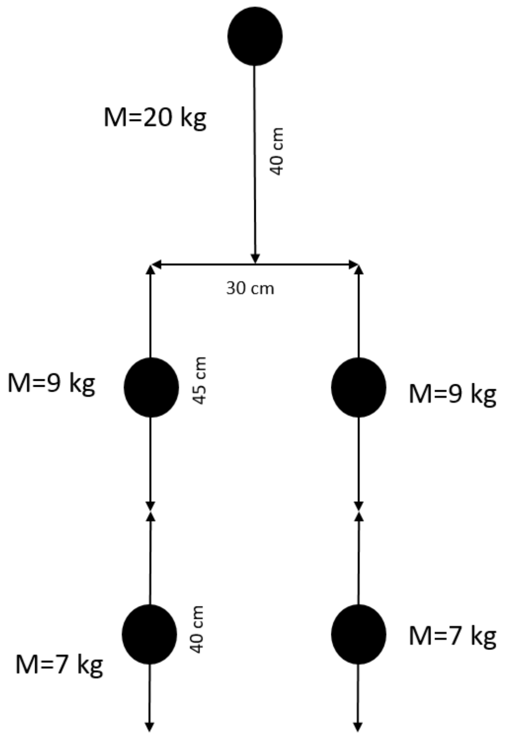


Figure 6.4. Mass distribution and length considered as an example for verification of two dynamic models

In order to test two developed dynamic models, a sample motion was generated for a random joint. Therefore, a motion from zero to 45 degrees in 2 seconds was generated for left hip abductiuon-adduction joint and the torque on the right inversion-deversion ankle joint was calculated with both methods. Figure 6.5, shows the trajectory of the moving leg in 2D.

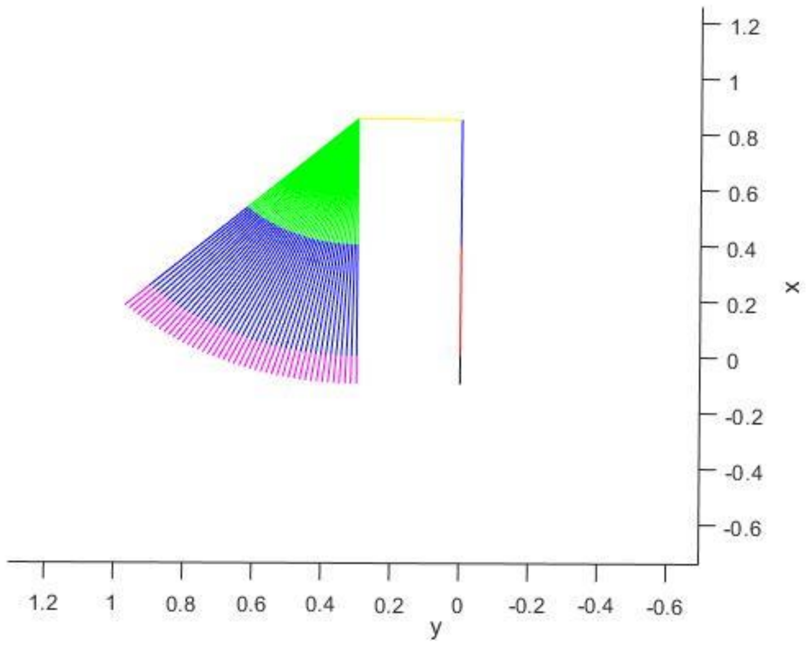


Figure 6.5. Generated motion for the hip joint as an example

After generating the motion, angle,angular velocity and angular acceleration of all joints was imported into dynamic models and the result was calculated. It should be mentioned that the calculation with newton-euler method was drastically faster as expected. Figure 6.6, shows the calculated torques with lagrange and newton-euler methods and both are highly similar. Other motions have been tested with these two methods and all had the same results for all joints. Therefore, both methods were verified and could be used to estimate the torque of all body joints.

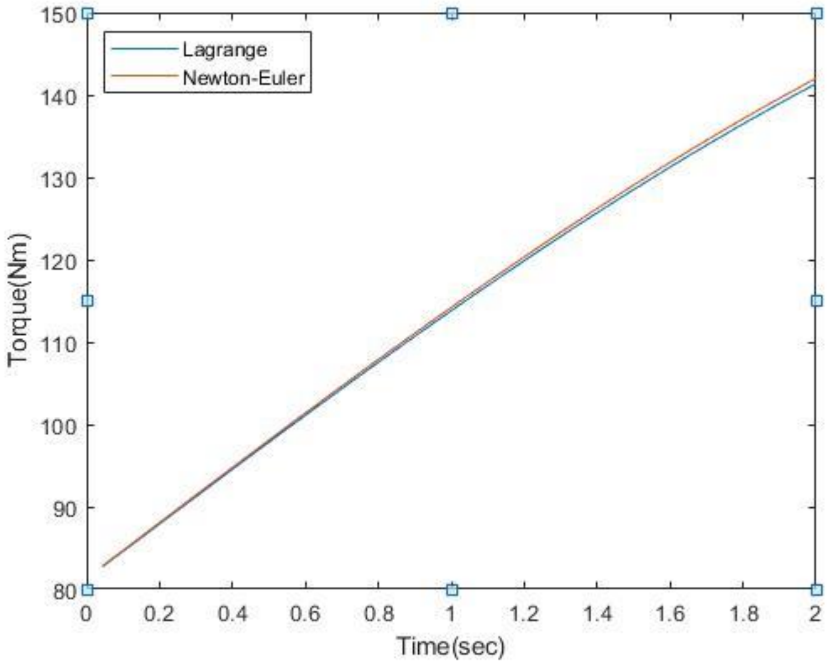


Figure 6.6. Calculated torques with two methods for the sample generated motion

Appendix C.

Python code for training GRF/M models with neural network method

```
# -*- coding: utf-8 -*-

import numpy as np
import csv

from keras.models import Sequential
from keras.layers import Dense, Activation
from keras import optimizers
from keras import backend as K

import matplotlib.pyplot as plt


x_train=[]
y_train=[]
x_test=[]
y_test=[]

# load training dataset
traindata = np.loadtxt("input_train_forRealRobot_0.4_knee_ang.csv", delimiter=",")

# split into input (X) and output (Y) variables
x_train = traindata
OutputTrain =np.loadtxt("output_train_filtered_forRealRobot_0.4_knee_ang.csv", delimiter=",")

y_train = OutputTrain[:,2]

# load testing dataset
testdata = np.loadtxt("input_test_forRealRobot_0.4_knee_ang.csv", delimiter=",")

# split into input (X) and output (Y) variables
OutputTest = np.loadtxt("output_test_filtered_forRealRobot_0.4_knee_ang.csv", delimiter=",")

x_test= testdata
y_test = OutputTest[:,2]



x_train=np.array(x_train)
x_test=np.array(x_test)
y_train=np.array(y_train)
y_test=np.array(y_test)
print(x_train[0][0])


model = Sequential()
model.add(Dense(150, activation="relu", input_dim=24, use_bias=False))
model.add(Dense(150, activation="relu", use_bias=True))
model.add(Dense(150, activation="relu", use_bias=True))
model.add(Dense(150, activation="relu", use_bias=True))
model.add(Dense(150, activation="relu", use_bias=True))
model.add(Dense(150, activation="relu", use_bias=True))
model.add(Dense(1,activation="linear"))

model.compile(optimizer='rmsprop',loss='mse')


model.summary()

batch_size = 2
num_epochs = 5

history=model.fit(x_train,y_train,epochs=num_epochs,batch_size=batch_size, validation_split=0.15)

outval=model.evaluate(x=x_test, y=y_test, batch_size=1, verbose=1)
outval

my_array=model.predict( x_test, batch_size=1, verbose=1)
my_array=my_array*cf
==============================================================
plt.plot(my_array)
plt.plot(y_test,'r')
plt.axis([14500,15500,-100,1000])
plt.ylabel('force')
plt.xlabel('num')
plt.show()

np.savetxt("predicted_output.csv", my_array, delimiter=",")

model.save('convnet_exo.h5')
model.save_weights('convnet_exo_weights_60hz_30on60.h5')
print('model saved')

tr_y_test=np.transpose(y_test)
res_data=[my_array,tr_y_test]
print(history.history.keys())
```

MATLAB code for training GRF/M models with support vector machine method:

```
input_train=csvread('H:\Thesis\Codes\input_train.csv');
output_train=csvread('H:\Thesis\Codes\output_train.csv');

input_test=csvread('H:\Thesis\Codes\input_test.csv');
output_test=csvread('H:\Thesis\Codes\output_test.csv');


%% Prepare Cross-Validation and variables for Bayesian Optimization
c = cvpartition((length(input_train)),'KFold',10);
sigma = optimizableVariable('sigma',[2.^(-
5),2.^(5)],'Type','real','Transform','none');
```

```
box = optimizableVariable('box',[2.^(-
5),2.^(5)],'Type','real','Transform','none');
eps = optimizableVariable('eps',[2.^(-
8),2.^(4)],'Type','real','Transform','none');

minfn1 =
@(z)kfoldLoss(fitrsvm(input_train,output_train(:,3),'CVPartition',c,...

'KernelFunction','rbf','Standardize',true,'BoxConstraint',z.box,'KernelScale',z
.sigma,'Epsilon',z.eps));
results1 = bayesopt(minfn1,[sigma,box,eps],'IsObjectiveDeterministic',true,...
    'AcquisitionFunctionName','expected-improvement-plus')

%% Training using optimized parameters

x(1) = results1.XAtMinObjective.sigma;
x(2) = results1.XAtMinObjective.box;
x(3) = results1.XAtMinObjective.eps;
x=[14.559   , 31.988   , 15.895];
Mdl1 = fitrsvm(input_train,output_train(:,3),'KernelFunction','rbf',...
    'Standardize',true,'KernelScale',x(1),'BoxConstraint',x(2),'Epsilon',x(3));

%% Save the model
save(sprintf('SVM_RBFRegModel_GRFz_201802'), 'Mdl1');

%% Makig predictions from the trained model
Mdl1_GRFz = predict(Mdl1,input_test);
Actual_GRFz = output_test(:,3);
plot(Mdl1_GRFz,'r')
hold on
plot(Actual_GRFz)
Tbl_op = table(Actual_GRFz,Mdl1_GRFz);
R2_RF = 1 - sum((Actual_GRFz - Mdl1_GRFz).^2)/sum((Actual_GRFz -
nanmean(Actual_GRFz)).^2)
```

MATLAB code for training GRF/M models with random forest method

```
t_RF_start = tic;

input_train=csvread('H:\Thesis\Codes\input_train_forRealRobot_0.4.csv');
output_train_filtered=csvread('H:\Thesis\Codes\output_train_filtered_forRealRobot_0.4.csv
');

input_test=csvread('H:\Thesis\Codes\input_test_forRealRobot_0.4.csv');
output_test_filtered=csvread('H:\Thesis\Codes\output_test_filtered_forRealRobot_0.4.csv')
;

mdl_RF =
TreeBagger(150,input_train,output_train_filtered(:,3),'OOBPred','On','Method','regression
','OOBVarImp','on','OOBPredictorImportance','on');
save('model_Fz_forRealRobot_0.4.mat','mdl_RF');
%%
PREDL_RF =  predict(mdl_RF, input_test);
R2_RF = 1 - sum((output_test_filtered(:,3) - PREDL_RF).^2)/sum(
(output_test_filtered(:,3) - nanmean(output_test_filtered(:,3))).^2)
RMSE_RF = sqrt(sum((output_test_filtered(:,3) -
PREDL_RF).^2)/length(output_test_filtered(:,3)))
Max_Y_RF = max(output_test_filtered(:,3));
Min_Y_RF = min(output_test_filtered(:,3));
```

```
Range_Y_RF = max(output_test_filtered(:,3))-min(output_test_filtered(:,3));
RMSEN_RF = RMSE_RF/Range_Y_RF;
t_RF_sum = toc(t_RF_start);

plot(PREDL_RF)
hold on
plot(output_test_filtered(:,3))

%%
imp = mdl_RF.OOBPermutedPredictorDeltaError;
figure;
bar(imp);
title('Curvature Test');
ylabel('Predictor importance estimates');
xlabel('Predictors');
h = gca;
h.XTickLabel = mdl_RF.PredictorNames;
h.XTickLabelRotation = 45;
h.TickLabelInterpreter = 'none';
```

MATLAB codes for Newton-Euler dynamic model:

```
        %%Right Foot is Moving and Left foot is stable on the ground

        clear all
        clc
        close(figure(1),figure(2),figure(3))

        %% Test Data
        NoD=46;
        T=2;
        dt=T/NoD;
        time=dt:dt:T;

        teta1=zeros(NoD,1);
        teta2=zeros(NoD,1);
        teta3=zeros(NoD,1);
        teta4=[0:-1:-45].'*pi/180;
        teta5=zeros(NoD,1);
        teta6=zeros(NoD,1);
        teta7=zeros(NoD,1);
        teta8=[0:-1:-45].'*pi/180;
        teta9=zeros(NoD,1);  %% Z9 and Z10 or opposit wrt to Newton Euler  %%
        teta10=[0:-1:-45].'*pi/180;
        teta11=zeros(NoD,1);
        teta12=zeros(NoD,1);
        teta13=zeros(NoD,1);
        teta14=zeros(NoD,1);

        %%
        for i=2:NoD
            qdot1(i)=[teta1(i)-teta1(i-1)]/dt;
            qdot2(i)=[teta2(i)-teta2(i-1)]/dt;
            qdot3(i)=[teta3(i)-teta3(i-1)]/dt;
            qdot4(i)=[teta4(i)-teta4(i-1)]/dt;
            qdot5(i)=[teta5(i)-teta5(i-1)]/dt;
            qdot6(i)=[teta6(i)-teta6(i-1)]/dt;
            qdot7(i)=[teta7(i)-teta7(i-1)]/dt;
            qdot8(i)=[teta8(i)-teta8(i-1)]/dt;
            qdot9(i)=[teta9(i)-teta9(i-1)]/dt;
            qdot10(i)=[teta10(i)-teta10(i-1)]/dt;
            qdot11(i)=[teta11(i)-teta11(i-1)]/dt;
            qdot12(i)=[teta12(i)-teta12(i-1)]/dt;
            qdot13(i)=[teta13(i)-teta13(i-1)]/dt;
            qdot14(i)=[teta14(i)-teta14(i-1)]/dt;

        end

        for i=2:NoD
            dqdot1(i)=[qdot1(i)-qdot1(i-1)]/dt;
            dqdot2(i)=[qdot2(i)-qdot2(i-1)]/dt;
            dqdot3(i)=[qdot3(i)-qdot3(i-1)]/dt;
            dqdot4(i)=[qdot4(i)-qdot4(i-1)]/dt;
            dqdot5(i)=[qdot5(i)-qdot5(i-1)]/dt;
            dqdot6(i)=[qdot6(i)-qdot6(i-1)]/dt;
            dqdot7(i)=[qdot7(i)-qdot7(i-1)]/dt;
            dqdot8(i)=[qdot8(i)-qdot8(i-1)]/dt;
            dqdot9(i)=[qdot9(i)-qdot9(i-1)]/dt;
            dqdot10(i)=[qdot10(i)-qdot10(i-1)]/dt;
            dqdot11(i)=[qdot11(i)-qdot11(i-1)]/dt;
            dqdot12(i)=[qdot12(i)-qdot12(i-1)]/dt;
            dqdot13(i)=[qdot13(i)-qdot13(i-1)]/dt;
            dqdot14(i)=[qdot14(i)-qdot14(i-1)]/dt;
        end

        %%

qdot1(1)=qdot1(2);qdot2(1)=qdot2(2);qdot3(1)=qdot3(2);qdot4(1)=qdot4(2);qdot5(1)=qdot5(2)
;qdot6(1)=qdot6(2);qdot7(1)=qdot7(2);qdot8(1)=qdot8(2);qdot9(1)=qdot9(2);qdot10(1)=qdot10
(2);qdot11(1)=qdot11(2);qdot12(1)=qdot12(2);qdot13(1)=qdot13(2);qdot14(1)=qdot14(2);dqdot
1(1) =dqdot1(3);dqdot2(1) =dqdot2(3);dqdot3(1) =dqdot3(3);dqdot4(1) =dqdot4(3);dqdot5(1)
=dqdot5(3);dqdot6(1) =dqdot6(3);dqdot7(1) =dqdot7(3);dqdot8(1) =dqdot8(3);dqdot9(1)
=dqdot9(3);dqdot10(1)=dqdot10(3);dqdot11(1)=dqdot11(3);dqdot12(1)=dqdot12(3);dqdot13(1)=d
qdot13(3);dqdot14(1)=dqdot14(3);   dqdot1(2) =dqdot1(3);dqdot2(2) =dqdot2(3);dqdot3(2)
=dqdot3(3);dqdot4(2) =dqdot4(3);dqdot5(2) =dqdot5(3);dqdot6(2) =dqdot6(3);dqdot7(2)
=dqdot7(3);dqdot8(2) =dqdot8(3);dqdot9(2)
=dqdot9(3);dqdot10(2)=dqdot10(3);dqdot11(2)=dqdot11(3);dqdot12(2)=dqdot12(3);dqdot13(2)=d
qdot13(3);dqdot14(2)=dqdot14(3);

        walk_data=[teta1,teta2,teta3,teta4,teta5,teta6,teta7,teta8,teta9,teta10,teta11,tet
a12,teta13,teta14];
        %%
        m1=0; %mass of lower thigh
        m1_2=0; m1_3=7;
        m2=9; %mass of upper thigh
        m3=0; % mass of upper body
        m3_2=0; m3_3=20;
        m4=0; % mass of upper thigh
        m4_2=0; m4_3=9;
        m5=7; % mass of lower thigh
        m6=0; %mass of foot
        m6_2=0; m6_3=2;

        lS2c=0.05;
        lA2c=0.2;
        lK2c=0.45/2;
        lH2cy=0.15;
        lH2cz=0.4;
        rK2cz=0.2;
        rA2cz=0.08;
        rH2cz=0.45/2;

        lA2K=0.4; %length of lower thigh
        lK2H=0.45;
        lH2rH=0.3;
        rH2K=0.45;
        rK2A=0.4;
        lS2A=0.1;
        data_save=zeros(NoD,1);
        %%
        for i=1:NoD
        w00=[0;0;0];
        dw00=[0;0;0];
        dq1=0;
        ddq1=0;
        v00=[0;0;0];
        dv00=[9.81;0;0];
        % i=1: Left Ankle Qx
        R01=[cos(teta1(i)), -sin(teta1(i)), 0;
             sin(teta1(i))*cos(0) ,cos(teta1(i))*cos(0) ,-sin(0);
             sin(teta1(i))*sin(0) ,cos(teta1(i))*sin(0) , cos(0)];
        R10=R01.';
        Z11=[0;0;1];
        P01=[0;0;0];
        Pc11=[0;0;0];
        RAdq1=qdot1(i); %???input
        RAddq1=dqdot1(i); %???input
        dq1=RAdq1;
        ddq1=RAddq1;
        lc1=0.3; %???
        Ic11=[1,0,0;
              0,1,0;
              0,0,1];
        w11(:,i)=R10*w00+dq1*Z11;
        dw11(:,i)=R10*dw00+cross(R10*w00,dq1*Z11)+ddq1*Z11;
        v11(:,i)=R10*(v00+cross(w00,P01));
        dv11(:,i)=R10*(cross(dw00,P01)+cross(w00,cross(w00,P01))+dv00);
        dvc11(:,i)=cross(dw11(:,i),Pc11)+cross(w11(:,i),cross(w11(:,i),Pc11))+dv11(:,i);
        F11(:,i)=m1*dvc11(:,i);
        N11(:,i)=Ic11*dw11(:,i)+cross(w11(:,i),Ic11*w11(:,i));

        % i=2: Left Ankle Qy
        R12=[cos(teta2(i)+pi/2)      ,    -sin(teta2(i)+pi/2)   ,     0   ;
             sin(teta2(i)+pi/2)*cos(pi/2), cos(teta2(i)+pi/2)*cos(pi/2), -sin(pi/2);
             sin(teta2(i)+pi/2)*sin(pi/2), cos(teta2(i)+pi/2)*sin(pi/2),  cos(pi/2)];
        R21=R12.';
        Z22=[0;0;1];
        P12=[0;0;0];
        Pc22=[0;0;0]
        LAdq2=qdot2(i);
        LAddq2=dqdot2(i)
        dq2=LAdq2;%???input
        ddq2=LAddq2;%???inpu
        lc1=0.3; %???
Ic22=[1,0,0;
              0,1,0;
              0,0,1];
        w22(:,i)=R21*w11(:,i)+dq2*Z22;
        dw22(:,i)=R21*dw11(:,i)+cross(R21*w11(:,i),dq2*Z22)+ddq2*Z22;
        v22(:,i)=R21*(v11(:,i)+cross(w11(:,i),P12));
        dv22(:,i)=R21*(cross(dw11(:,i),P12)+cross(w11(:,i),cross(w11(:,i),P12))+dv11(:,i))
;
        dvc22(:,i)=cross(dw22(:,i),Pc22)+cross(w22(:,i),cross(w22(:,i),Pc22))+dv22(:,i);
        F22(:,i)=m1_2*dvc22(:,i);
        N22(:,i)=Ic22*dw22(:,i)+cross(w22(:,i),Ic22*w22(:,i));

        % i=3: Left Ankle Qz

        R23=[cos(teta3(i)+pi)      ,    -sin(teta3(i)+pi)   ,     0   ;
             sin(teta3(i)+pi)*cos(pi/2), cos(teta3(i)+pi)*cos(pi/2), -sin(pi/2);
             sin(teta3(i)+pi)*sin(pi/2), cos(teta3(i)+pi)*sin(pi/2),  cos(pi/2)];
        R32=R23.';
        Z33=[0;0;1];
        P23=[0;0;0];
        Pc33=[0;0;lA2c];
        LAdq3=qdot3(i);
        LAddq3=dqdot3(i);
        dq3=LAdq3;%???input
        ddq3=LAddq3;%???input
        m1_3=7;%??? mass of lower thigh
        lc1=0.3; %???
Ic33=[1,0,0;
              0,1,0;
              0,0,1];

        w33(:,i)=R32*w22(:,i)+dq3*Z33;
        dw33(:,i)=R32*dw22(:,i)+cross(R32*w22(:,i),dq3*Z33)+ddq3*Z33;
        v33(:,i)=R32*(v22(:,i)+cross(w22(:,i),P23));
        dv33(:,i)=R32*(cross(dw22(:,i),P23)+cross(w22(:,i),cross(w22(:,i),P23))+dv22(:,i))
;
        dvc33(:,i)=cross(dw33(:,i),Pc33)+cross(w33(:,i),cross(w33(:,i),Pc33))+dv33(:,i);
        F33(:,i)=m1_3*dvc33(:,i);
        N33(:,i)=Ic33*dw33(:,i)+cross(w33(:,i),Ic33*w33(:,i));
        % i=4: Left Knee Q

        R34=[cos(teta4(i)+pi/2)       ,    -sin(teta4(i)+pi/2)    ,     0    ;
             sin(teta4(i)+pi/2)*cos(pi/2), cos(teta4(i)+pi/2)*cos(pi/2), -sin(pi/2);
             sin(teta4(i)+pi/2)*sin(pi/2), cos(teta4(i)+pi/2)*sin(pi/2),  cos(pi/2)];
        R43=R34.';
        Z44=[0;0;1];
        P34=[0;0;lA2K];
        Pc44=[lK2c;0;0]
        LKdq4=qdot4(i);
        LKddq4=dqdot4(i)
        dq4=LKdq4;%???input
        ddq4=LKddq4;%???inpu
        lc1=0.3; %???
        Ic44=[1,0,0;
               0,1,0;
               0,0,1];
        w44(:,i)=R43*w33(:,i)+dq4*Z44;
        dw44(:,i)=R43*dw33(:,i)+cross(R43*w33(:,i),dq4*Z44)+ddq4*Z44;
        v44(:,i)=R43*(v33(:,i)+cross(w33(:,i),P34));
        dv44(:,i)=R43*(cross(dw33(:,i),P34)+cross(w33(:,i),cross(w33(:,i),P34))+dv33(:,i))
;
        dvc44(:,i)=cross(dw44(:,i),Pc44)+cross(w44(:,i),cross(w44(:,i),Pc44))+dv44(:,i);
        F44(:,i)=m2*dvc44(:,i);
        N44(:,i)=Ic44*dw44(:,i)+cross(w44(:,i),Ic44*w44(:,i));
        % i=5: Left Hip Qx
        R45=[cos(teta5(i))       ,    -sin(teta5(i))    ,     0    ;
             sin(teta5(i))*cos(-pi/2), cos(teta5(i))*cos(-pi/2), -sin(-pi/2);
             sin(teta5(i))*sin(-pi/2), cos(teta5(i))*sin(-pi/2),  cos(-pi/2)];
        R54=R45.';
        Z55=[0;0;1];
        P45=[lK2H;0;0];
        Pc55=[0;0;0];
        LHdq5=qdot5(i);
        LHddq5=dqdot5(i);
        dq5=LHdq5;%???input
        ddq5=LHddq5;%???input
        lc1=0.3; %???
        Ic55=[1,0,0;
              0,1,0;
              0,0,1];
        w55(:,i)=R54*w44(:,i)+dq5*Z55;
        dw55(:,i)=R54*dw44(:,i)+cross(R54*w44(:,i),dq5*Z55)+ddq5*Z55;
        v55(:,i)=R54*(v44(:,i)+cross(w44(:,i),P45));
        dv55(:,i)=R54*(cross(dw44(:,i),P45)+cross(w44(:,i),cross(w44(:,i),P45))+dv44(:,i))
;
        dvc55(:,i)=cross(dw55(:,i),Pc55)+cross(w55(:,i),cross(w55(:,i),Pc55))+dv55(:,i);
        F55(:,i)=m3*dvc55(:,i);
        N55(:,i)=Ic55*dw55(:,i)+cross(w55(:,i),Ic55*w55(:,i));

        % i=6: Left Hip Qy
        R56=[cos(teta6(i)+pi/2)       ,    -sin(teta6(i)+pi/2)    ,     0    ;
             sin(teta6(i)+pi/2)*cos(pi/2), cos(teta6(i)+pi/2)*cos(pi/2), -sin(pi/2);
             sin(teta6(i)+pi/2)*sin(pi/2), cos(teta6(i)+pi/2)*sin(pi/2),  cos(pi/2)];

        R65=R56.';
        Z66=[0;0;1];
        P56=[0;0;0];
        Pc66=[0;0;0];
        LHdq6=qdot6(i);
        LHddq6=dqdot6(i)
        dq6=LHdq6;%???input
        ddq6=LHddq6;%???input
        lc1=0.3; %???
Ic66=[1,0,0;
              0,1,0;
              0,0,1];
        w66(:,i)=R65*w55(:,i)+dq6*Z66;
        dw66(:,i)=R65*dw55(:,i)+cross(R65*w55(:,i),dq6*Z66)+ddq6*Z66;
        v66(:,i)=R65*(v55(:,i)+cross(w55(:,i),P56));
        dv66(:,i)=R65*(cross(dw55(:,i),P56)+cross(w55(:,i),cross(w55(:,i),P56))+dv55(:,i))
;
        dvc66(:,i)=cross(dw66(:,i),Pc66)+cross(w66(:,i),cross(w66(:,i),Pc66))+dv66(:,i);
        F66(:,i)=m3_2*dvc66(:,i);
        N66(:,i)=Ic66*dw66(:,i)+cross(w66(:,i),Ic66*w66(:,i));
        % i=7: Left Hip Qz
        R67=[cos(teta7(i)-pi/2)      ,    -sin(teta7(i)-pi/2)   ,     0   ;
             sin(teta7(i)-pi/2)*cos(pi/2), cos(teta7(i)-pi/2)*cos(pi/2), -sin(pi/2);
             sin(teta7(i)-pi/2)*sin(pi/2), cos(teta7(i)-pi/2)*sin(pi/2),  cos(pi/2)];

        R76=R67.';
        Z77=[0;0;1];
        P67=[0;0;0];
        Pc77=[lH2cy;0;lH2cz];
        LHdq7=qdot7(i);
        LHddq7=dqdot7(i);
        dq7=LHdq7;%???input
        ddq7=LHddq7;%???input

        lc1=0.3; %???
        Ic77=[1,0,0;
              0,1,0;
              0,0,1];
        w77(:,i)=R76*w66(:,i)+dq7*Z77;
        dw77(:,i)=R76*dw66(:,i)+cross(R76*w66(:,i),dq7*Z77)+ddq7*Z77;
        v77(:,i)=R76*(v66(:,i)+cross(w66(:,i),P67));
        dv77(:,i)=R76*(cross(dw66(:,i),P67)+cross(w66(:,i),cross(w66(:,i),P67))+dv66(:,i))
;
        dvc77(:,i)=cross(dw77(:,i),Pc77)+cross(w77(:,i),cross(w77(:,i),Pc77))+dv77(:,i);
        F77(:,i)=m3_3*dvc77(:,i);
        N77(:,i)=Ic77*dw77(:,i)+cross(w77(:,i),Ic77*w77(:,i));

        % i=8: Right Hip Qx
        R78=[cos(teta8(i))       ,    -sin(teta8(i))   ,     0   ;
             sin(teta8(i))*cos(-pi/2), cos(teta8(i))*cos(-pi/2), -sin(-pi/2);
             sin(teta8(i))*sin(-pi/2), cos(teta8(i))*sin(-pi/2),  cos(-pi/2)];
        R87=R78.';
        Z88=[0;0;1];
        P78=[lH2rH;0;0];
        Pc88=[0;0;0]
        RHdq8=qdot8(i);
        RHddq8=dqdot8(i)
        dq8=RHdq8;%???input
        ddq8=RHddq8;%???input

        lc1=0.3; %???
        Ic88=[1,0,0;
              0,1,0;
              0,0,1];

        w88(:,i)=R87*w77(:,i)+dq8*Z88;
        dw88(:,i)=R87*dw77(:,i)+cross(R87*w77(:,i),dq8*Z88)+ddq8*Z88;
        v88(:,i)=R87*(v77(:,i)+cross(w77(:,i),P78));
        dv88(:,i)=R87*(cross(dw77(:,i),P78)+cross(w77(:,i),cross(w77(:,i),P78))+dv77(:,i))
;
        dvc88(:,i)=cross(dw88(:,i),Pc88)+cross(w88(:,i),cross(w88(:,i),Pc88))+dv88(:,i);
        F88(:,i)=m4*dvc88(:,i);
        N88(:,i)=Ic88*dw88(:,i)+cross(w88(:,i),Ic88*w88(:,i));
        % i=9: Right Hip Qy
        R89=[cos(teta9(i)-pi/2)      ,    -sin(teta9(i)-pi/2)   ,     0   ;
             sin(teta9(i)-pi/2)*cos(pi/2), cos(teta9(i)-pi/2)*cos(pi/2), -sin(pi/2);
             sin(teta9(i)-pi/2)*sin(pi/2), cos(teta9(i)-pi/2)*sin(pi/2),  cos(pi/2)];

        R98=R89.';
        Z99=[0;0;1];
        P89=[0;0;0];
        Pc99=[0;0;0];
        RHdq9=qdot9(i);
        RHddq9=dqdot9(i);
        dq9=RHdq9;%???input
        ddq9=RHddq9;%???input
        lc1=0.3; %???
        Ic99=[1,0,0;
              0,1,0;
              0,0,1];
        w99(:,i)=R98*w88(:,i)+dq9*Z99;
        dw99(:,i)=R98*dw88(:,i)+cross(R98*w88(:,i),dq9*Z99)+ddq9*Z99;
        v99(:,i)=R98*(v88(:,i)+cross(w88(:,i),P89));
        dv99(:,i)=R98*(cross(dw88(:,i),P89)+cross(w88(:,i),cross(w88(:,i),P89))+dv88(:,i))
;
        dvc99(:,i)=cross(dw99(:,i),Pc99)+cross(w99(:,i),cross(w99(:,i),Pc99))+dv99(:,i);
        F99(:,i)=m4_2*dvc99(:,i);
        N99(:,i)=Ic99*dw99(:,i)+cross(w99(:,i),Ic99*w99(:,i));
        % i=10: Right Hip Qz
        R910=[cos(teta10(i)-pi/2)      ,    -sin(teta10(i)-pi/2)   ,     0   ;
              sin(teta10(i)-pi/2)*cos(pi/2), cos(teta10(i)-pi/2)*cos(pi/2), -sin(pi/2);
              sin(teta10(i)-pi/2)*sin(pi/2), cos(teta10(i)-pi/2)*sin(pi/2),  cos(pi/2)];

        R109=R910.';
        Z1010=[0;0;1];
        P910=[0;0;0];
        Pc1010=[rH2cz;0;0];
        RHdq10=qdot10(i);
        RHddq10=dqdot10(i);

        dq10=RHdq10;%???input
        ddq10=RHddq10;%???input

        lc1=0.3; %???
        Ic1010=[1,0,0;
                0,1,0;
                0,0,1];
        w1010(:,i)=R109*w99(:,i)+dq10*Z1010;
        dw1010(:,i)=R109*dw99(:,i)+cross(R109*w99(:,i),dq10*Z1010)+ddq10*Z1010;
        v1010(:,i)=R109*(v99(:,i)+cross(w99(:,i),P910));
        dv1010(:,i)=R109*(cross(dw99(:,i),P910)+cross(w99(:,i),cross(w99(:,i),P910))+dv99(
:,i));
        dvc1010(:,i)=cross(dw1010(:,i),Pc1010)+cross(w1010(:,i),cross(w1010(:,i),Pc1010))+
dv1010(:,i);
        F1010(:,i)=m4_3*dvc1010(:,i);
        N1010(:,i)=Ic1010*dw1010(:,i)+cross(w1010(:,i),Ic1010*w1010(:,i));
        % i=11: Right Knee Qy
        R1011=[cos(teta11(i))      ,    -sin(teta11(i))   ,     0   ;
               sin(teta11(i))*cos(0), cos(teta11(i))*cos(0), -sin(0);
               sin(teta11(i))*sin(0), cos(teta11(i))*sin(0),  cos(0)];
        R1110=R1011.';
        Z1111=[0;0;1];
        P1011=[rH2K;0;0];
        Pc1111=[rK2cz;0;0];
        RHdq11=qdot11(i);
        RHddq11=dqdot11(i);

        dq11=RHdq11;%???input
        ddq11=RHddq11;%???input

        lc1=0.3; %???
        Ic1111=[1,0,0;
                0,1,0;
                0,0,1];
        w1111(:,i)=R1110*w1010(:,i)+dq11*Z1111;
        dw1111(:,i)=R1110*dw1010(:,i)+cross(R1110*w1010(:,i),dq11*Z1111)+ddq11*Z1111;
        v1111(:,i)=R1110*(v1010(:,i)+cross(w1010(:,i),P1011));
        dv1111(:,i)=R1110*(cross(dw1010(:,i),P1011)+cross(w1010(:,i),cross(w1010(:,i),P101
1))+dv1010(:,i));
        dvc1111(:,i)=cross(dw1111(:,i),Pc1111)+cross(w1111(:,i),cross(w1111(:,i),Pc1111))+
dv1111(:,i);
        F1111(:,i)=m5*dvc1111(:,i);
        N1111(:,i)=Ic1111*dw1111(:,i)+cross(w1111(:,i),Ic1111*w1111(:,i));

        % i=12: Right Ankle Qx
        R1112=[cos(teta12(i)+pi)      ,    -sin(teta12(i)+pi)   ,     0   ;
```

```
              sin(teta12(i)+pi)*cos(pi/2), cos(teta12(i)+pi)*cos(pi/2), -sin(pi/2);
              sin(teta12(i)+pi)*sin(pi/2), cos(teta12(i)+pi)*sin(pi/2),  cos(pi/2)];
       R1211=R1112.';
       Z1212=[0;0;1];
       P1112=[rK2A;0;0];
       Pc1212=[0;0;0];


       RHdq12=qdot12(i);
       RHddq12=dqdot12(i);

       dq12=RHdq12;%???input
       ddq12=RHddq12;%???input

       % m6=2; % mass of right foot
       lc1=0.3; %???
       Ic1212=[1,0,0;
               0,1,0;
               0,0,1];
       w1212(:,i)=R1211*w1111(:,i)+dq12*Z1212;
       dw1212(:,i)=R1211*dw1111(:,i)+cross(R1211*w1111(:,i),dq12*Z1212)+ddq12*Z1212;
       v1212(:,i)=R1211*(v1111(:,i)+cross(w1111(:,i),P1112));
       dv1212(:,i)=R1211*(cross(dw1111(:,i),P1112)+cross(w1111(:,i),cross(w1111(:,i),P111
2))+dv1111(:,i));
       dvc1212(:,i)=cross(dw1212(:,i),Pc1212)+cross(w1212(:,i),cross(w1212(:,i),Pc1212))+
dv1212(:,i);
       F1212(:,i)=m6*dvc1212(:,i);
       N1212(:,i)=Ic1212*dw1212(:,i)+cross(w1212(:,i),Ic1212*w1212(:,i));
       % i=13: Right Ankle Qy
       R1213=[cos(teta13(i)+pi/2)      ,    -sin(teta13(i)+pi/2)    ,     0   ;
              sin(teta13(i)+pi/2)*cos(pi/2), cos(teta13(i)+pi/2)*cos(pi/2), -sin(pi/2);
              sin(teta13(i)+pi/2)*sin(pi/2), cos(teta13(i)+pi/2)*sin(pi/2),  cos(pi/2)];

       R1312=R1213.'
       rA2cz=0.08;
       Z1313=[0;0;1];
       P1213=[0;0;0];
       Pc1313=[0;0;0]
       RHdq13=qdot13(i);
       RHddq13=dqdot13(i);
       dq13=RHdq13;%???input
       ddq13=RHddq13;%???input
       lc1=0.3; %???
       Ic1313=[1,0,0;
               0,1,0;
               0,0,1];
       w1313(:,i)=R1312*w1212(:,i)+dq13*Z1313;
       dw1313(:,i)=R1312*dw1212(:,i)+cross(R1312*w1212(:,i),dq13*Z1313)+ddq13*Z1313;
       v1313(:,i)=R1312*(v1212(:,i)+cross(w1212(:,i),P1213));
       dv1313(:,i)=R1312*(cross(dw1212(:,i),P1213)+cross(w1212(:,i),cross(w1212(:,i),P121
3))+dv1212(:,i));
       dvc1313(:,i)=cross(dw1313(:,i),Pc1313)+cross(w1313(:,i),cross(w1313(:,i),Pc1313))+
dv1313(:,i);
       F1313(:,i)=m6_2*dvc1313(:,i);
       N1313(:,i)=Ic1313*dw1313(:,i)+cross(w1313(:,i),Ic1313*w1313(:,i));

       % i=14: Right Ankle Qz
       R1314=[cos(teta14(i))      ,    -sin(teta14(i))    ,     0   ;
              sin(teta14(i))*cos(pi/2), cos(teta14(i))*cos(pi/2), -sin(pi/2);
              sin(teta14(i))*sin(pi/2), cos(teta14(i))*sin(pi/2),  cos(pi/2)];
       R1413=R1314.';
       rA2cz=0.08;
       Z1414=[0;0;1];
       P1314=[0;0;0];
       Pc1414=[0;0;-rA2cz]
       RHdq14=qdot14(i);
       RHddq14=dqdot14(i);
       dq14=RHdq14;%???input
       ddq14=RHddq14;%???input
       lc1=0.3; %???
       Ic1414=[1,0,0;
```

```
            0,1,0;
            0,0,1];
    w1414(:,i)=R1413*w1313(:,i)+dq14*Z1414;
    dw1414(:,i)=R1413*dw1313(:,i)+cross(R1413*w1313(:,i),dq14*Z1414)+ddq14*Z1414;
    v1414(:,i)=R1413*(v1313(:,i)+cross(w1313(:,i),P1314));
    dv1414(:,i)=R1413*(cross(dw1313(:,i),P1314)+cross(w1313(:,i),cross(w1313(:,i),P131
4))+dv1313(:,i));
    dvc1414(:,i)=cross(dw1414(:,i),Pc1414)+cross(w1414(:,i),cross(w1414(:,i),Pc1414))+
dv1414(:,i);
    F1414(:,i)=m6_3*dvc1414(:,i);
    N1414(:,i)=Ic1414*dw1414(:,i)+cross(w1414(:,i),Ic1414*w1414(:,i))
    %
    Tr01=[R01(1,1),R01(1,2),R01(1,3),0;
          R01(2,1),R01(2,2),R01(2,3),0;
          R01(3,1),R01(3,2),R01(3,3),0;
            0    ,    0   ,   0    ,1];
    Tr12=[R12(1,1),R12(1,2),R12(1,3),0;
          R12(2,1),R12(2,2),R12(2,3),0;
          R12(3,1),R12(3,2),R12(3,3),0;
            0    ,    0   ,   0    ,1];
    Tr23=[R23(1,1),R23(1,2),R23(1,3),0;
          R23(2,1),R23(2,2),R23(2,3),0;
          R23(3,1),R23(3,2),R23(3,3),0;
            0    ,    0   ,   0    ,1];
    Tr34=[R34(1,1),R34(1,2),R34(1,3),0;
          R34(2,1),R34(2,2),R34(2,3),0;
          R34(3,1),R34(3,2),R34(3,3),0;
            0    ,    0   ,   0    ,1];
    Tr45=[R45(1,1),R45(1,2),R45(1,3),0;
          R45(2,1),R45(2,2),R45(2,3),0;
          R45(3,1),R45(3,2),R45(3,3),0;
            0    ,    0   ,   0    ,1];
    Tr56=[R56(1,1),R56(1,2),R56(1,3),0;
          R56(2,1),R56(2,2),R56(2,3),0;
          R56(3,1),R56(3,2),R56(3,3),0;
            0    ,    0   ,   0    ,1];
    Tr67=[R67(1,1),R67(1,2),R67(1,3),0;
          R67(2,1),R67(2,2),R67(2,3),0;
          R67(3,1),R67(3,2),R67(3,3),0;
            0    ,    0   ,   0    ,1];
    Tr78=[R78(1,1),R78(1,2),R78(1,3),0;
          R78(2,1),R78(2,2),R78(2,3),0;
          R78(3,1),R78(3,2),R78(3,3),0;
            0    ,    0   ,   0    ,1];
    Tr89=[R89(1,1),R89(1,2),R89(1,3),0;
          R89(2,1),R89(2,2),R89(2,3),0;
          R89(3,1),R89(3,2),R89(3,3),0;
            0    ,    0   ,   0    ,1];
    Tr910=[R910(1,1),R910(1,2),R910(1,3),0;
          R910(2,1),R910(2,2),R910(2,3),0;
          R910(3,1),R910(3,2),R910(3,3),0;
               0   ,    0    ,   0     ,1];
    Tr1011=[R1011(1,1),R1011(1,2),R1011(1,3),0;
            R1011(2,1),R1011(2,2),R1011(2,3),0;
            R1011(3,1),R1011(3,2),R1011(3,3),0;
               0     ,     0    ,    0     ,1];
    Tr1112=[R1112(1,1),R1112(1,2),R1112(1,3),0;
            R1112(2,1),R1112(2,2),R1112(2,3),0;
            R1112(3,1),R1112(3,2),R1112(3,3),0;
               0     ,     0    ,    0     ,1];
    Tr1213=[R1213(1,1),R1213(1,2),R1213(1,3),0;
            R1213(2,1),R1213(2,2),R1213(2,3),0;
            R1213(3,1),R1213(3,2),R1213(3,3),0;
               0     ,     0    ,    0     ,1];
    Tr1314=[R1314(1,1),R1314(1,2),R1314(1,3),0;
            R1314(2,1),R1314(2,2),R1314(2,3),0;
            R1314(3,1),R1314(3,2),R1314(3,3),0;
               0     ,     0    ,    0     ,1]
    %
    Tp34=[1,0,0,P34(1);
          0,1,0,P34(2);
```

```
            0,0,1,P34(3);
            0,0,0,  1  ];
      Tp45=[1,0,0,P45(1);
            0,1,0,P45(2);
            0,0,1,P45(3);
            0,0,0,  1  ];
      Tp78=[1,0,0,P78(1);
            0,1,0,P78(2);
            0,0,1,P78(3);
            0,0,0,  1  ];
      Tp1011=[1,0,0,P1011(1);
              0,1,0,P1011(2);
              0,0,1,P1011(3);
              0,0,0,   1   ];
      Tp1112=[1,0,0,P1112(1);
              0,1,0,P1112(2);
              0,0,1,P1112(3);
              0,0,0,   1   ];
      LAnklePos=[0;0;0;1];
      LKneePos=Tr01*Tr12*Tr23*[P34;1];
      LHipPos=Tr01*Tr12*Tr23*Tp34*Tr34*[P45;1];
      RHipPos=Tr01*Tr12*Tr23*Tp34*Tr34*Tp45*Tr45*Tr56*Tr67*[P78;1];
      RKneePos=Tr01*Tr12*Tr23*Tp34*Tr34*Tp45*Tr45*Tr56*Tr67*Tp78*Tr78*Tr89*Tr910*[P1011;
1];
      RAnklePos=Tr01*Tr12*Tr23*Tp34*Tr34*Tp45*Tr45*Tr56*Tr67*Tp78*Tr78*Tr89*Tr910*Tp1011
*Tr1011*[P1112;1];

      RSolePos=Tr01*Tr12*Tr23*Tp34*Tr34*Tp45*Tr45*Tr56*Tr67*Tp78*Tr78*Tr89*Tr910*Tp1011*
Tr1011*Tp1112*Tr1112*Tr1213*Tr1314*[0;0;-0.1;1];
      f1=figure(3);
      movegui(f1,'north');
      plot3([-0.1 0],[0 0],[0 0],'k');hold on;
      plot3([0 LKneePos(1)],[0 LKneePos(2)],[0 LKneePos(3)],'r');hold on;
      plot3([LKneePos(1) LHipPos(1)],[LKneePos(2) LHipPos(2)],[LKneePos(3)
LHipPos(3)],'b');hold on;
      plot3([LHipPos(1) RHipPos(1)],[LHipPos(2) RHipPos(2)],[LHipPos(3)
RHipPos(3)],'y');hold on;
      plot3([RHipPos(1) RKneePos(1)],[RHipPos(2) RKneePos(2)],[RHipPos(3)
RKneePos(3)],'g');hold on;
      plot3([RKneePos(1) RAnklePos(1)],[RKneePos(2) RAnklePos(2)],[RKneePos(3)
RAnklePos(3)],'b');hold on;
      plot3([RAnklePos(1) RSolePos(1)],[RAnklePos(2) RSolePos(2)],[RAnklePos(3)
RSolePos(3)],'m');hold on;
      xlabel('x');
      ylabel('y');
      zlabel('z');
      xlim([-1 1])
      ylim([-1 1])
      zlim([-1 1])
      %% Velocities
      v04(:,i)=R01*R12*R23*R34*v44(:,i);
      v05(:,i)=R01*R12*R23*R34*R45*v55(:,i);
      v06(:,i)=R01*R12*R23*R34*R45*R56*v66(:,i);
      v07(:,i)=R01*R12*R23*R34*R45*R56*R67*v77(:,i);
      v08(:,i)=R01*R12*R23*R34*R45*R56*R67*R78*v88(:,i);
      %% Dynamics
      % i=14
      f1515=[0;0;0]; %we don't need foot sensor data here
      n1515=[0;0;0];
      R1415=[1,0,0 ;
             0,1,0 ;
             0,0,1];
      rA2S=0.1;
      P1415=[0;0;-rA2S];
      f1414=R1415*f1515+F1414(:,i);
      n1414=N1414(:,i)+R1415*n1515+cross(Pc1414,F1414(:,i))+cross(P1415,R1415*f1515);
      Tau14=transpose(n1414)*Z1414;

      % i=13
      % R1413=R1314.';
      f1313=R1314*f1414+F1313(:,i);
```

```
n1313=N1313(:,i)+R1314*n1414+cross(Pc1313,F1313(:,i))+cross(P1314,R1314*f1414);
Tau13=transpose(n1313)*Z1313;
% i=12
% R1312=R1213.';
f1212=R1213*f1313+F1212(:,i);
n1212=N1212(:,i)+R1213*n1313+cross(Pc1212,F1212(:,i))+cross(P1213,R1213*f1313);
Tau12=transpose(n1212)*Z1212;
% i=11
% R1211=R1112.';
f1111=R1112*f1212+F1111(:,i);
n1111=N1111(:,i)+R1112*n1212+cross(Pc1111,F1111(:,i))+cross(P1112,R1112*f1212);
Tau11=transpose(n1111)*Z1111
% i=10
R1110=R1011.';
f1010=R1011*f1111+F1010(:,i);
n1010=N1010(:,i)+R1011*n1111+cross(Pc1010,F1010(:,i))+cross(P1011,R1011*f1111);
Tau10=transpose(n1010)*Z1010;

% i=9
R109=R910.';
f99=R910*f1010+F99(:,i);
n99=N99(:,i)+R910*n1010+cross(Pc99,F99(:,i))+cross(P910,R910*f1010);
Tau9=transpose(n99)*Z99;

% i=8

R98=R89.'
f88=R89*f99+F88(:,i);
n88=N88(:,i)+R89*n99+cross(Pc88,F88(:,i))+cross(P89,R89*f99);
Tau8=transpose(n88)*Z88;
%
% i=7
R87=R78.';

f77=R78*f88+F77(:,i);
n77=N77(:,i)+R78*n88+cross(Pc77,F77(:,i))+cross(P78,R78*f88);
Tau7=transpose(n77)*Z77;
Tauu5=20*9.81*[0.15*cos(-teta2(i))-0.4*sin(-teta2(i))];
% i=6
R76=R67.';
f66=R67*f77+F66(:,i);
n66=N66(:,i)+R67*n77+cross(Pc66,F66(:,i))+cross(P67,R67*f77);
Tau6=transpose(n66)*Z66;

% i=5
R65=R56.';

f55=R56*f66+F55(:,i);
n55=N55(:,i)+R56*n66+cross(Pc55,F55(:,i))+cross(P56,R56*f66);
Tau5=transpose(n55)*Z55;

% i=
R54=R45.';

f44=R45*f55+F44(:,i);
n44=N44(:,i)+R45*n55+cross(Pc44,F44(:,i))+cross(P45,R45*f55)
Tau4=transpose(n44)*Z44;

% i=3
R43=R34.';

f33=R34*f44+F33(:,i);
n33=N33(:,i)+R34*n44+cross(Pc33,F33(:,i))+cross(P34,R34*f44);
Tau3=transpose(n33)*Z33;
% i=2
R32=R23.';
f22=R23*f33+F22(:,i);
n22=N22(:,i)+R23*n33+cross(Pc22,F22(:,i))+cross(P23,R23*f33);
Tau2=transpose(n22)*Z22;
% i=1
R21=R12.';
```

```
	f11=R12*f22+F11(:,i);
	n11=N11(:,i)+R12*n22+cross(Pc11,F11(:,i))+cross(P12,R12*f22);
	Tau1=transpose(n11)*Z11;
	%% Plot
	f2=figure(1);
	plot(time(i),Tau1,'*');
	hold on
	xlabel('Torque-newton');
	movegui(f2,'northwest');
	data_save(i)=Tau1;
	end
	fid=fopen('NE.txt','wt');
	fprintf(fid,'%f\n',data_save);
	fclose(fid);
```

MATLAB code for Lagrange dynamic model:

```
	syms teta1 teta2 teta3 teta4 teta5 teta6 teta7 teta8 teta9 teta10 teta11 
teta12 teta13 teta14 dq1 dq2 dq3 dq4 dq5 dq6 dq7 dq8 dq9 dq10 dq11 dq12 dq13 
dq14 ddq1 ddq2 ddq3 ddq4 ddq5 ddq6 ddq7 ddq8 ddq9 ddq10 ddq11 ddq12 ddq13 ddq14
	q=[teta1,teta2,teta3,teta4,teta5,teta6,teta7,teta8,teta9,teta10,teta11,te
ta12,teta13,teta14];
	dq=[dq1,dq2,dq3,dq4,dq5,dq6,dq7,dq8,dq9,dq10,dq11,dq12,dq13,dq14];
	ddq=[ddq1,ddq2,ddq3,ddq4,ddq5,ddq6,ddq7,ddq8,ddq9,ddq10,ddq11,ddq12,ddq13
,ddq14];

	g=9.81;
	%%
	Tr01=[1,0,0,0;
	      0,cos(teta1),-sin(teta1),0;
	      0,sin(teta1), cos(teta1),0;
	      0,    0     ,    0      ,1];

	Tr12=[cos(teta2),0,+sin(teta2),0;
	      0,1,0,0;
	      -sin(teta2),0,cos(teta2),0;
	      0,0,0,1];

	Tr23=[cos(teta3),-sin(teta3),0,0;
	      sin(teta3),cos(teta3),0,0;
	      0,0,1,0;
	      0,0,0,1];

	Tr34=[cos(teta4),0,+sin(teta4),0;
	      0,1,0,0;
	      -sin(teta4),0,cos(teta4),0;
	      0,0,0,1];

	Tr45=[1,0,0,0;
	      0,cos(teta5),-sin(teta5),0;
	      0,sin(teta5), cos(teta5),0;
	      0,0,0,1];

	Tr56=[cos(teta6),0,+sin(teta6),0;
	      0,1,0,0;
	      -sin(teta6),0,cos(teta6),0;
	      0,0,0,1];

	Tr67=[cos(teta7),-sin(teta7),0,0;
	     sin(teta7), cos(teta7),0,0;
```

```
     0,0,1,0;
     0,0,0,1];

Tr78=[1,0,0,0;
     0,cos(teta8),-sin(teta8),0;
     0,sin(teta8), cos(teta8),0;
     0,0,0,1];

Tr89=[cos(teta9),0,+sin(teta9),0;
     0,1,0,0;
     -sin(teta9),0,cos(teta9),0;
     0,0,0,1];

Tr910=[cos(teta10),-sin(teta10),0,0;
     sin(teta10), cos(teta10),0,0;
     0,0,1,0;
     0,0,0,1];

Tr1011=[cos(teta11),0,+sin(teta11),0;
     0,1,0,0;
     -sin(teta11)  ,0,cos(teta11),0;
     0,0,0,1];

Tr1112=[1,0,0,0;
     0,cos(teta12),-sin(teta12),0;
     0,sin(teta12),cos(teta12),0;
     0,0,0,1];

Tr1213=[cos(teta13),0,+sin(teta13),0;
         0        ,1,      0,0;
       -sin(teta13),0,cos(teta13),0;
       0,0,0,1];

Tr1314=[cos(teta14),-sin(teta14),0,0;
        sin(teta14),cos(teta14) ,0,0;
            0      ,       0    ,1,0;
            0      ,       0    ,0,1];


%%
m1=7;
m2=9;
m3=20;
m4=9;
m5=7;
m6=2
lS2c=0.05;
lA2c=0.2;
lK2c=0.45/2;
lH2cy=0.15;
lH2cz=0.4;
rH2c=0.45/2;
rK2c=0.2;
rA2c=0.05;
Pc00=[0;0;lS2c];
```

```
Pc11=[0;0;lA2c];
Pc44=[0;0;lK2c];
Pc55=[0;-lH2cy;lH2cz];
Pc1111=[0;0;-rH2c];
Pc1212=[0;0;-rK2c];
Pc1414=[0;0;-rA2c];
lA2K=0.4;
lK2H=0.45;
lH2rH=0.3;
rH2K=0.45;
rK2A=0.4;
lS2A=0.0;
P10=[0;0;lS2A];
P43=[0;0;lA2K];
P54=[0;0;lK2H];
P87=[0;-lH2rH;0];
P1110=[0;0;-rH2K];
P1211=[0;0;-rK2A];

Tp10=[1,0,0,P10(1);
      0,1,0,P10(2);
      0,0,1,P10(3);
      0,0,0,  1  ];
Tp43=[1,0,0,P43(1);
      0,1,0,P43(2);
      0,0,1,P43(3);
      0,0,0,  1  ];
Tp54=[1,0,0,P54(1);
      0,1,0,P54(2);
      0,0,1,P54(3);
      0,0,0,  1  ];
Tp87=[1,0,0,P87(1);
      0,1,0,P87(2);
      0,0,1,P87(3);
      0,0,0,  1  ];
Tp1110=[1,0,0,P1110(1);
        0,1,0,P1110(2);
        0,0,1,P1110(3);
        0,0,0,   1   ];
Tp1211=[1,0,0,P1211(1);
        0,1,0,P1211(2);
        0,0,1,P1211(3);
        0,0,0,   1   ];

%% Foot Position
LAnklePos=[P10;1];

LKneePos=Tp10*Tr01*Tr12*Tr23*[P43;1];
LKnee_x=LKneePos(1);
LKnee_y=LKneePos(2);
LKnee_z=LKneePos(3);

LHipPos=Tp10*Tr01*Tr12*Tr23*Tp43*Tr34*[P54;1];
LHip_x=LHipPos(1);
LHip_y=LHipPos(2);
```

```
        LHip_z=LHipPos(3);

        RHipPos=Tp10*Tr01*Tr12*Tr23*Tp43*Tr34*Tp54*Tr45*Tr56*Tr67*[P87;1];
        RHip_x=RHipPos(1);
        RHip_y=RHipPos(2);
        RHip_z=RHipPos(3);

        RKneePos=Tp10*Tr01*Tr12*Tr23*Tp43*Tr34*Tp54*Tr45*Tr56*Tr67*Tp87*Tr78*Tr89
*Tr910*[P1110;1];
        RKnee_x=RKneePos(1);
        RKnee_y=RKneePos(2);
        RKnee_z=RKneePos(3);

        RAnklePos=Tp10*Tr01*Tr12*Tr23*Tp43*Tr34*Tp54*Tr45*Tr56*Tr67*Tp87*Tr78*Tr8
9*Tr910*Tp1110*Tr1011*[P1211;1];
        RAnkle_x=RAnklePos(1);
        RAnkle_y=RAnklePos(2);
        RAnkle_z=RAnklePos(3);

        RSolePos=Tp10*Tr01*Tr12*Tr23*Tp43*Tr34*Tp54*Tr45*Tr56*Tr67*Tp87*Tr78*Tr89
*Tr910*Tp1110*Tr1011*Tp1211*Tr1112*Tr1213*Tr1314*[0;0;-0.1;1];
        RSole_x=RSolePos(1);
        RSole_y=RSolePos(2);
        RSole_z=RSolePos(3);

        %% Foot Velocities
        %
        LKnee_xdot=[jacobian(LKnee_x,teta1),jacobian(LKnee_x,teta2),jacobian(LKne
e_x,teta3),jacobian(LKnee_x,teta4),jacobian(LKnee_x,teta5),jacobian(LKnee_x,tet
a6),jacobian(LKnee_x,teta7),jacobian(LKnee_x,teta8),jacobian(LKnee_x,teta9),jac
obian(LKnee_x,teta10),jacobian(LKnee_x,teta11),jacobian(LKnee_x,teta12),jacobia
n(LKnee_x,teta13),jacobian(LKnee_x,teta14)]*dq.';
        LKnee_ydot=[jacobian(LKnee_y,teta1),jacobian(LKnee_y,teta2),jacobian(LKne
e_y,teta3),jacobian(LKnee_y,teta4),jacobian(LKnee_y,teta5),jacobian(LKnee_y,tet
a6),jacobian(LKnee_y,teta7),jacobian(LKnee_y,teta8),jacobian(LKnee_y,teta9),jac
obian(LKnee_y,teta10),jacobian(LKnee_y,teta11),jacobian(LKnee_y,teta12),jacobia
n(LKnee_y,teta13),jacobian(LKnee_y,teta14)]*dq.';
        LKnee_zdot=[jacobian(LKnee_z,teta1),jacobian(LKnee_z,teta2),jacobian(LKne
e_z,teta3),jacobian(LKnee_z,teta4),jacobian(LKnee_z,teta5),jacobian(LKnee_z,tet
a6),jacobian(LKnee_z,teta7),jacobian(LKnee_z,teta8),jacobian(LKnee_z,teta9),jac
obian(LKnee_z,teta10),jacobian(LKnee_z,teta11),jacobian(LKnee_z,teta12),jacobia
n(LKnee_z,teta13),jacobian(LKnee_z,teta14)]*dq.';
        VLKnee=sqrt(LKnee_xdot^2+LKnee_ydot^2+LKnee_zdot^2);

        RAnkle_xdot=[jacobian(RAnkle_x,teta1),jacobian(RAnkle_x,teta2),jacobian(R
Ankle_x,teta3),jacobian(RAnkle_x,teta4),jacobian(RAnkle_x,teta5),jacobian(RAnkl
e_x,teta6),jacobian(RAnkle_x,teta7),jacobian(RAnkle_x,teta8),jacobian(RAnkle_x,
teta9),jacobian(RAnkle_x,teta10),jacobian(RAnkle_x,teta11),jacobian(RAnkle_x,te
ta12),jacobian(RAnkle_x,teta13),jacobian(RAnkle_x,teta14)]*dq.';
        RAnkle_ydot=[jacobian(RAnkle_y,teta1),jacobian(RAnkle_y,teta2),jacobian(R
Ankle_y,teta3),jacobian(RAnkle_y,teta4),jacobian(RAnkle_y,teta5),jacobian(RAnkl
e_y,teta6),jacobian(RAnkle_y,teta7),jacobian(RAnkle_y,teta8),jacobian(RAnkle_y,
teta9),jacobian(RAnkle_y,teta10),jacobian(RAnkle_y,teta11),jacobian(RAnkle_y,te
ta12),jacobian(RAnkle_y,teta13),jacobian(RAnkle_y,teta14)]*dq.';
```

```
       RAnkle_zdot=[jacobian(RAnkle_z,teta1),jacobian(RAnkle_z,teta2),jacobian(R
Ankle_z,teta3),jacobian(RAnkle_z,teta4),jacobian(RAnkle_z,teta5),jacobian(RAnkl
e_z,teta6),jacobian(RAnkle_z,teta7),jacobian(RAnkle_z,teta8),jacobian(RAnkle_z,
teta9),jacobian(RAnkle_z,teta10),jacobian(RAnkle_z,teta11),jacobian(RAnkle_z,te
ta12),jacobian(RAnkle_z,teta13),jacobian(RAnkle_z,teta14)]*dq.';
       VRAnkle=sqrt(RAnkle_xdot^2+RAnkle_ydot^2+RAnkle_zdot^2);

       RSole_xdot=[jacobian(RSole_x,teta1),jacobian(RSole_x,teta2),jacobian(RSol
e_x,teta3),jacobian(RSole_x,teta4),jacobian(RSole_x,teta5),jacobian(RSole_x,tet
a6),jacobian(RSole_x,teta7),jacobian(RSole_x,teta8),jacobian(RSole_x,teta9),jac
obian(RSole_x,teta10),jacobian(RSole_x,teta11),jacobian(RSole_x,teta12),jacobia
n(RSole_x,teta13),jacobian(RSole_x,teta14)]*dq.';
       RSole_ydot=[jacobian(RSole_y,teta1),jacobian(RSole_y,teta2),jacobian(RSol
e_y,teta3),jacobian(RSole_y,teta4),jacobian(RSole_y,teta5),jacobian(RSole_y,tet
a6),jacobian(RSole_y,teta7),jacobian(RSole_y,teta8),jacobian(RSole_y,teta9),jac
obian(RSole_y,teta10),jacobian(RSole_y,teta11),jacobian(RSole_y,teta12),jacobia
n(RSole_y,teta13),jacobian(RSole_y,teta14)]*dq.';
       RSole_zdot=[jacobian(RSole_z,teta1),jacobian(RSole_z,teta2),jacobian(RSol
e_z,teta3),jacobian(RSole_z,teta4),jacobian(RSole_z,teta5),jacobian(RSole_z,tet
a6),jacobian(RSole_z,teta7),jacobian(RSole_z,teta8),jacobian(RSole_z,teta9),jac
obian(RSole_z,teta10),jacobian(RSole_z,teta11),jacobian(RSole_z,teta12),jacobia
n(RSole_z,teta13),jacobian(RSole_z,teta14)]*dq.';
       VRSole=sqrt(RSole_xdot^2+RSole_ydot^2+RSole_zdot^2);

       %% CoG Positions
       LSole_AnkleCoG=[Pc00;1];

       LAnkle_KneeCoG=Tp10*Tr01*Tr12*Tr23*[Pc11;1];
       LAnkle_KneeCoG_x=LAnkle_KneeCoG(1);
       LAnkle_KneeCoG_y=LAnkle_KneeCoG(2);
       LAnkle_KneeCoG_z=LAnkle_KneeCoG(3);

       LKnee_HipCoG=Tp10*Tr01*Tr12*Tr23*Tp43*Tr34*[Pc44;1];
       LKnee_HipCoG_x=LKnee_HipCoG(1);
       LKnee_HipCoG_y=LKnee_HipCoG(2);
       LKnee_HipCoG_z=LKnee_HipCoG(3);

       UHipCoG=Tp10*Tr01*Tr12*Tr23*Tp43*Tr34*Tp54*Tr45*Tr56*Tr67*[Pc55;1];
       UHipCoG_x=UHipCoG(1);
       UHipCoG_y=UHipCoG(2);
       UHipCoG_z=UHipCoG(3);

       RHip_KneeCoG=Tp10*Tr01*Tr12*Tr23*Tp43*Tr34*Tp54*Tr45*Tr56*Tr67*Tp87*Tr78*
Tr89*Tr910*[Pc1111;1];
       RHip_KneeCoG_x=RHip_KneeCoG(1);
       RHip_KneeCoG_y=RHip_KneeCoG(2);
       RHip_KneeCoG_z=RHip_KneeCoG(3);

       RKnee_AnkleCoG=Tp10*Tr01*Tr12*Tr23*Tp43*Tr34*Tp54*Tr45*Tr56*Tr67*Tp87*Tr7
8*Tr89*Tr910*Tp1110*Tr1011*[Pc1212;1];
       RKnee_AnkleCoG_x=RKnee_AnkleCoG(1);
       RKnee_AnkleCoG_y=RKnee_AnkleCoG(2);
       RKnee_AnkleCoG_z=RKnee_AnkleCoG(3);
```

```
	RSoleCoG=Tp10*Tr01*Tr12*Tr23*Tp43*Tr34*Tp54*Tr45*Tr56*Tr67*Tp87*Tr78*Tr89
*Tr910*Tp1110*Tr1011*Tp1211*Tr1112*Tr1213*Tr1314*[Pc1414;1];
	RSoleCoG_x=RSoleCoG(1);
	RSoleCoG_y=RSoleCoG(2);
	RSoleCoG_z=RSoleCoG(3);

	%% CoG Velocities
	%
	LAnkle_KneeCoG_xdot=[jacobian(LAnkle_KneeCoG_x,teta1),jacobian(LAnkle_Kne
eCoG_x,teta2),jacobian(LAnkle_KneeCoG_x,teta3),jacobian(LAnkle_KneeCoG_x,teta4)
,jacobian(LAnkle_KneeCoG_x,teta5),jacobian(LAnkle_KneeCoG_x,teta6),jacobian(LAn
kle_KneeCoG_x,teta7),jacobian(LAnkle_KneeCoG_x,teta8),jacobian(LAnkle_KneeCoG_x
,teta9),jacobian(LAnkle_KneeCoG_x,teta10),jacobian(LAnkle_KneeCoG_x,teta11),jac
obian(LAnkle_KneeCoG_x,teta12),jacobian(LAnkle_KneeCoG_x,teta13),jacobian(LAnkl
e_KneeCoG_x,teta14)]*dq.';
	LAnkle_KneeCoG_ydot=[jacobian(LAnkle_KneeCoG_y,teta1),jacobian(LAnkle_Kne
eCoG_y,teta2),jacobian(LAnkle_KneeCoG_y,teta3),jacobian(LAnkle_KneeCoG_y,teta4)
,jacobian(LAnkle_KneeCoG_y,teta5),jacobian(LAnkle_KneeCoG_y,teta6),jacobian(LAn
kle_KneeCoG_y,teta7),jacobian(LAnkle_KneeCoG_y,teta8),jacobian(LAnkle_KneeCoG_y
,teta9),jacobian(LAnkle_KneeCoG_y,teta10),jacobian(LAnkle_KneeCoG_y,teta11),jac
obian(LAnkle_KneeCoG_y,teta12),jacobian(LAnkle_KneeCoG_y,teta13),jacobian(LAnkl
e_KneeCoG_y,teta14)]*dq.';
	LAnkle_KneeCoG_zdot=[jacobian(LAnkle_KneeCoG_z,teta1),jacobian(LAnkle_Kne
eCoG_z,teta2),jacobian(LAnkle_KneeCoG_z,teta3),jacobian(LAnkle_KneeCoG_z,teta4)
,jacobian(LAnkle_KneeCoG_z,teta5),jacobian(LAnkle_KneeCoG_z,teta6),jacobian(LAn
kle_KneeCoG_z,teta7),jacobian(LAnkle_KneeCoG_z,teta8),jacobian(LAnkle_KneeCoG_z
,teta9),jacobian(LAnkle_KneeCoG_z,teta10),jacobian(LAnkle_KneeCoG_z,teta11),jac
obian(LAnkle_KneeCoG_z,teta12),jacobian(LAnkle_KneeCoG_z,teta13),jacobian(LAnkl
e_KneeCoG_z,teta14)]*dq.';
	VLAnkle_KneeCoG=sqrt(LAnkle_KneeCoG_xdot^2+LAnkle_KneeCoG_ydot^2+LAnkle_K
neeCoG_zdot^2);
	%
	LKnee_HipCoG_xdot=[jacobian(LKnee_HipCoG_x,teta1),jacobian(LKnee_HipCoG_x
,teta2),jacobian(LKnee_HipCoG_x,teta3),jacobian(LKnee_HipCoG_x,teta4),jacobian(
LKnee_HipCoG_x,teta5),jacobian(LKnee_HipCoG_x,teta6),jacobian(LKnee_HipCoG_x,te
ta7),jacobian(LKnee_HipCoG_x,teta8),jacobian(LKnee_HipCoG_x,teta9),jacobian(LKn
ee_HipCoG_x,teta10),jacobian(LKnee_HipCoG_x,teta11),jacobian(LKnee_HipCoG_x,tet
a12),jacobian(LKnee_HipCoG_x,teta13),jacobian(LKnee_HipCoG_x,teta14)]*dq.';
	LKnee_HipCoG_ydot=[jacobian(LKnee_HipCoG_y,teta1),jacobian(LKnee_HipCoG_y
,teta2),jacobian(LKnee_HipCoG_y,teta3),jacobian(LKnee_HipCoG_y,teta4),jacobian(
LKnee_HipCoG_y,teta5),jacobian(LKnee_HipCoG_y,teta6),jacobian(LKnee_HipCoG_y,te
ta7),jacobian(LKnee_HipCoG_y,teta8),jacobian(LKnee_HipCoG_y,teta9),jacobian(LKn
ee_HipCoG_y,teta10),jacobian(LKnee_HipCoG_y,teta11),jacobian(LKnee_HipCoG_y,tet
a12),jacobian(LKnee_HipCoG_y,teta13),jacobian(LKnee_HipCoG_y,teta14)]*dq.';
	LKnee_HipCoG_zdot=[jacobian(LKnee_HipCoG_z,teta1),jacobian(LKnee_HipCoG_z
,teta2),jacobian(LKnee_HipCoG_z,teta3),jacobian(LKnee_HipCoG_z,teta4),jacobian(
LKnee_HipCoG_z,teta5),jacobian(LKnee_HipCoG_z,teta6),jacobian(LKnee_HipCoG_z,te
ta7),jacobian(LKnee_HipCoG_z,teta8),jacobian(LKnee_HipCoG_z,teta9),jacobian(LKn
ee_HipCoG_z,teta10),jacobian(LKnee_HipCoG_z,teta11),jacobian(LKnee_HipCoG_z,tet
a12),jacobian(LKnee_HipCoG_z,teta13),jacobian(LKnee_HipCoG_z,teta14)]*dq.';
	VLKnee_HipCoG=sqrt(LKnee_HipCoG_xdot^2+LKnee_HipCoG_ydot^2+LKnee_HipCoG_z
dot^2);
	%
	UHipCoG_xdot=[jacobian(UHipCoG_x,teta1),jacobian(UHipCoG_x,teta2),jacobia
n(UHipCoG_x,teta3),jacobian(UHipCoG_x,teta4),jacobian(UHipCoG_x,teta5),jacobian
```

```
(UHipCoG_x,teta6),jacobian(UHipCoG_x,teta7),jacobian(UHipCoG_x,teta8),jacobian(
UHipCoG_x,teta9),jacobian(UHipCoG_x,teta10),jacobian(UHipCoG_x,teta11),jacobian
(UHipCoG_x,teta12),jacobian(UHipCoG_x,teta13),jacobian(UHipCoG_x,teta14)]*dq.';
       UHipCoG_ydot=[jacobian(UHipCoG_y,teta1),jacobian(UHipCoG_y,teta2),jacobia
n(UHipCoG_y,teta3),jacobian(UHipCoG_y,teta4),jacobian(UHipCoG_y,teta5),jacobian
(UHipCoG_y,teta6),jacobian(UHipCoG_y,teta7),jacobian(UHipCoG_y,teta8),jacobian(
UHipCoG_y,teta9),jacobian(UHipCoG_y,teta10),jacobian(UHipCoG_y,teta11),jacobian
(UHipCoG_y,teta12),jacobian(UHipCoG_y,teta13),jacobian(UHipCoG_y,teta14)]*dq.';
       UHipCoG_zdot=[jacobian(UHipCoG_z,teta1),jacobian(UHipCoG_z,teta2),jacobia
n(UHipCoG_z,teta3),jacobian(UHipCoG_z,teta4),jacobian(UHipCoG_z,teta5),jacobian
(UHipCoG_z,teta6),jacobian(UHipCoG_z,teta7),jacobian(UHipCoG_z,teta8),jacobian(
UHipCoG_z,teta9),jacobian(UHipCoG_z,teta10),jacobian(UHipCoG_z,teta11),jacobian
(UHipCoG_z,teta12),jacobian(UHipCoG_z,teta13),jacobian(UHipCoG_z,teta14)]*dq.';
       VUHipCoG=sqrt(UHipCoG_xdot^2+UHipCoG_ydot^2+UHipCoG_zdot^2);
       %
       RHip_KneeCoG_xdot=[jacobian(RHip_KneeCoG_x,teta1),jacobian(RHip_KneeCoG_x
,teta2),jacobian(RHip_KneeCoG_x,teta3),jacobian(RHip_KneeCoG_x,teta4),jacobian(
RHip_KneeCoG_x,teta5),jacobian(RHip_KneeCoG_x,teta6),jacobian(RHip_KneeCoG_x,te
ta7),jacobian(RHip_KneeCoG_x,teta8),jacobian(RHip_KneeCoG_x,teta9),jacobian(RHi
p_KneeCoG_x,teta10),jacobian(RHip_KneeCoG_x,teta11),jacobian(RHip_KneeCoG_x,tet
a12),jacobian(RHip_KneeCoG_x,teta13),jacobian(RHip_KneeCoG_x,teta14)]*dq.';
       RHip_KneeCoG_ydot=[jacobian(RHip_KneeCoG_y,teta1),jacobian(RHip_KneeCoG_y
,teta2),jacobian(RHip_KneeCoG_y,teta3),jacobian(RHip_KneeCoG_y,teta4),jacobian(
RHip_KneeCoG_y,teta5),jacobian(RHip_KneeCoG_y,teta6),jacobian(RHip_KneeCoG_y,te
ta7),jacobian(RHip_KneeCoG_y,teta8),jacobian(RHip_KneeCoG_y,teta9),jacobian(RHi
p_KneeCoG_y,teta10),jacobian(RHip_KneeCoG_y,teta11),jacobian(RHip_KneeCoG_y,tet
a12),jacobian(RHip_KneeCoG_y,teta13),jacobian(RHip_KneeCoG_y,teta14)]*dq.';
       RHip_KneeCoG_zdot=[jacobian(RHip_KneeCoG_z,teta1),jacobian(RHip_KneeCoG_z
,teta2),jacobian(RHip_KneeCoG_z,teta3),jacobian(RHip_KneeCoG_z,teta4),jacobian(
RHip_KneeCoG_z,teta5),jacobian(RHip_KneeCoG_z,teta6),jacobian(RHip_KneeCoG_z,te
ta7),jacobian(RHip_KneeCoG_z,teta8),jacobian(RHip_KneeCoG_z,teta9),jacobian(RHi
p_KneeCoG_z,teta10),jacobian(RHip_KneeCoG_z,teta11),jacobian(RHip_KneeCoG_z,tet
a12),jacobian(RHip_KneeCoG_z,teta13),jacobian(RHip_KneeCoG_z,teta14)]*dq.';
       VRHip_KneeCoG=sqrt(RHip_KneeCoG_xdot^2+RHip_KneeCoG_ydot^2+RHip_KneeCoG_z
dot^2);
       %
       RKnee_AnkleCoG_xdot=[jacobian(RKnee_AnkleCoG_x,teta1),jacobian(RKnee_Ankl
eCoG_x,teta2),jacobian(RKnee_AnkleCoG_x,teta3),jacobian(RKnee_AnkleCoG_x,teta4)
,jacobian(RKnee_AnkleCoG_x,teta5),jacobian(RKnee_AnkleCoG_x,teta6),jacobian(RKn
ee_AnkleCoG_x,teta7),jacobian(RKnee_AnkleCoG_x,teta8),jacobian(RKnee_AnkleCoG_x
,teta9),jacobian(RKnee_AnkleCoG_x,teta10),jacobian(RKnee_AnkleCoG_x,teta11),jac
obian(RKnee_AnkleCoG_x,teta12),jacobian(RKnee_AnkleCoG_x,teta13),jacobian(RKnee
_AnkleCoG_x,teta14)]*dq.';
       RKnee_AnkleCoG_ydot=[jacobian(RKnee_AnkleCoG_y,teta1),jacobian(RKnee_Ankl
eCoG_y,teta2),jacobian(RKnee_AnkleCoG_y,teta3),jacobian(RKnee_AnkleCoG_y,teta4)
,jacobian(RKnee_AnkleCoG_y,teta5),jacobian(RKnee_AnkleCoG_y,teta6),jacobian(RKn
ee_AnkleCoG_y,teta7),jacobian(RKnee_AnkleCoG_y,teta8),jacobian(RKnee_AnkleCoG_y
,teta9),jacobian(RKnee_AnkleCoG_y,teta10),jacobian(RKnee_AnkleCoG_y,teta11),jac
obian(RKnee_AnkleCoG_y,teta12),jacobian(RKnee_AnkleCoG_y,teta13),jacobian(RKnee
_AnkleCoG_y,teta14)]*dq.';
       RKnee_AnkleCoG_zdot=[jacobian(RKnee_AnkleCoG_z,teta1),jacobian(RKnee_Ankl
eCoG_z,teta2),jacobian(RKnee_AnkleCoG_z,teta3),jacobian(RKnee_AnkleCoG_z,teta4)
,jacobian(RKnee_AnkleCoG_z,teta5),jacobian(RKnee_AnkleCoG_z,teta6),jacobian(RKn
ee_AnkleCoG_z,teta7),jacobian(RKnee_AnkleCoG_z,teta8),jacobian(RKnee_AnkleCoG_z
,teta9),jacobian(RKnee_AnkleCoG_z,teta10),jacobian(RKnee_AnkleCoG_z,teta11),jac
```

```
obian(RKnee_AnkleCoG_z,teta12),jacobian(RKnee_AnkleCoG_z,teta13),jacobian(RKnee
_AnkleCoG_z,teta14)]*dq.';
       VRKnee_AnkleCoG=sqrt(RKnee_AnkleCoG_xdot^2+RKnee_AnkleCoG_ydot^2+RKnee_An
kleCoG_zdot^2);
       %
       RSoleCoG_xdot=[jacobian(RSoleCoG_x,teta1),jacobian(RSoleCoG_x,teta2),jaco
bian(RSoleCoG_x,teta3),jacobian(RSoleCoG_x,teta4),jacobian(RSoleCoG_x,teta5),ja
cobian(RSoleCoG_x,teta6),jacobian(RSoleCoG_x,teta7),jacobian(RSoleCoG_x,teta8),
jacobian(RSoleCoG_x,teta9),jacobian(RSoleCoG_x,teta10),jacobian(RSoleCoG_x,teta
11),jacobian(RSoleCoG_x,teta12),jacobian(RSoleCoG_x,teta13),jacobian(RSoleCoG_x
,teta14)]*dq.';
       RSoleCoG_ydot=[jacobian(RSoleCoG_y,teta1),jacobian(RSoleCoG_y,teta2),jaco
bian(RSoleCoG_y,teta3),jacobian(RSoleCoG_y,teta4),jacobian(RSoleCoG_y,teta5),ja
cobian(RSoleCoG_y,teta6),jacobian(RSoleCoG_y,teta7),jacobian(RSoleCoG_y,teta8),
jacobian(RSoleCoG_y,teta9),jacobian(RSoleCoG_y,teta10),jacobian(RSoleCoG_y,teta
11),jacobian(RSoleCoG_y,teta12),jacobian(RSoleCoG_y,teta13),jacobian(RSoleCoG_y
,teta14)]*dq.';
       RSoleCoG_zdot=[jacobian(RSoleCoG_z,teta1),jacobian(RSoleCoG_z,teta2),jaco
bian(RSoleCoG_z,teta3),jacobian(RSoleCoG_z,teta4),jacobian(RSoleCoG_z,teta5),ja
cobian(RSoleCoG_z,teta6),jacobian(RSoleCoG_z,teta7),jacobian(RSoleCoG_z,teta8),
jacobian(RSoleCoG_z,teta9),jacobian(RSoleCoG_z,teta10),jacobian(RSoleCoG_z,teta
11),jacobian(RSoleCoG_z,teta12),jacobian(RSoleCoG_z,teta13),jacobian(RSoleCoG_z
,teta14)]*dq.';
       VRSoleCoG=sqrt(RSoleCoG_xdot^2+RSoleCoG_xdot^2+RSoleCoG_xdot^2);
       %%
       T=0.5*m1*VLAnkle_KneeCoG^2+0.5*m2*VLKnee_HipCoG^2+0.5*m3*VUHipCoG^2+0.5*m
4*VRHip_KneeCoG^2+0.5*m5*VRKnee_AnkleCoG^2+0.5*m6*VRSoleCoG^2;
       U=m1*g*(LAnkle_KneeCoG_z-lA2c)+m2*g*(LKnee_HipCoG_z-
(lK2c+lA2K))+m3*g*(UHipCoG_z-(lK2H+lA2K+lH2cz))+m4*g*(RHip_KneeCoG_z-
(lK2c+lA2K+lS2A))+m5*g*(RKnee_AnkleCoG_z-(lS2A+lA2c))+m6*g*(RSoleCoG_z-lA2c);
       L=T-U;
       tau=jacobian(jacobian(T,dq),q)*dq.'+jacobian(jacobian(T,dq),dq)*ddq.'-
jacobian(T,q).'+jacobian(U,q).';

       n1=tau(1);n2=tau(2);n3=tau(3);n4=tau(4);n5=tau(5);n6=tau(6);n7=tau(7);n8=
tau(8);n9=tau(9);n10=tau(10);n11=tau(11);n12=tau(12);n13=tau(13);n14=tau(14);
       %%
       NoD=46;
       T=2;
       dt=T/NoD;
       time=dt:dt:T;

       tetan1=zeros(NoD);tetan2=zeros(NoD);tetan3=zeros(NoD);tetan4=zeros(NoD);
       tetan5=zeros(NoD);tetan6=zeros(NoD);tetan7=zeros(NoD);tetan8=90*sin(0:-
pi/180:-pi/4).'*pi/180;%teta8=[-
45:0].'*pi/180;tetan9=zeros(NoD);tetan10=zeros(NoD);tetan11=zeros(NoD);tetan12=
zeros(NoD);tetan13=zeros(NoD);tetan14=zeros(NoD);
       %%
       for i=2:NoD
           dqn1(i)=[tetan1(i)-tetan1(i-1)]/dt;    dqn2(i)=[tetan2(i)-tetan2(i-
1)]/dt;dqn3(i)=[tetan3(i)-tetan3(i-1)]/dt;dqn4(i)=[tetan4(i)-tetan4(i-
1)]/dt;dqn5(i)=[tetan5(i)-tetan5(i-1)]/dt;dqn6(i)=[tetan6(i)-tetan6(i-1)]/dt;
           dqn7(i)=[tetan7(i)-tetan7(i-1)]/dt;dqn8(i)=[tetan8(i)-tetan8(i-
1)]/dt;dqn9(i)=[tetan9(i)-tetan9(i-1)]/dt;dqn10(i)=[tetan10(i)-tetan10(i-
1)]/dt;dqn11(i)=[tetan11(i)-tetan11(i-1)]/dt;dqn12(i)=[tetan12(i)-tetan12(i-
1)]/dt;dqn13(i)=[tetan13(i)-tetan13(i-1)]/dt;dqn14(i)=[tetan14(i)-tetan14(i-
1)]/dt;
       end

       for i=2:NoD
       ddqn1(i)=[dqn1(i)-dqn1(i-1)]/dt;ddqn2(i)=[dqn2(i)-dqn2(i-1)]/dt;
           ddqn3(i)=[dqn3(i)-dqn3(i-1)]/dt;ddqn4(i)=[dqn4(i)-dqn4(i-1)]/dt;
           ddqn5(i)=[dqn5(i)-dqn5(i-1)]/dt;ddqn6(i)=[dqn6(i)-dqn6(i-1)]/dt;
           ddqn7(i)=[dqn7(i)-dqn7(i-1)]/dt;ddqn8(i)=[dqn8(i)-dqn8(i-1)]/dt;
           ddqn9(i)=[dqn9(i)-dqn9(i-1)]/dt;ddqn10(i)=[dqn10(i)-dqn10(i-1)]/dt;
           ddqn11(i)=[dqn11(i)-dqn11(i-1)]/dt;ddqn12(i)=[dqn12(i)-dqn12(i-
1)]/dt;ddqn13(i)=[dqn13(i)-dqn13(i-1)]/dt;ddqn14(i)=[dqn14(i)-dqn14(i-1)]/dt;
       end

       %%
           dqn1(1)=dqn1(2);dqn2(1)=dqn2(2);dqn3(1)=dqn3(2);dqn4(1)=dqn4(2);
dqn5(1)=dqn5(2);dqn6(1)=dqn6(2);dqn7(1)=dqn7(2);dqn8(1)=dqn8(2);
dqn9(1)=dqn9(2);dqn10(1)=dqn10(2);dqn11(1)=dqn11(2);
dqn12(1)=dqn12(2);dqn13(1)=dqn13(2);dqn14(1)=dqn14(2);

           ddqn1(1) =ddqn1(3);ddqn2(1) =ddqn2(3);ddqn3(1) =ddqn3(3);ddqn4(1)
=ddqn4(3);ddqn5(1) =ddqn5(3);ddqn6(1) =ddqn6(3);ddqn7(1) =ddqn7(3);ddqn8(1)
=ddqn8(3);ddqn9(1) =ddqn9(3);ddqn10(1)=ddqn10(3);
ddqn11(1)=ddqn11(3);ddqn12(1)=ddqn12(3);dqn13(1)=ddqn13(3);ddqn14(1)=ddqn14(3);

           ddqn1(2) =ddqn1(3);ddqn2(2) =ddqn2(3);
           ddqn3(2) =ddqn3(3);ddqn4(2) =ddqn4(3);
           ddqn5(2) =ddqn5(3);ddqn6(2) =ddqn6(3);
           ddqn7(2) =ddqn7(3);ddqn8(2) =ddqn8(3);
           ddqn9(2) =ddqn9(3);ddqn10(2)=ddqn10(3);
           ddqn11(2)=ddqn11(3);ddqn12(2)=ddqn12(3);
           ddqn13(2)=ddqn13(3);ddqn14(2)=ddqn14(3);

       %%
       for i=1:NoD
           g1(i,1)= subs(n1,dq1,dqn1(i));
           g1(i,1)= subs(g1(i,1),dq2,dqn2(i));
           g1(i,1)= subs(g1(i,1),dq3,dqn3(i));
           g1(i,1)= subs(g1(i,1),dq4,dqn4(i));
           g1(i,1)= subs(g1(i,1),dq5,dqn5(i));
           g1(i,1)= subs(g1(i,1),dq6,dqn6(i));
           g1(i,1)= subs(g1(i,1),dq7,dqn7(i));
           g1(i,1)= subs(g1(i,1),dq8,dqn8(i));
           g1(i,1)= subs(g1(i,1),dq9,dqn9(i));
           g1(i,1)= subs(g1(i,1),dq10,dqn10(i));
           g1(i,1)= subs(g1(i,1),dq11,dqn11(i));
           g1(i,1)= subs(g1(i,1),dq12,dqn12(i));
           g1(i,1)= subs(g1(i,1),dq13,dqn13(i));
           g1(i,1)= subs(g1(i,1),dq14,dqn14(i));

           g1(i,1)= subs(g1(i,1),ddq1,ddqn1(i));
           g1(i,1)= subs(g1(i,1),ddq2,ddqn2(i));
           g1(i,1)= subs(g1(i,1),ddq3,ddqn3(i));
           g1(i,1)= subs(g1(i,1),ddq4,ddqn4(i));
           g1(i,1)= subs(g1(i,1),ddq5,ddqn5(i));
```

```
    g1(i,1)= subs(g1(i,1),ddq6,ddqn6(i));
    g1(i,1)= subs(g1(i,1),ddq7,ddqn7(i));
    g1(i,1)= subs(g1(i,1),ddq8,ddqn8(i));
    g1(i,1)= subs(g1(i,1),ddq9,ddqn9(i));
    g1(i,1)= subs(g1(i,1),ddq10,ddqn10(i));
    g1(i,1)= subs(g1(i,1),ddq11,ddqn11(i));
    g1(i,1)= subs(g1(i,1),ddq12,ddqn12(i));
    g1(i,1)= subs(g1(i,1),ddq13,ddqn13(i));
    g1(i,1)= subs(g1(i,1),ddq14,ddqn14(i));

    g1(i,1)= subs(g1(i,1),teta1,tetan1(i));
    g1(i,1)= subs(g1(i,1),teta2,tetan2(i));
    g1(i,1)= subs(g1(i,1),teta3,tetan3(i));
    g1(i,1)= subs(g1(i,1),teta4,tetan4(i));
    g1(i,1)= subs(g1(i,1),teta5,tetan5(i));
    g1(i,1)= subs(g1(i,1),teta6,tetan6(i));
    g1(i,1)= subs(g1(i,1),teta7,tetan7(i));
    g1(i,1)= subs(g1(i,1),teta8,tetan8(i));
    g1(i,1)= subs(g1(i,1),teta9,tetan9(i));
    g1(i,1)= subs(g1(i,1),teta10,tetan10(i));
    g1(i,1)= subs(g1(i,1),teta11,tetan11(i));
    g1(i,1)= subs(g1(i,1),teta12,tetan12(i));
    g1(i,1)= subs(g1(i,1),teta13,tetan13(i));
    g1(i,1)= subs(g1(i,1),teta14,tetan14(i));
    g1(i,1)= vpa(g1(i,1));
end
%% Plots
figure(12)
plot(g1);
```