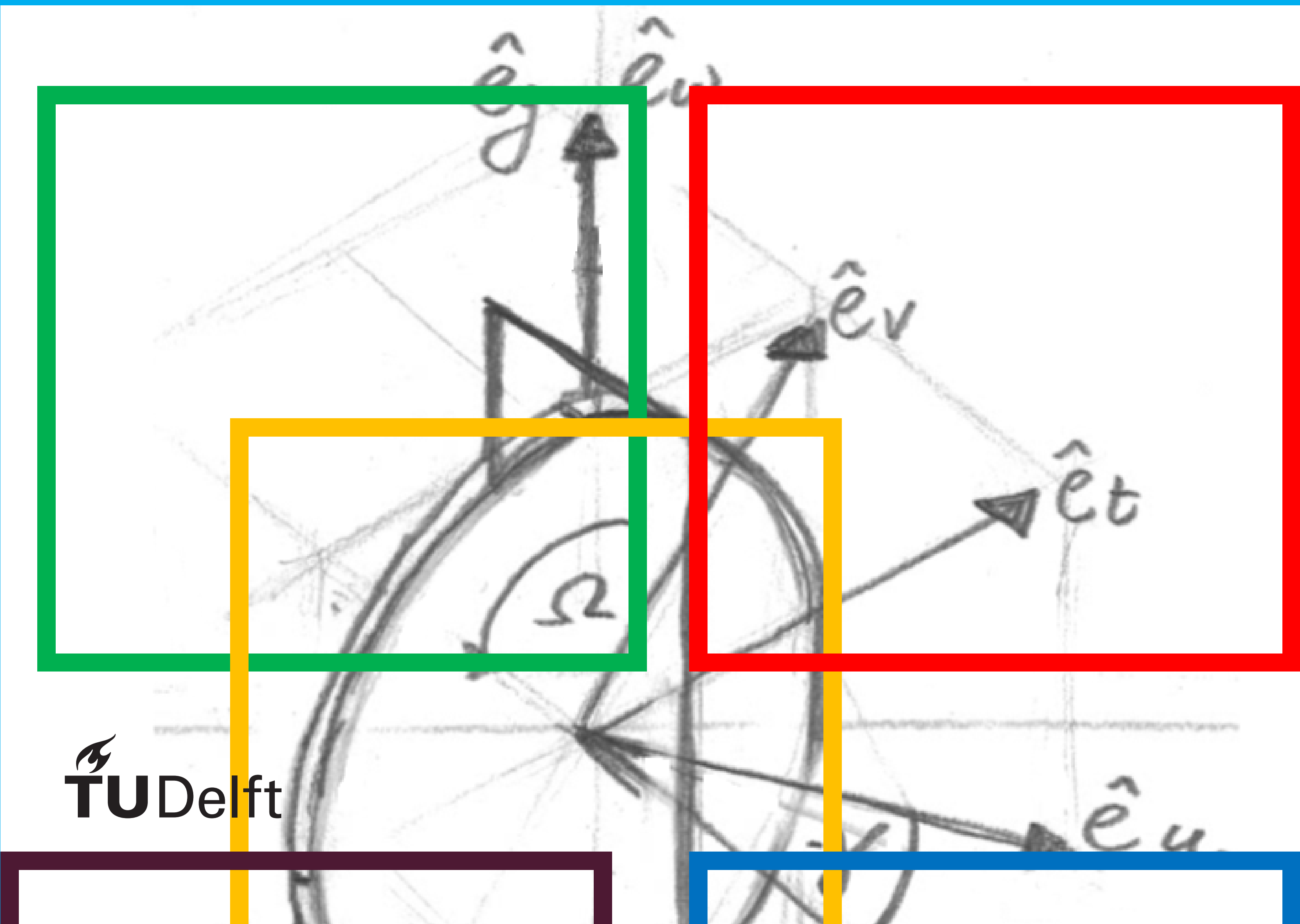
Optimal Design of a Passively-Controlled Gyro for Balance Assistance

R.M. Helwig

May the torque be with you



Optimal Design of a Passively-Controlled Gyro for Balance Assistance

by

R.M. Helwig

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Friday February 28, 2020 at 13:30 pM.

Student number:	4480090	
Project duration:	May , 2019 – February , 2020	
Thesis committee:	Prof. dr. ing. H. Vallery,	TU Delft, supervisor
	Ir. A. Berry	TU Delft
	MSc. B. Sterke	TU Delft
	Dr. Ir. D. Lemus	TU Delft
	Dr. Ir. A. Sakes	TU Delft, external member

This thesis is confidential and cannot be made public until February 28, 2022.

An electronic version of this thesis is available at http://repository.tudelft.nl/.

TU Delft

Abstract

Falling is a significant problem for older adults. It can cause severe injury and even death. Furthermore, the fear of falling has a significant influence on the life of the elderly, and therefore they reduce their physical activity. Two new balance assistive devices are being developed to reduce the risk of falling. Both devices use a control moment gyroscope (CMG) to generate a moment to counter the falling motion. One device consists of a single CMG. The other device consists of two CMGs that are coupled such that the gimbals rotate in opposite direction. This is called a scissored pair CMG (SPCMG). The purpose of this study was to examine whether it is possible to design an (SP)CMG with a passive mechanism that exploits gyroscopic precession of gimbal(s) to emulate different types of impedances for balance assistance.
To examine this, first, the equations of motion of a CMG and an SPCMG were derived. Next, the equations of motion were used to derive the impedance of the system. The impedance was optimized such that it would simulate the behaviour of a spring, a damper, a mass, a mass-spring-damper system, and a rotational PD controller which is proportional to the XCoM (PDXCoM), a measure of stability. The optimization used a gradient-based algorithm to find the minimum. Multiple optimizations with different random initial guesses were performed to increase the chance to find the global minimum. Two sets of optimizations were performed. One optimization with and one optimization without bounds on the optimization. The sets parameters that led to the best fit were used in a walking simulation to calculate the moments the device would generate during normal walking.
It is shown that it is possible to simulate the dynamics of a spring, a damper, a mass, and a mass-spring-damper system with a CMG and an SPCMG. However, it was not possible to replicate the dynamics of the PDXCoM with a CMG and an SPCMG. A walking simulation showed that the generated moments of the (SP)CMG were in the opposite direction of the angular velocity of the human. Therefore, using a passive mechanism to control an (SP)CMG could be used as balance assistance.

Preface

This thesis was made to describe the research to passively exploit gyroscopic presession to control a gyroscope for balance assistance. This thesis would not have been possible without the help of my daily supervisors, Andrew Berry, Bram Sterke and Daniel Lemus. Furthermore, I would like to thank Heike Vallery for the supervision of this project. I also would like to thank my all friends from both within the TU Delft, as outside TU Delft, who helped me pass courses, and gave me moral support while writing this thesis. Last but definitely not least, I would like to thank my family for supporting me during all those years I spend studying.

R.M. Helwig

Contents

Nomenclature

Table 1: Nomenclature list

Symbol	Meaning
$\boldsymbol{R}$	Vector R
$\mathbf{R}$	Matrix R
Ω	Angular velocity of the flywheel
$\dot{\gamma}$	Angular velocity of the gimbal
$\boldsymbol{F}$	Force vector
$\boldsymbol{H}$	Angular momentum vector
$\boldsymbol{M}$	Moment vector
${}^{\mathcal{Q}}\boldsymbol{R}$	Vector R expressed in the $\mathcal{Q}$ frame
$({}^{\mathcal{Q}}\dot{\boldsymbol{R}})_{\mathcal{S}}$	Change of R with respect to the $\mathcal{S}$ frame, expressed in the $\mathcal{Q}$ frame
$\boldsymbol{\omega}$	Angular velocity of the (human) body
$\{\hat{\boldsymbol{e}}_s, \hat{\boldsymbol{e}}_t, \hat{\boldsymbol{e}}_g\}$	Gimbal fixed frame
$\{\hat{\boldsymbol{e}}_u, \hat{\boldsymbol{e}}_v, \hat{\boldsymbol{e}}_w\}$	Body fixed frame
$\mathscr{L}$	Laplace transform
$\mathcal{D}$	Discriminant

1
Introduction

1.1. Motivation

Older adults are more likely to lose their balance and fall. This can cause serious injury, immobility, premature nursing home placement, and even death [37]. In 2002, about 1000 people older than 50 years died because of falling in Finland, a population of about 5 million people [20]. The fear of falling has a high impact on the lives of the elderly. About a third of the elderly is afraid to fall [41]. Due to the fear of falling, the elderly decrease their physical activity. This decrease in physical activity can cause deconditioning, reduced- health, physical functioning and participation in society [38], which lead to an increased risk of falling. Risk factors for falling can be classified into intrinsic and extrinsic. The most important intrinsic factors are fatigue, the use of medication, muscle weakness, balance deficit, and mobility limitations[11, 18]. Extrinsic factors are mainly interaction with the environment [18]. This can include unexpected steps or changes in grade, and terrain that is slippery, or loose.
Humans have a variety of balance techniques. One such technique is to produce a moment around the ankle to keep the body upright. To generate this moment, the plantar- and dorsiflexors around the ankle are used to control the human body. This ankle strategy only works for perturbations with a frequency lower than 1 Hz and with a small amplitude [1, 22]. For perturbations with a higher frequency, the hip strategy is used. With this, the upper body is moved in the opposite direction of the lower body[1, 22]. These techniques are used during stance. The task of balance is to keep the centre of gravity above the base support. During walking, the base support is small since the human is only supported on one foot. Therefore, walking is a challenging daily activity to maintain balance [43]. Keeping balance becomes even harder since, during walking, humans have to initiate, and terminate gait, avoid objects and thereby altering the gait cycle, and might bump into objects or other people. It is during walking that about 50% of all falls occur [2]. The primary way to prevent a fall is a correct foot placement and body sway, such that the centre of gravity is above the foot. To do this, response time is of great importance [40]. About a third of all falls occur because the response time was too long [34]. With longer recovery time, the response time of the person can be slower.

Fall prevention programs are used to teach the elderly how to manoeuvre better and how to fall. Here, robots like KineAssist [33] are already used to reduce the workload of physiotherapists and increase training intensity. Additionally, technical solutions are being proposed to prevent falling. This includes a robotic cane [7], which moves to a position where it is able to support the falling human. And a stroller-like robot with actuated arms [12] that give support to the user. For these devices, the user has to use one or both arms to keep balance. Additionally, it requires the user to actively provide a force to prevent falling. Therefore, a certain strength is needed for the user to stay upright. An older person might not be able to provide the necessary amount of force needed to do this. Another assistive device is a wearable robot with two legs that can move to a posture to provide assistance [31]. This design is however very bulky which makes manoeuvring in compact spaces, like in a living room, more difficult. Apart from these robotic devices, also exoskeletons like, Ekso(Ekso Bionics, USA), XoR [16], and BALANCE (EU) are used for balance control. These are strong enough to move limbs and are therefore bulky, and complicated to use. Moreover, the actuation that the exoskeletons provide generates internal moments. Therefore, it does not directly change the angular momentum of the body.

Another creative, solution for fall prevention is proposed by Li and Vallery [25]. Here, control moment gyroscopes (CMGs) are used to create a moment to counter the falling motion. If a flywheel has a high angular velocity and it is rotated about a second axis, a moment about a third axis is generated. This moment can be used to prevent falling or reduce the falling speed to give the person extra time to recover.

This concept of using a gyroscope is minimalistic and allows the user to keep their hands free. Moreover, many people with balance impairment are functionally capable of walking and thus do not need full muscle support. They only need assistance for fall prevention, and therefore an exoskeleton is unnecessary. The concept of using a CMG for balance assistance has gained some momentum over time. Scissored paired CMGs have been used to steer the moment provided by the CMGs in the desired direction and prevent sway [6, 36]. Furthermore, a prototype has been developed using an inverted pendulum to replace a human [24]. Here they were able to produce a CMG moment of 70 N m. All of these concepts, however, use a motor to control the gimbal. This motor adds weight due to the transmission, and the battery, which is undesirable. Passive control also requires no sensors, is therefore very fast and reliable.

Currently CMGs are mainly used to steer satellites and other space crafts [23] or to stabilize ships [32]. Here, the angular velocity of the base structure is low and will, therefore, not induce a significant gyroscopic effect. Furthermore, obects with a high angular velocity have been stabilized using a CMG such as bicycles [3], robots [5], and a ropeway carrier [30].

Also in wearable applications, the angular velocities can be large enough to induce a significant gyroscopic effect. It might be possible to use this effect to control the CMG. If the CMG is controlled passively via direct mechanical coupling, it will overcome some drawbacks that active control entails. A significant drawback that active control brings is time delay, which reduces the predictability of the device. Moreover, some electronics might fail. With a mechanical coupling, there is no time delay and no electronics.

1.2. Background information

To understand the rest of the report, some backgournd information is needed about CMGs and bodeplots. Reaction wheels and CMGs can both be used to generate a moment by changing the angular momentum of the flywheel. A reaction wheel accelerates or decelerates its flywheel about the spin axis and thereby generates a moment. CMGs also have a rotation flywheel, but they generate a moment by a rotation about a different axis than the flywheel spin axis. This is typically done by rotating a gimbal. This produces moments that are much larger than a reaction wheel could provide. This moment will be orthogonal to both the spin axis of the flywheel and the gimbal.

To control the gyroscope, the dynamics of the gyroscope will be used. When the gimbal rotates about the $\hat{\boldsymbol{e}}_t$ axis and the flywheel has an angular momentum in direction $\hat{\boldsymbol{e}}_s$, see Fig. 1.1, a torque will be generated about an axis perpendicular to both $\hat{\boldsymbol{e}}_t$ and $\hat{\boldsymbol{e}}_s$. To determine in which direction the torque is generated, the right-hand rule is used. The thumb points in the direction of the angular velocity of the gimbal and the index finger in the direction of the angular momentum. This shows that the torque is generated in the positive $\hat{\boldsymbol{e}}_g$ direction. This moment will start to rotate the flywheel about this $\hat{\boldsymbol{e}}_g$ axis and therefore a new moment is generated perpendicular to $\hat{\boldsymbol{e}}_s$ and $\hat{\boldsymbol{e}}_g$, which will be in the $\hat{\boldsymbol{e}}_t$ direction. This is called the cascaded gyroscopic effect. This means that a gyroscope has an output torque in the opposite direction of the input angular velocity.

When the output of a system is in the opposite dirction of the input, a system has a phase of 180 deg or it is non-minimum phase [10]. At least one zero exists in the right-half plane when a system is non-minimum phase. In the result section, the frequency responses of the (SP)CMG are shown with different parameters. Therefore it is imporant to be able to interpret bodeplots. When drawing the bode plot of a non-minimum phase system, the normal "rule book" for drawing bode plots do not apply. For drawing a bode plot of a non-minimum phase system, some rules have to added. These can be seen in Table 1.1. Non-minimum phase system can have a "strange" behaviour. When an odd number of zeros exist in the RHP, the initial direction of the step response will be in the opposite direction of the final value [13].

1.3. Project overview

The research question of this project is; "Is it possible to design a (SP)CMG with a passive mechanism, such that (SP)CMG dynamics can be exploited in a way that it can generate effective moments for balance assistance?"

The goal of this thesis is to investigate whether it is possible to passively exploit a (SP)CMG for balance assistance. This will be done by making a theoretical model of an (SP)CMG with a passive mechanism, which will be optimized such that it can replicate the impedance of arbitrary systems. The scope of this project will be limited to theoretical analysis and using measured data to predict the moments the (SP)CMG will generate.

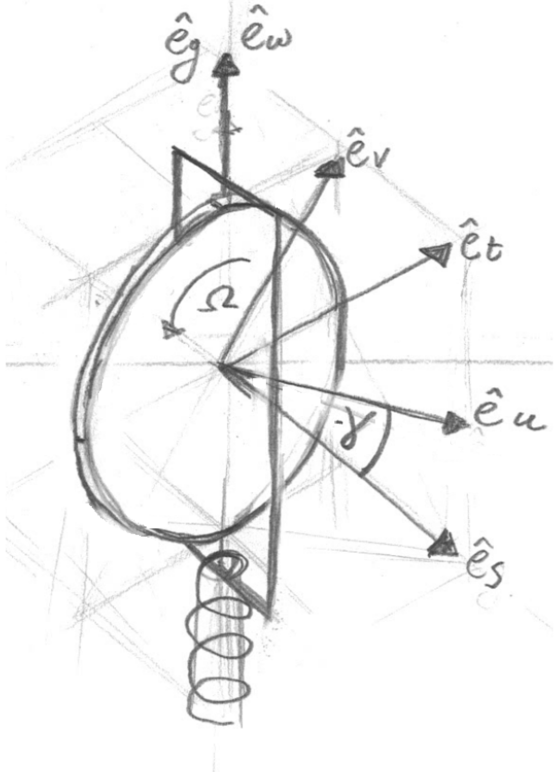


Figure 1.1: Hand sketch of flywheel with gimbal. The body-fixed frame, $\{\hat{\boldsymbol{e}}_u, \hat{\boldsymbol{e}}_v, \hat{\boldsymbol{e}}_w\}$is rotated with an angle γ with respect to the gimbal-fixed frame, $\{\hat{\boldsymbol{e}}_s, \hat{\boldsymbol{e}}_t, \hat{\boldsymbol{e}}_g\}$. The flywheel rotates with an angular velocity of Ω. The spring and damper provide a moment along the $\hat{\boldsymbol{e}}_w / \hat{\boldsymbol{e}}_g$ axis.

There will be no experiments on humans subjects.
In Chapter 2, the equations of motion of a CMG and SPCMG will be explained as well as how the transfer function are obtained and the optimization method. In Chapter 3, a case study will be discussed. Herein, specific impedances will be chosen, and the CMG impedance will be matched to this. In Chapter 4, the results of the parameter optimization are shown. Furthermore, the time response of the CMG and SPCMG are shown with one set of optimized parameters. In Chapter 5, the results and method will be discussed as well as future directions. The conclusion will be given in Chapter 6. Additional graphs and formulas can be found in the appendices, as well as the Matlab code that was used.

Table 1.1: Table with rules for drawing bode plots

		Magnitude	Phase	Initial Phase
Minimum Phase	Zero	20 dB/dec	+90°	0°
	Double Zero	40 dB/dec	+180°	
	Pole	−20 dB/dec	−90°	
	Double Pole	−40 dB/dec	−180°	
Non Minimum Phase	Zero	20 dB/dec	−90°	−180°
	Pole	−20 dB/dec	+90°	

2

Mechanism Design

In this chapter, the equations of motions of a single CMG and SPCMG are derived. These are then used to obtain the impedances. The impedance is then optimized such that the (SP)CMG simulates the behaviour of simple mechanical systems.

2.1. Equations of motion for a single CMG

In this section, the equations of motion are derived for the single CMG. A CMG system is composed of a flywheel, with moment of inertia tensor $\mathbf{I}_\mathrm{w}$ with values I_ws, I_wt and I_wt on the diagonal, spinning at a high angular velocity (Ω). Moreover, a gimbal with a moment of inertia tensor $\mathbf{I}_\mathrm{g}$ with values I_gs, I_gt and I_gg on the diagonal, can rotate with respect to the body with angular velocity $\dot{\gamma}$. We propose, a passive mechanism between the human body and the gimbal, consisting of a spring with stiffness k and a damper with damping coefficient b. This passive mechanism provides a moment to the gimbal. The equations of motion are in the body-fixed frame with both the Newton-Euler methods and the Lagrange methods.

2.1.1. Definitions of angles and angular velocities

The equations of motion are generated for body fixed sensing. The term body refers to the human body. The body-fixed frame ($\mathcal{B}$) consists of unit vectors $\{\hat{e}_u, \hat{e}_v, \hat{e}_w\}$. Where $\hat{\boldsymbol{e}}_u$ is in the direction of the left-right axis where the positive direction is right, $\hat{\boldsymbol{e}}_v$ is in the direction of the sagittal axis where the positive direction is ventral, and $\hat{\boldsymbol{e}}_w$ is in the longitudinal direction of the human where the positive direction is cranial. The definitions can all be seen in Fig. 2.2. The gimbal-fixed frame ($\mathcal{G}$) consists of the unit vectors $\{\hat{e}_s, \hat{e}_t, \hat{e}_g\}$, see Fig. 2.2. The projections of the body-fixed frame on the gimbal-fixed frame can be seen in Fig. 2.1 and are defined as follows:

$$^{\mathcal{G}}\hat{\boldsymbol{e}}_u = \begin{pmatrix} \cos(-\gamma) \\ \sin(-\gamma) \\ 0 \end{pmatrix}, \quad ^{\mathcal{G}}\hat{\boldsymbol{e}}_v = \begin{pmatrix} -\sin(-\gamma) \\ \cos(-\gamma) \\ 0 \end{pmatrix}, \quad ^{\mathcal{G}}\hat{\boldsymbol{e}}_w = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}, \tag{2.1}$$

The rotation matrix from the body-fixed frame to the gimbal-fixed frame is:

$$^{\mathcal{G}}\mathbf{R}(\gamma)_{\mathcal{B}} \quad = [\hat{\boldsymbol{e}}_u \quad \hat{\boldsymbol{e}}_v \quad \hat{\boldsymbol{e}}_w] \tag{2.2}$$

The rotation matrix from the gimbal-fixed frame to the body-fixed frame is the transpose of Eq. (2.2). This will results in, $^{\mathcal{B}}\mathbf{R}(\gamma)_{\mathcal{G}} = ^{\mathcal{G}}\mathbf{R}(\gamma)_{\mathcal{B}}^T$. The angular velocities between the wheel fixed frame ($\mathcal{W}$) and the gimbal fixed frame ($\mathcal{G}$), angular velocities between the $\mathcal{G}$ and the body fixed frame ($\mathcal{B}$) are expressed as:

$$^{\mathcal{G}}\boldsymbol{\omega}_{\mathcal{W}/\mathcal{G}} = \begin{pmatrix} \Omega \\ 0 \\ 0 \end{pmatrix}, \quad ^{\mathcal{G}}\boldsymbol{\omega}_{\mathcal{G}/\mathcal{B}} = \begin{pmatrix} 0 \\ 0 \\ \dot{\gamma} \end{pmatrix} \tag{2.3}$$

The angular velocity between $\mathcal{B}$ and the inertial frame ($\mathcal{N}$) is expressed as:

$$^{\mathcal{B}}\boldsymbol{\omega}_{\mathcal{B}/\mathcal{N}} = \begin{pmatrix} \omega_u \\ \omega_v \\ \omega_w \end{pmatrix} \tag{2.4}$$

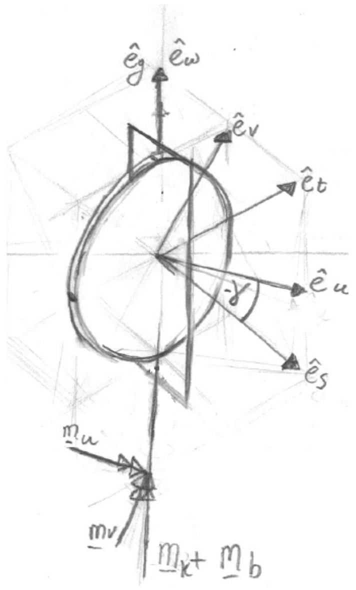


Figure 2.1: Free body diagram of a flywheel with a gimbal. The body-fixed frame, $\{\hat{\boldsymbol{e}}_u, \hat{\boldsymbol{e}}_v, \hat{\boldsymbol{e}}_w\}$is rotated with an angle γ with respect to the gimbal-fixed frame, $\{\hat{\boldsymbol{e}}_s, \hat{\boldsymbol{e}}_t, \hat{\boldsymbol{e}}_g\}$. The flywheel rotates with an angular velocity of Ω. The moments $\boldsymbol{M}_u$ and $\boldsymbol{M}_v$ are the reaction moments of the bearing in the $\hat{\boldsymbol{e}}_u, \hat{\boldsymbol{e}}_v$ respectively. The moments $\boldsymbol{M}_k$ and $\boldsymbol{M}_b$ are generated by a spring with spring stiffness k and a damper with damping coefficient b respectively.

From this it follow that the angular velocity between $\mathcal{G}$ and $\mathcal{N}$ is ${}^{\mathcal{G}}\boldsymbol{\omega}_{\mathcal{G}/\mathcal{N}} = {}^{\mathcal{G}}\boldsymbol{\omega}_{\mathcal{G}/\mathcal{B}} + {}^{\mathcal{G}}\mathbf{R}(\gamma)^{\mathcal{B}}_{\mathcal{B}}\boldsymbol{\omega}_{\mathcal{B}/\mathcal{N}}$.

2.1.2. Newton-Euler Approach for a Single CMG with body-fixed Rotations

The Newton-Euler method was used to generate the equations of motion. The angular momentum of flywheel and gimbal in the gimbal-fixed frame are:

$$\begin{aligned}
{}^{\mathcal{G}}\boldsymbol{H}_{\mathrm{w}} &= \mathbf{I}_{\mathrm{w}}({}^{\mathcal{G}}\boldsymbol{\omega}_{\mathcal{W}/\mathcal{G}} + {}^{\mathcal{G}}\boldsymbol{\omega}_{\mathcal{G}/\mathcal{B}} + {}^{\mathcal{G}}\mathbf{R}(\gamma)^{\mathcal{B}}_{\mathcal{B}}\boldsymbol{\omega}_{\mathcal{B}/\mathcal{N}}) \\
{}^{\mathcal{G}}\boldsymbol{H}_{\mathrm{g}} &= \mathbf{I}_{\mathrm{g}}({}^{\mathcal{G}}\boldsymbol{\omega}_{\mathcal{G}/\mathcal{B}} + {}^{\mathcal{G}}\mathbf{R}(\gamma)^{\mathcal{B}}_{\mathcal{B}}\boldsymbol{\omega}_{\mathcal{B}/\mathcal{N}}) \\
{}^{\mathcal{G}}\boldsymbol{H} &= {}^{\mathcal{G}}\boldsymbol{H}_{\mathrm{b}} + {}^{\mathcal{G}}\boldsymbol{H}_{\mathrm{g}}
\end{aligned} \tag{2.5}$$

To calculate the change of angular momentum with respect to the $\mathcal{N}$ frame, we will first derive the change of angular momentum with respect to the $\mathcal{G}$ frame. Since we assume that Ω is constant, the derivative of ${}^{\mathcal{G}}\boldsymbol{\omega}_{\mathcal{W}/\mathcal{G}}$ equals zero. Therefore, ${}^{\mathcal{G}}(\dot{\boldsymbol{H}})_{\mathcal{G}}$ can be calculated as follows.

$$\begin{aligned}
{}^{\mathcal{G}}(\dot{\boldsymbol{H}}_{\mathrm{w}})_{\mathcal{G}} &= \mathbf{I}_{\mathrm{w}}({}^{\mathcal{G}}(\dot{\boldsymbol{\omega}}_{\mathcal{G}/\mathcal{B}})_{\mathcal{G}} + {}^{\mathcal{G}}\mathbf{R}(\gamma)^{\mathcal{B}}_{\mathcal{B}}(\dot{\boldsymbol{\omega}}_{\mathcal{B}/\mathcal{N}})_{\mathcal{G}}) \\
{}^{\mathcal{G}}(\dot{\boldsymbol{H}}_{\mathrm{g}})_{\mathcal{G}} &= \mathbf{I}_{\mathrm{g}}({}^{\mathcal{G}}(\dot{\boldsymbol{\omega}}_{\mathcal{G}/\mathcal{B}})_{\mathcal{G}} + {}^{\mathcal{G}}\mathbf{R}(\gamma)^{\mathcal{B}}_{\mathcal{B}}(\dot{\boldsymbol{\omega}}_{\mathcal{B}/\mathcal{N}})_{\mathcal{G}}) \\
{}^{\mathcal{G}}(\dot{\boldsymbol{H}})_{\mathcal{G}} &= {}^{\mathcal{G}}(\dot{\boldsymbol{H}}_{\mathrm{w}})_{\mathcal{G}} + {}^{\mathcal{G}}(\dot{\boldsymbol{H}}_{\mathrm{g}})_{\mathcal{G}}
\end{aligned} \tag{2.6}$$

To derive the derivative of ${}^{\mathcal{B}}\boldsymbol{\omega}_{\mathcal{B}/\mathcal{N}}$ with respect to the $\mathcal{G}$ frame, we need to use the transport theorem.

$${}^{\mathcal{B}}(\dot{\boldsymbol{\omega}}_{\mathcal{B}/\mathcal{N}})_{\mathcal{G}} = {}^{\mathcal{B}}(\dot{\boldsymbol{\omega}}_{\mathcal{B}/\mathcal{N}})_{\mathcal{B}} + {}^{\mathcal{B}}\boldsymbol{\omega}_{\mathcal{B}/\mathcal{G}} \times {}^{\mathcal{B}}\boldsymbol{\omega}_{\mathcal{B}/\mathcal{N}} \tag{2.7}$$

Now we can derive the change of angular momentum with respect to the $\mathcal{N}$ frame by using the transport theorem again.

$${}^{\mathcal{G}}(\dot{\boldsymbol{H}})_{\mathcal{N}} = {}^{\mathcal{G}}(\dot{\boldsymbol{H}})_{\mathcal{G}} + {}^{\mathcal{G}}\boldsymbol{\omega}_{\mathcal{G}/\mathcal{N}} \times {}^{\mathcal{G}}\boldsymbol{H} \tag{2.8}$$

So ${}^{\mathcal{B}}(\dot{\boldsymbol{H}})_{\mathcal{N}} = {}^{\mathcal{B}}\mathbf{R}(\gamma)^{\mathcal{G}}_{\mathcal{G}}(\dot{\boldsymbol{H}})_{\mathcal{N}}$. The written out form of this equation can be seen in Eq. (A.1). The moments generated by the spring, damper and the bearings are:

$${}^{\mathcal{B}}\boldsymbol{M} = \begin{pmatrix} M_u \\ M_v \\ +b\dot{\gamma} + k(\gamma - \gamma 0) \end{pmatrix} \tag{2.9}$$

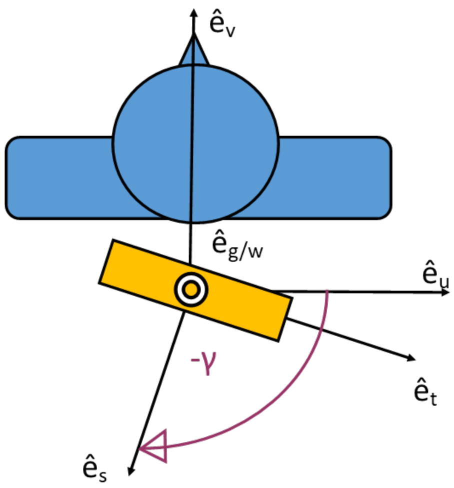


Figure 2.2: Diagram of the body fixed frame, $\{\hat{\boldsymbol{e}}_u, \hat{\boldsymbol{e}}_v, \hat{\boldsymbol{e}}_u\}$ and the gimbal fixed frame $\{\hat{\boldsymbol{e}}_s, \hat{\boldsymbol{e}}_t, \hat{\boldsymbol{e}}_g\}$

Using Euler's 2nd law of motion, we state:

$$^{\mathcal{B}}\boldsymbol{M} = -^{\mathcal{B}}(\dot{\boldsymbol{H}})_{\mathcal{N}} \tag{2.10}$$

This can be solved for $\ddot{\gamma}$ which leads to:

$$\begin{gathered}\ddot{\gamma} = -[b\dot{\gamma} - k(\gamma_0 - \gamma) + \dot{\omega}_w(I_{\text{gg}} + I_{\text{wt}}) - I_{\text{gs}}(\omega_u \cos\gamma + \omega_v \sin\gamma)(\omega_v \cos\gamma - \omega_u \sin\gamma) + I_{gt}(\omega_u \cos\gamma + \omega_v \sin\gamma) \\ (\omega_v \cos\gamma - \omega_u \sin\gamma) + I_{\text{wt}}(\omega_u \cos\gamma + \omega_v \sin\gamma)(\omega_v \cos\gamma - \omega_u \sin\gamma) - I_{\text{ws}}(\omega_v \cos\gamma - \omega_u \sin\gamma) \\ (\Omega + \omega_u \cos\gamma + \omega_v \sin\gamma)]/(I_{\text{gg}} + I_{\text{wt}})\end{gathered} \tag{2.11}$$

2.2. Frequency response analysis of a single CMG

The goal of this subsection is to generate equations to describe the impedance, $\frac{M_i}{\omega_i}$, of the system. This impedance denotes the change in moment due to a rotation disturbance. Generating the impedance is done by using the moments due the change in angular momentum that act on the human body, $^{\mathcal{G}}\boldsymbol{M}$. The moment is not solely dependent on $\boldsymbol{\omega}_{\mathcal{B}/\mathcal{N}}$ but also on $\gamma, \dot{\gamma}$, and $\ddot{\gamma}$. Therefore, the dynamics of $\ddot{\gamma}$ must be implicitly included in the impedance to get a complete description of the impedance. Therefore, γ has to be written as a function of s and $\boldsymbol{\omega}_{\mathcal{B}/\mathcal{N}}$ first. The equations of motion are linearized around an equilibrium point with arbitrary $\omega_u, \omega_v, \omega_w, \gamma$ and with $\dot{\gamma} = 0$.

$$\begin{gathered}\mathbf{A}_{\text{M}} = \frac{\partial \boldsymbol{M}}{\partial \boldsymbol{x}} \\ \mathbf{A}_{\ddot{\gamma}} = \frac{\partial \ddot{\gamma}}{\partial \boldsymbol{y}}\end{gathered} \tag{2.12}$$

Where $\boldsymbol{x} = [\ddot{\gamma}, \dot{\gamma}, \gamma, \omega_u, \omega_v, \omega_w, \dot{\omega}_s, \dot{\omega}_t, \dot{\omega}_g]^T$ and $\boldsymbol{y} = [\dot{\gamma}, \gamma, \omega_u, \omega_v, \omega_w, \dot{\omega}_s, \dot{\omega}_t, \dot{\omega}_g]^T$. The resulting state space equations are:

$$\begin{gathered}\hat{\boldsymbol{M}} = \mathbf{A}_{\text{M}}(\boldsymbol{x} - \boldsymbol{x}_0) \\ \hat{\ddot{\gamma}} = \mathbf{A}_{\ddot{\gamma}}(\boldsymbol{y} - \boldsymbol{y}_0)\end{gathered} \tag{2.13}$$

Next, Eq. (2.13) is transformed into frequency domain by taking the Laplace transform, $\mathscr{L}\{\hat{\boldsymbol{M}}\}$, and $\mathscr{L}\{\hat{\ddot{\boldsymbol{\gamma}}}\}$. Now we can solve $\mathscr{L}\{\hat{\ddot{\boldsymbol{\gamma}}}\}$ for γ such that $\gamma = f(s, \omega_u, \omega_v, \omega_w)$. The function $f(s, \omega_u, \omega_v, \omega_w)$ can be substituted for γ into $\mathscr{L}\{\hat{\boldsymbol{M}}\}$.

Now that $^{\mathcal{B}}\boldsymbol{M}$ is linearized, transformed into frequency domain, and γ is substituted, it still equals the moments. Hence, $\mathscr{L}\{\hat{\boldsymbol{M}}\} = [M_u, M_v, M_w]^T$. We are only interested in the impedances $\frac{\boldsymbol{M}_i}{\boldsymbol{\omega}_i}$ of the transfer function matrix. So the impedances that are derived are:

$$\begin{bmatrix} \frac{M_u}{\omega_u} & \frac{M_v}{\omega_u} & \frac{M_w}{\omega_u} \\ \frac{M_u}{\omega_v} & \frac{M_v}{\omega_v} & \frac{M_w}{\omega_v} \\ \frac{M_u}{\omega_w} & \frac{M_v}{\omega_w} & \frac{M_w}{\omega_w} \end{bmatrix} \tag{2.14}$$

This leads to the following transfer functions when linearized around $\gamma = \gamma^\star$ and $\omega_u = \omega_u^\star, \omega_v = \omega_v^\star, \omega_w = \omega_w^\star$, which can have arbitrary values. Furthermore only the transfer functions $\frac{M_u}{\omega_u}$, $\frac{M_v}{\omega_v}$ and, $\frac{M_w}{\omega_w}$ are shown. The rest can be found in appendix B. Herein, it is assumed that the gimbal is a sphere, so $I_{\mathrm{gs}} = I_{gt}$. To simplify the equations the following simplification is used.

$$\begin{aligned} J_s &= I_{\mathrm{ws}} + I_{\mathrm{gs}} \\ J_t &= I_{\mathrm{wt}} + I_{gt} \\ J_g &= I_{\mathrm{wt}} + I_{\mathrm{gg}} \end{aligned} \tag{2.15}$$

$$\begin{aligned} \frac{M_u}{\omega_u} &= (\omega_w^\star \sin\left(2\gamma^\star\right)(I_{\mathrm{ws}} - I_{\mathrm{wt}}))/2 - s(J_{\mathrm{s}} + (I_{\mathrm{wt}} - I_{\mathrm{ws}}) \sin\left(\gamma^\star\right)^2) \\ &- \frac{(\omega_w^\star((I_{\mathrm{ws}}-I_{\mathrm{wt}})\omega_u^\star \cos(2\gamma^\star)+(I_{\mathrm{ws}}-I_{\mathrm{wt}})\omega_v^\star \sin(2\gamma^\star)+I_{\mathrm{ws}}\Omega\cos(\gamma^\star))((I_{\mathrm{wt}}-I_{\mathrm{ws}})\omega_v^\star\cos(2\gamma^\star)+(I_{\mathrm{ws}}-I_{\mathrm{wt}})\omega_u^\star\sin(2\gamma^\star)+I_{\mathrm{ws}}\Omega\sin(\gamma^\star)))}{k+bs+J_g s^2+(I_{\mathrm{ws}}-I_{\mathrm{wt}})\omega_u^{\star 2}\cos(2\gamma^\star)+(I_{\mathrm{wt}}-I_{\mathrm{ws}})\omega_v^{\star 2}\cos(2\gamma^\star)+I_{\mathrm{ws}}\Omega\omega_u^\star\cos(\gamma^\star)+I_{\mathrm{ws}}\Omega\omega_v^\star\sin(\gamma^\star)+2(I_{\mathrm{ws}}-I_{\mathrm{wt}})\omega_u^\star\omega_v^\star\sin(2\gamma^\star)} \\ &+ \frac{(s((I_{\mathrm{wt}}-I_{\mathrm{ws}})\omega_v^\star\cos(2\gamma^\star)+(I_{\mathrm{ws}}-I_{\mathrm{wt}})\omega_u^\star\sin(2\gamma^\star)+I_{\mathrm{ws}}\Omega\sin(\gamma^\star))(J_g\omega_v^\star+(I_{\mathrm{ws}}-I_{\mathrm{wt}})\omega_v^\star\cos(2\gamma^\star)+(I_{\mathrm{wt}}-I_{\mathrm{ws}})\omega_u^\star\sin(2\gamma^\star)-I_{\mathrm{ws}}\Omega\sin(\gamma^\star)))}{k+bs+J_g s^2+(I_{\mathrm{ws}}-I_{\mathrm{wt}})\omega_u^{\star 2}\cos(2\gamma^\star)+(I_{\mathrm{wt}}-I_{\mathrm{ws}})\omega_v^{\star 2}\cos(2\gamma^\star)+I_{\mathrm{ws}}\Omega\omega_u^\star\cos(\gamma^\star)+I_{\mathrm{ws}}\Omega\omega_v^\star\sin(\gamma^\star)+2(I_{\mathrm{ws}}-I_{\mathrm{wt}})\omega_u^\star\omega_v^\star\sin(2\gamma^\star)} \end{aligned} \tag{2.16}$$

$$\begin{aligned} \frac{M_v}{\omega_v} &= -(\omega_w^\star \sin\left(2\gamma^\star\right)(I_{\mathrm{ws}} - I_{\mathrm{wt}}))/2 - s(J_t + (I_{\mathrm{ws}} - I_{\mathrm{wt}}) \sin\left(\gamma^\star\right)^2) \\ &+ \frac{\omega_w^\star[(I_{\mathrm{ws}}-I_{\mathrm{wt}})\omega_u^\star\cos(2\gamma^\star)+(I_{\mathrm{ws}}-I_{\mathrm{wt}})\omega_v^\star\sin(2\gamma^\star)+I_{\mathrm{ws}}\Omega\cos(\gamma^\star)][(I_{\mathrm{wt}}-I_{\mathrm{ws}})\omega_v^\star\cos(2\gamma^\star)+(I_{\mathrm{ws}}-I_{\mathrm{wt}})\omega_u^\star\sin(2\gamma^\star)+I_{\mathrm{ws}}\Omega\sin(\gamma^\star)]}{k+bs+J_g s^2+(I_{\mathrm{ws}}-I_{\mathrm{wt}})\omega_u^{\star 2}\cos(2\gamma^\star)+(I_{\mathrm{wt}}-I_{\mathrm{ws}})\omega_v^{\star 2}\cos(2\gamma^\star)+I_{\mathrm{ws}}\Omega\omega_u^\star\cos(\gamma^\star)+I_{\mathrm{ws}}\Omega\omega_v^\star\sin(\gamma^\star)+2(I_{\mathrm{ws}}-I_{\mathrm{wt}})\omega_u^\star\omega_v^\star\sin(2\gamma^\star)} \\ &- \frac{s[(I_{\mathrm{ws}}-I_{\mathrm{wt}})\omega_u^\star\cos(2\gamma^\star)+(I_{\mathrm{ws}}-I_{\mathrm{wt}})\omega_v^\star\sin(2\gamma^\star)+I_{\mathrm{ws}}\Omega\cos(\gamma^\star)][(I_{\mathrm{gs}}-I_{\mathrm{gg}})\omega_u^\star+(I_{\mathrm{ws}}-I_{\mathrm{wt}})\omega_u^\star\cos(\gamma^\star)^2+((I_{\mathrm{ws}}-I_{\mathrm{wt}})\omega_v^\star\sin(2\gamma^\star))/2+I_{\mathrm{ws}}\Omega\cos(\gamma^\star)]}{k+bs+J_g s^2+(I_{\mathrm{ws}}-I_{\mathrm{wt}})\omega_u^{\star 2}\cos(2\gamma^\star)+(I_{\mathrm{wt}}-I_{\mathrm{ws}})\omega_v^{\star 2}\cos(2\gamma^\star)+I_{\mathrm{ws}}\Omega\omega_u^\star\cos(\gamma^\star)+I_{\mathrm{ws}}\Omega\omega_v^\star\sin(\gamma^\star)+2(I_{\mathrm{ws}}-I_{\mathrm{wt}})\omega_u^\star\omega_v^\star\sin(2\gamma^\star)} \end{aligned} \tag{2.17}$$

$$\frac{M_w}{\omega_w} = -\frac{sJ_g(k+bs)}{k+bs+J_g s^2+(I_{\mathrm{ws}}-I_{\mathrm{wt}})\omega_u^{\star 2}\cos(2\gamma^\star)+(I_{\mathrm{wt}}-I_{\mathrm{ws}})\omega_v^{\star 2}\cos(2\gamma^\star)+I_{\mathrm{ws}}\Omega\omega_u^\star\cos(\gamma^\star)+I_{\mathrm{ws}}\Omega\omega_v^\star\sin(\gamma^\star)+2(I_{\mathrm{ws}}-I_{\mathrm{wt}})\omega_u^\star\omega_v^\star\sin(2\gamma^\star)} \tag{2.18}$$

To maximize the moment in $\hat{\boldsymbol{e}}_v$, γ has to be zero. This can be used to simplify the impedances.

$$\begin{aligned} \frac{M_u}{\omega_u} &= \frac{\omega_v^\star\omega_w^\star(I_{\mathrm{ws}}-I_{\mathrm{wt}})(I_{\mathrm{ws}}\Omega+I_{\mathrm{ws}}\omega_u^\star-I_{\mathrm{wt}}\omega_u^\star}{k+bs+J_g s^2+(I_{\mathrm{ws}}-I_{\mathrm{wt}})\omega_u^{\star 2}+(I_{\mathrm{wt}}-I_{\mathrm{ws}})\omega_v^{\star 2}+I_{\mathrm{ws}}\Omega\omega_u^\star} \\ &- \frac{(s\omega_v^{\star 2}(I_{\mathrm{gg}}-I_{\mathrm{gs}})(I_{\mathrm{ws}}-I_{\mathrm{wt}}))}{k+bs+J_g s^2+(I_{\mathrm{ws}}-I_{\mathrm{wt}})\omega_u^{\star 2}+(I_{\mathrm{wt}}-I_{\mathrm{ws}})\omega_v^{\star 2}+I_{\mathrm{ws}}\Omega\omega_u^\star} \end{aligned} \tag{2.19}$$

$$\begin{aligned} \frac{M_v}{\omega_v} &= -sJ_t \\ &- \frac{s(I_{\mathrm{ws}}\Omega+(I_{\mathrm{ws}}-I_{\mathrm{wt}})\omega_u^\star)(I_{\mathrm{ws}}\Omega-I_{\mathrm{gg}}\omega_u^\star J_s\omega_u^\star-I_{\mathrm{wt}}\omega_u^\star}{k+bs+J_g s^2+(I_{\mathrm{ws}}-I_{\mathrm{wt}})\omega_u^{\star 2}+(I_{\mathrm{wt}}-I_{\mathrm{ws}})\omega_v^{\star 2}+I_{\mathrm{ws}}\Omega\omega_u^\star} \\ &- \frac{\omega_v^\star\omega_w^\star(I_{\mathrm{ws}}-I_{\mathrm{wt}})(I_{\mathrm{ws}}\Omega+I_{\mathrm{ws}}\omega_u^\star-I_{\mathrm{wt}}\omega_u^\star}{k+bs+J_g s^2+(I_{\mathrm{ws}}-I_{\mathrm{wt}})\omega_u^{\star 2}+(I_{\mathrm{wt}}-I_{\mathrm{ws}})\omega_v^{\star 2}+I_{\mathrm{ws}}\Omega\omega_u^\star} \end{aligned} \tag{2.20}$$

$$\frac{M_w}{\omega_w} = -\frac{sJ_g(k+bs)}{k+bs+J_g s^2+(I_{\mathrm{ws}}-I_{\mathrm{wt}})\omega_u^{\star 2}+(I_{\mathrm{wt}}-I_{\mathrm{ws}})\omega_v^{\star 2}+I_{\mathrm{ws}}\Omega\omega_u^\star} \tag{2.21}$$

The poles of the simplified impedance are described by:

$$\begin{aligned} p_{1,2} &= \tfrac{-B\pm\sqrt{B^2-4AC}}{2A} \\ A &= J_g \\ B &= b \\ C &= k + I_{\mathrm{ws}}(\Omega\omega_u^\star + \omega_u^{\star 2} - \omega_v^{\star 2}) + I_{\mathrm{wt}}(\omega_v^{\star 2} - \omega_u^{\star 2}) \end{aligned} \tag{2.22}$$

2.3. Equations of motion of a scissored pair CMG

In this section, the equations of motion are derived for a scissored pair CMG (SPCMG). The equations of motion are expressed in the body-fixed frame. The gimbals are coupled such that the angular rotations are always opposite. Therefore, two rotation matrices are needed. The first, ${}^{\mathcal{G}_1}\mathbf{R}_{\mathcal{B}}$ is equal to Eq. (2.2). For the second rotation matrix, ${}^{\mathcal{G}_2}\mathbf{R}_{\mathcal{B}}$, the same rotation matrix is used but $-\gamma$ is substituted for γ. The angular velocities of the second CMG can be seen in Eq. (2.23). A schematic figure of the SPCMG can be seen in Fig. 2.3.

$$ {}^{\mathcal{G}_2}\boldsymbol{\omega}_{\mathcal{W}/\mathcal{G}_2} = \begin{pmatrix} -\Omega \\ 0 \\ 0 \end{pmatrix}, \quad {}^{\mathcal{G}_2}\boldsymbol{\omega}_{\mathcal{G}_2/\mathcal{B}} = \begin{pmatrix} 0 \\ 0 \\ -\dot{\gamma} \end{pmatrix} \tag{2.23} $$

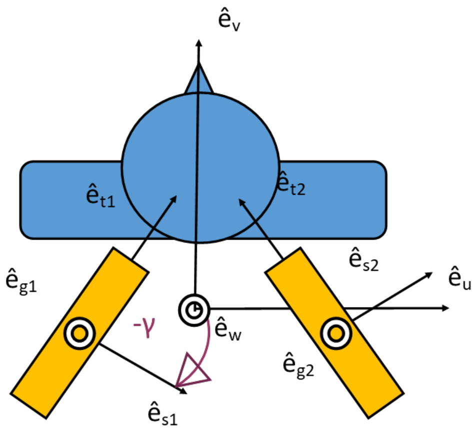


Figure 2.3: Simplistic top view of scissored pair gyroscope. The blue disks rotate in opposite direction. The orange rectangles represent the flywheel

The same method to generate the equations of motion is used for the first CMG as in Section 2.1.2 except that the second gimbal applies a moment, $\boldsymbol{M}_c$, on the first gimbal because they are coupled. So the moment applied to the first gimbal is:

$$ \boldsymbol{M}_1 = \begin{pmatrix} M_{1u} \\ M_{1v} \\ k(\gamma-\gamma_0)+b\dot{\gamma}+M_c \end{pmatrix} \tag{2.24} $$

For the second gyro, the method is very similar to the first. However, the second gimbal rotates in the opposite direction compared to the first gimbal. Therefore, we fill in γ for $-\gamma$, Ω rotates in the $-\hat{\boldsymbol{e}}_s$ direction, and ${}^{\mathcal{G}_2}\mathbf{R}_{\mathcal{B}}$ is used. This also means that the angular velocity is in the opposite direction. The moment due to the coupling also applies to the second gimbal.

$$ \boldsymbol{M}_2 = \begin{pmatrix} M_{2u} \\ M_{2u} \\ -k(\gamma-\gamma_0)-b\dot{\gamma}+M_c \end{pmatrix} \tag{2.25} $$

Now we solve the equation ${}^{\mathcal{B}}(\dot{\boldsymbol{H}}_2)_{\mathcal{N}} = {}^{\mathcal{B}}\,\boldsymbol{M}_2$ for M_c and substitute this result in $\boldsymbol{M}_1$. So, $\boldsymbol{M}_1$ consists of $-I_2\ddot{\gamma} - 2b\dot{\gamma} - 2k\gamma$ among other terms related to the gyroscopic effect. Now the total change of angular momentum can be calculated with:

$$ -{}^{\mathcal{B}}\,(\dot{\boldsymbol{H}}_1)_{\mathcal{N}} - {}^{\mathcal{B}}\,(\dot{\boldsymbol{H}}_2)_{\mathcal{N}} = \boldsymbol{M}_1 \tag{2.26} $$

The written out version of this equation can be seen in Eq. (A.2). When solved for $\ddot{\gamma}$, it results in:

$$ \begin{aligned} \ddot{\gamma} = -[2b\dot{\gamma} - 2(\gamma_0-\gamma)k + I_{\mathrm{gs}}\omega_u^2\sin(2\gamma) - I_{gt}\omega_u^2\sin(2\gamma) - I_{\mathrm{gs}}\omega_v^2\sin(2\gamma) \\ + I_{gt}\omega_v^2\sin(2\gamma) + I_{\mathrm{ws}}\omega_u^2\sin(2\gamma) - I_{\mathrm{wt}}\omega_u^2\sin(2\gamma) - I_{\mathrm{ws}}\omega_v^2\sin(2\gamma) \\ + I_{\mathrm{wt}}\omega_v^2\sin(2\gamma) + 2I_{\mathrm{ws}}\Omega\omega_u\sin(\gamma)/[2\gamma^\star] \end{aligned} \tag{2.27} $$

To check whether the equations of motion are correct, also the Lagrange method was used to generate the equations of motion. The equations of motion found with the Lagrange method were equal to the equations of motion found with the Newton-Euler method. Furthermore, $\boldsymbol{H}_1 + \boldsymbol{H}_2$ was numerically differentiated and this was matched with ${}^{\mathcal{B}}(\dot{\boldsymbol{H}}_1)_{\mathcal{N}} + {}^{\mathcal{B}}\,(\dot{\boldsymbol{H}}_2)_{\mathcal{N}}$. Both validation checks can be seen in appendix A.

2.4. Frequency response analysis of scissored pair CMG

The method of computing the impedance of the SPCMG is exactly the same as for a single CMG from Section 2.2. This leads to the following transfer functions when linearized around $\gamma = \gamma^\star$ and $\omega_u = \omega_u^\star, \omega_v = \omega_v^\star, \omega_w = \omega_w^\star$, which can have arbitrary values. Furthermore only the transfer functions $\frac{M_u}{\omega_u}$, $\frac{M_v}{\omega_v}$ and, $\frac{M_w}{\omega_w}$ are shown. The rest can be found in appendix Appendix B.

$$\begin{aligned}\frac{M_u}{\omega_u} &= -s\cos(\gamma^\star)2J_s \\ &- \frac{s\omega_u^\star \sin(2\gamma^\star)(I_{\text{ws}}-I_{\text{wt}})[2(I_{\text{ws}}-I_{\text{wt}})\omega_v^\star\cos(\gamma^\star)+2I_{\text{gg}}\omega_u^\star\sin(\gamma^\star)+2I_{\text{ws}}\omega_u^\star\sin(\gamma^\star)]}{k+bs+J_g s^2+(I_{\text{ws}}-I_{\text{wt}})\omega_u^{\star 2}\cos(2\gamma^\star)+(I_{\text{wt}}-I_{\text{wt}})\omega_v^{\star 2}\cos(2\gamma^\star)+I_{\text{ws}}\Omega\omega_v^\star\sin(\gamma^\star)} \\ &- \frac{2\omega_u^\star\omega_v^\star\omega_w^\star\sin(2\gamma^\star)\sin(\gamma^\star)(I_{\text{gg}}-I_{\text{gs}})(I_{\text{ws}}-I_{\text{wt}})}{k+bs+J_g s^2+(I_{\text{ws}}-I_{\text{wt}})\omega_u^{\star 2}\cos(2\gamma^\star)+(I_{\text{wt}}-I_{\text{wt}})\omega_v^{\star 2}\cos(2\gamma^\star)+I_{\text{ws}}\Omega\omega_v^\star\sin(\gamma^\star)}\end{aligned} \tag{2.28}$$

$$\begin{aligned}\frac{M_v}{\omega_v} &= -s\cos(\gamma^\star)(2J_t) \\ &- \frac{s[(I_{\text{ws}}-I_{\text{wt}})\omega_v^\star\sin(2\gamma^\star)+I_{\text{ws}}\Omega\cos(\gamma^\star)][2I_{\text{ws}}\Omega+2(I_{\text{wt}}-I_{\text{ws}})\omega_u^\star\cos(\gamma^\star)+2(I_{\text{ws}}-I_{\text{gg}}-2I_{\text{wt}})\omega_v^\star\sin(\gamma^\star)]}{k+bs+J_g s^2+(I_{\text{ws}}-I_{\text{wt}})\omega_u^{\star 2}\cos(2\gamma^\star)+(I_{\text{wt}}-I_{\text{wt}})\omega_v^{\star 2}\cos(2\gamma^\star)+I_{\text{ws}}\Omega\omega_v^\star\sin(\gamma^\star)} \\ &- \frac{2\omega_u^\star\omega_w^\star\sin(\gamma^\star)[I_{\text{ws}}\omega_v^\star\sin(2\gamma^\star)-I_{\text{wt}}\omega_v^\star\sin(2\gamma^\star)+I_{\text{ws}}\Omega\cos(\gamma^\star)](J_g-J_s)}{k+bs+J_g s^2+(I_{\text{ws}}-I_{\text{wt}})\omega_u^{\star 2}\cos(2\gamma^\star)+(I_{\text{wt}}-I_{\text{wt}})\omega_v^{\star 2}\cos(2\gamma^\star)+I_{\text{ws}}\Omega\omega_v^\star\sin(\gamma^\star)}\end{aligned} \tag{2.29}$$

$$\frac{M_w}{\omega_w} = \nexists \tag{2.30}$$

The impedance $\frac{M_w}{\omega_w}$ does not exist because the term ω_w nor $\dot{\omega}_w$ does not appear in the equations of motion found in Eq. (2.26).

If $\gamma = 0$, Eq. (2.28) and Eq. (2.29) simplify to:

$$\frac{M_u}{\omega_u} = -2sJ_s \tag{2.31}$$

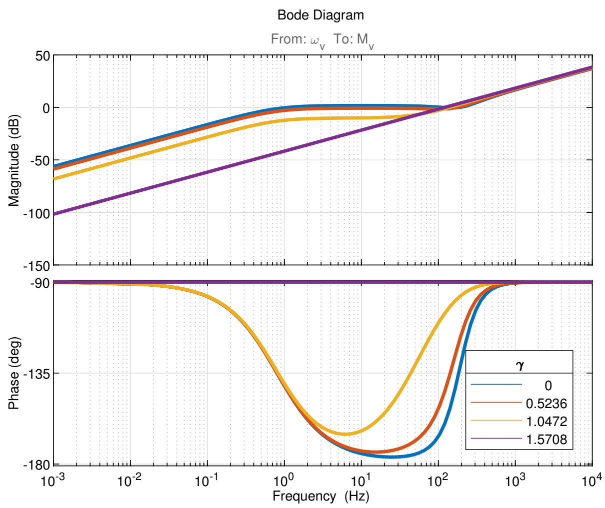
$$\frac{M_v}{\omega_v} = -s(2J_t) - \frac{(2I_{\text{ws}}^2\Omega^2 s)}{(k+bs+J_g s^2+(I_{\text{ws}}-I_{\text{wt}})\omega_u^{\star 2}+(I_{\text{wt}}-I_{\text{ws}})\omega_v^{\star 2})} \tag{2.32}$$

2.5. Effect of changing parameters on frequency response

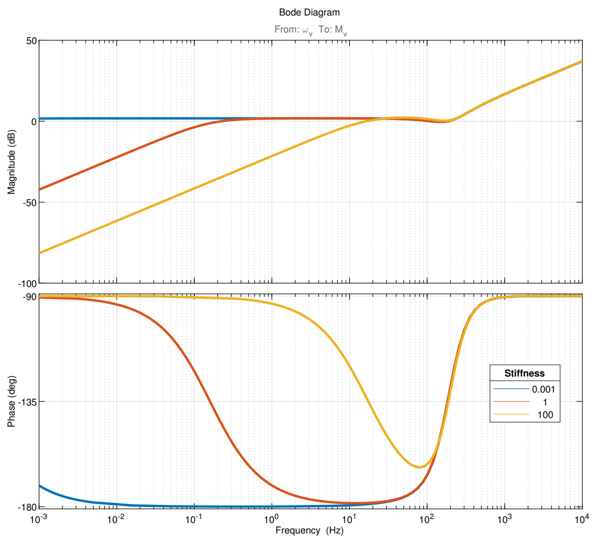
Multiple bode plots with changing parameters are shown in Fig. 2.4 to get an overview of how different parameters change the frequency response of a single CMG. The parameters can be seen in Table 2.1. The chosen parameters are similar to the parameters of mini-GYRO's that are being used in the bio-robotics lab.
In Fig. 2.4a the effect of γ on the frequency response is shown. It shows that the frequency response with γ between 0 rad and $\pi/3$ rad are very similar in both the magnitude and phase. Furthermore, when $\gamma = \pi/2$ rad, the impedance is that of a pure mass.
In Fig. 2.4b the effect of stiffness on the frequency response is shown. It shows that with low stiffness, the system behaves as a damper at low frequencies. With an increasing stiffness, the impedance will become more similar to a mass.
In Fig. 2.4c, the effect of damping on the frequency response is shown. It shows that with a low damper, a complex pole pair and a complex zero pair will exist. The pole pair exists at lower frequencies than the zero pair. With an increase in damping, the impedance will behave as a damper at lower frequencies. It should be noted, however, that the frequency response will depend on specific combinations of parameters. Therefore the frequency response can not be determined with a linear superposition.

Table 2.1: Arbitrary values for the parameters for transfer function $\frac{M_v}{\omega_v}$.

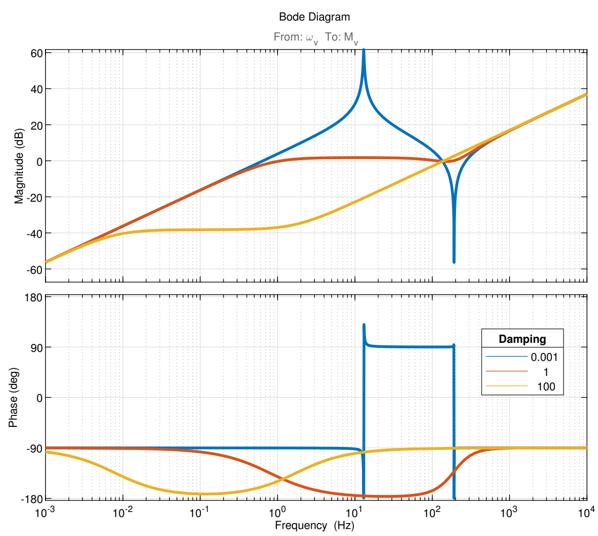
Parameter	Value	unit
k	5	N/m/rad
b	1	Nm/s/rad
I_{ws}	4.4e-04	kgm^2
I_{wt}	2.5e-04	kgm^2
I_{gs}	8.8e-04	kgm^2
I_{gg}	5.0e-04	kgm^2
γ	0.00	rad
Ω	2513	rad/s
ω_u	0	rad/s
ω_v	0	rad/s
ω_w	0	rad s



(a) Effect of changing $\gamma^{\star}$ on the bode plots



(b) Effect of a changing spring stiffness on the frequency response



(c) Effect of a changing damping on the frequency response

Figure 2.4: Bode plots of a single CMG in the body-fixed frame when different parameters are changed.

2.6. Optimization

We want to design an (SP)CMG that produces a specified impedance between the human body and the (SP)CMG. A gradient optimization was used to find a set of parameters for which the (SP)CMG produces this impedance. Gradient optimization is computationally efficient but has a chance to find local minima. Therefor multiple optimizations with different random initial guesses were performed. Knowledge about the system was used to determine the initial guess. Then some randomness was added to the initial guess to reduce the change of finding a local minimum even further. The algorithm minimizes the difference between the desired transfer function (TFdes) and the obtained transfer function (TF). Both the magnitude and phase are important. If the magnitude of the two transfer functions is the same, the pole and zero location of the transfer function are the same. However, this only holds when all poles and zeros are in the left half-plane. If there exists one zero or pole in the right half-plane, the phase shifts by 180°, this is called non-minimum phase. Therefore, the phase is considered more valuable. Furthermore, if the phase between the TFdes and TF differs 180°, the moment will be applied in the opposite direction than intended, which is worse than a moment with a different magnitude in the right direction. The algorithm used to solve the optimal parameter problem is as follows:

$i \leftarrow 100$
$x = u_b . \cdot R \sim U([0,1])$
$min_x ||C(x)||_2^2$

Where the cost function is:

$$C = w_1(imag(TF_{\text{des}} - imag(TF))) + real(TF_{\text{des}} - TF) \quad (2.33)$$

Other cost functions are discussed in Section 5.6. The optimization was performed with the MATLAB R2019b (MathWorks; Natick, USA) function, *lsqnonlin*. The algorithm minimizes the difference between the desired transfer function, TF_{des}, and the transfer function that was computed earlier, TF. A w_1 of 100 was chosen. This was because the phase was considered more important than the magnitude. The frequency vector consists of two hundred logarithmic spaced frequencies. These frequencies range from 0.1 Hz to 10 Hz. Two sets of optimizations were performed. One in which was examined how good the fit can theoretically get, and one with realistic bounds on the parameters. The parameters that were optimized are spring stiffness, damping, moments of inertia of the flywheel, moments of inertia of the gimbal, and the orientation of the flywheel. The angular momentum depends on both the moment of inertia and the angular velocity of the flywheel. Hence, there is redundancy between those parameters. Therefore, the angular velocity of the flywheel was fixed on 1500 rad/s for both types of optimizations. An angular velocity of 0 rad/s was used for all angular velocities of the human body. The bounds for the optimizations can be seen in Table 2.2. The bound on the inertia of the flywheel was based on the inertia of the flywheel of Lemus et al. [24], where the inertia $I_{\text{ws}} = 0.02\,\text{kg/m}^2$. Twice this value was used to give the optimization more space to explore. Since the gimbal does not provide gyroscopic torque, it has to be lightweight to reduce the mass of the overall system. Therefore, an upper bound of $0.2\,\text{kg/m}^2$ was chosen. The spring stiffness was based on the maximum spring stiffness of a torsion spring that was found in [14]. The damping coefficient was based on the rotary dampers found in [26]. The optimization was performed 100 times to increase the change of finding a global minimum.

Table 2.2: The lower and upper bounds for the for the variable parameters for the (SP)CMG

Parameter	Lower Bound	Upper Bound on Random Guess	Upper Bound	Unit
k	0	4500	4500	Nm/rad
b	0	3800	3800	Nm/s/rad
I_{ws}	0	0.3	0.04	kgm^2
I_{wt}	0	0.3	0.04	kgm^2
I_{gs}	0	0.3	0.02	kgm^2
I_{gg}	0	0.3	0.02	kgm^2
γ	$-\pi$	π	π	rad
Ω	1500	1500	1500	rad/s
ω_u	0	0	0	rad/s
ω_v	0	0	0	rad/s
ω_w	0	0	0	rad/s

3
Case Study

In the previous chapter, the impedance of the CMG and SPCMG were derived. Furthermore, the optimization algortim was explained. This chapter wil explain to which impedances the (SP)CMG will be matched.

3.1. Desired transfer function

To investigate whether it is possible to match impedances, the impedance of the (SP)CMGs were optimized for multiple impedances. The optimization was done for a spring, damper, mass, a mass-spring-damper system and a PD controller inspired by the XCoM, a measure of stability. The values for these systems were arbitrarily chosen. The desired transfer functions can be seen in Table 3.1.

Table 3.1: Table that shows the desired transfer functions that were used for the optimization.

Mechanism	Spring	Damper	Mass	Mass-Spring-Damper System	PDXCoM
Symbolic Transfer Function	$-\frac{k}{s}$	$-\frac{bs}{s}$	$-\frac{Js^2}{s}$	$-\frac{Js^2+bs+k}{s}$	$+\frac{k_\mathrm{p}+k_\mathrm{d}s}{s}$
Transfer Function	$-\frac{30}{s}$	$-\frac{5s}{s}$	$-\frac{0.5s^2}{s}$	$-\frac{0.5s^2+5s+30}{s}$	$+\frac{100+32s}{s}$

3.1.1. XCoM

The desired transfer function is modeled after a measure of dynamic stability, XCoM [15]. The assumptions are that the human body can be modeled as an inverted pendulum, see Fig. 3.1. Furthermore, there are no ankle moments applied and the moment of inertia of the human body is approximated as a point mass. The sum of the moments around the ankle is:

$$\sum \boldsymbol{M}: \quad J_\mathrm{f}\ddot{\theta} = mgL\sin\theta \tag{3.1}$$

Where m is the mass of the upper body, L is the length of the leg, θ is the angle of the leg with respect to the vertical, $J_\mathrm{f} = J_\mathrm{c} + mL^2$, and g is the gravitational constant. To get the transfer function, this will be linearized about $\theta = 0$.

$$J_\mathrm{f}\ddot{\theta} \approx mgL\theta \tag{3.2}$$

The natural frequency, $\omega_0 = \sqrt{\frac{mgl}{J_\mathrm{f}}}$ and the moment of inertia of the trunk is set to zero. When we substitude this in Eq. (3.2) we get:

$$\ddot{\theta} - \omega_0^2\theta = 0 \tag{3.3}$$

For the orbital energy, we have to multiply equation 3.3 by $\frac{1}{4}\dot{\theta}$ and integrate over time [19].

$$\begin{aligned} E_\mathrm{orb} &= \tfrac{1}{4}\int \dot{\theta}(\ddot{\theta} - \omega_0^2)\mathrm{d}t \\ E_\mathrm{orb} &= \tfrac{1}{2}(\dot{\theta} - \omega_0\theta)(\dot{\theta} + \omega_0\theta) \end{aligned} \tag{3.4}$$

XCoM is then defined as the distance from the stable trajectory. The stable trajectory can be seen in the phase plot of Fig. 3.2. The external moment that should be applied to make the system stable is:

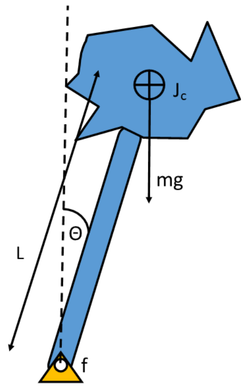


Figure 3.1: Figure of XCoM. The length of the leg is depicted by L. The angle of leg with respect to the vertical is depicted by θ. The moment of inertia of the body is depicted by J_c. The gravity force is depicted by mg.

$$\begin{aligned} M &= -kXCoM \\ M &= -kl(\theta + \omega_0^{-1}\dot{\theta}) \\ M &= -k_p l\theta - k_d \cdot l\omega_0^{-1}\dot{\theta} \end{aligned} \tag{3.5}$$

However, this is the moment generated by the CMG, so the moment applied on the human is in the opposite direction.

$$M = k_p l\theta + k_d \cdot l\omega_0^{-1}\dot{\theta} \tag{3.6}$$

If we can choose k_p and k_d independently, equation 3.5 can be interpreted as a PD controller. To make the system equivalent to the equations of the gyro, the equations are put in frequency domain and the variables are renamed.

$$\begin{aligned} \dot{\theta} &= \omega \\ \theta &= \frac{\omega}{s} \end{aligned} \tag{3.7}$$

For the gains, arbitrary values are used, $k_p = 100$ and $k_d = 32$. For the leg length we choose l = 1 m. From this, the desired transfer function is:

$$TF_{des} = \frac{100 + 32s}{s} \tag{3.8}$$

Since keeping balance around the sagittal axis is the most difficult for humans [35], it is decided that the transfer function that will be optimized for is $\frac{M_v}{\omega_v}$.

3.2. Relevant frequencies

In human balance control, it is common to use a cut-off frequency of about 10 Hz [4, 9]. This is because human typically can track frequencies up to 6 Hz [27]. Therefore, the optimization will be performed for a frequency range of 0.01 Hz to 10 Hz, which equals to 0.02πrad/s to 20π rad/s

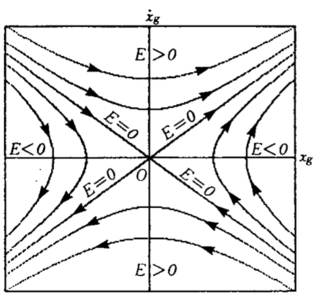


Figure 3.2: Phase plot of the orbital energy. The lines converging to the origin are the stable trajectories. Figure from Kajita et al. [19]. With permission.

3.3. Walking simulation

A feed-forward simulation of the (SP)CMG was made using human gait data. This means that the human gait data does not respond to the moments exerted by the (SP)CMG. The angular velocity and angular acceleration of the trunk were used. Furthermore, the orientation of the trunk with respect to the lab was used to determine the angular velocities and angular acceleration in the body-fixed frame. Two different walking speeds of the same subject were used as gait data. The walking speeds are 0-0.4 m/s, and a self-selected fast speed. The gait data that was used is from the data set of [39]. The angular velocities of the gait data can be seen in Fig. 3.3.

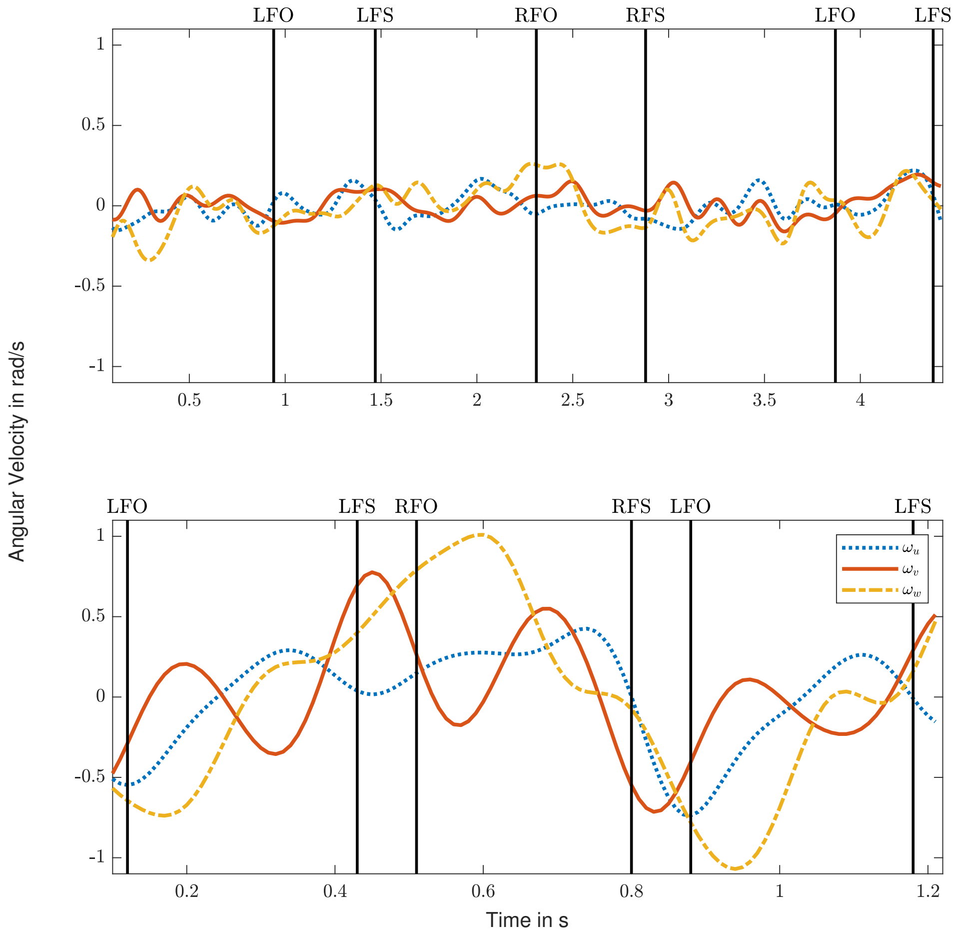


Figure 3.3: Angular velocity of a subject with a walking speed between 0-0.4 m/s for the top graph, and a walking speed between 1.9-2.2 m/s for the bottom graph. LFO = left foot off, LFS = lef foot strike, RFO = right foot strike, RFS = right foot strike.

4

Results

This chapter will show the results of the optimization. The bodeplots show the impedance of the (SP)CMG with the poles and zeros and the desired transferfunction. The parameters that were found with the optimizations are shown in a table. Furthermore, the walking simulation is shown when the (SP)CMG was optimized to simulate a damper.

4.1. Results of a single CMG

In the following sections, only the bode plots of the optimized impedance without bounds is shown. The bode plots of the impedance when the optimization was performed with realistic bounds can be seen in Appendix C. The parameters of both sets of optimizations are shown in this section. Furthermore, the time response of a CMG when optimized to simulate a damper with the realistic parameters is shown. The other time responses are shown in Appendix D.

4.1.1. Optimization of spring

The optimization was performed one-hundred times. The upper bounds of the initial guess were changed for the spring stiffness and the damping to 0.01 Nm/rad and 0.001 Nm/rad/s respectively. The squared norm of the residual (resnorm) of the best optimization was 2.1. The optimized parameters can be seen in table 4.1. Figure 4.1 shown the bode plots of both a spring (red dotted) and of the optimized impedance of a single CMG, $\frac{M_v}{\omega_v}$. The resulting impedance function has two poles located at $p_{1,2} = -2.6 \times 10^{-5} \pm 1.6 \times 10^{-4} i$ and three zeros located at $z_1 = 0$, and $z_{2,3} = -2.6 \times 10^{-5} \pm 7.01 \times 10^{2} i$. The damping in the system is $\zeta = 0.16$ and the natural frequency $\omega_n = 0.16 \times 10^{-3}$ rad/s.

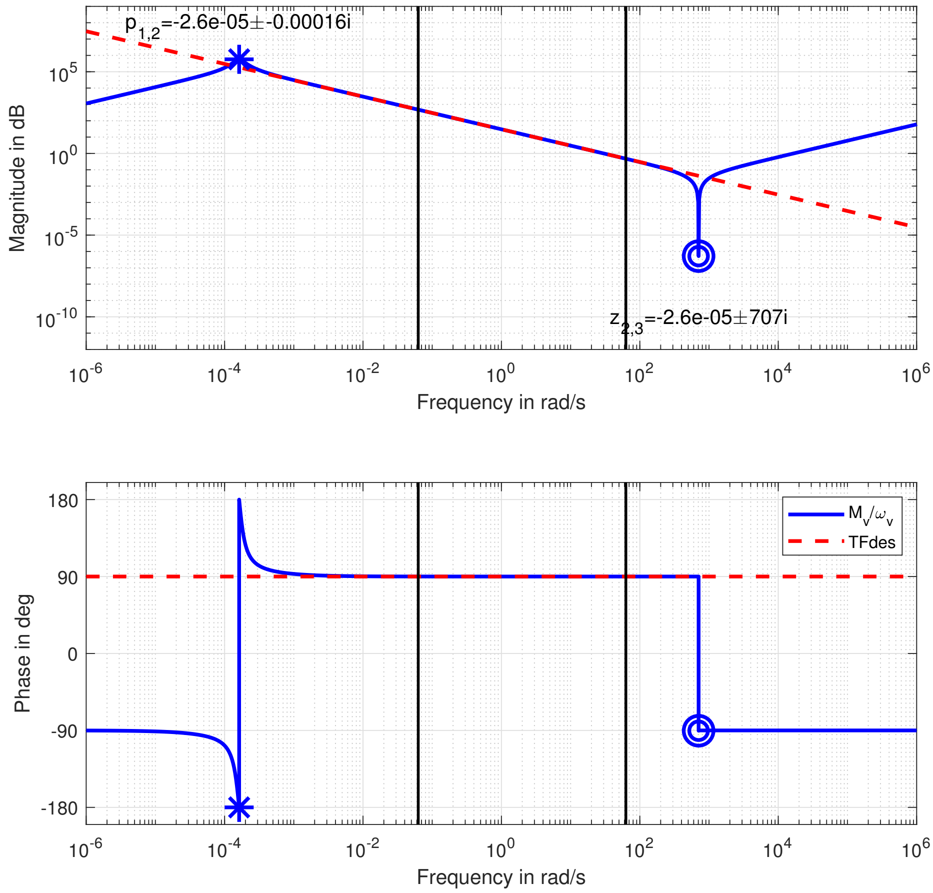


Figure 4.1: Bode plot of both TFdes, a spring, (red) and the optimized impedance (blue). The area between the vertical lines are the optimized frequencies. Zeros are indicated with a circle, poles are indicated with a cross.

4.1.2. Optimization to imitate a damper

The optimization was done one-hundred times for $\frac{M_v}{\omega_v}$. The upper bound of the initial guess for the spring was changed to 0.1 Nm/rad. The resnorm of the best optimization of the cost function was 1.6×10^{-5}. The optimized parameters are shown in Table 4.1. Figure 4.3 shows both the transfer function of a damper and $\frac{M_v}{\omega_v}$. The resulting transfer function has two poles and three zeros. The poles are located at $p_1 = -2812.5$, and $p_2 = -7.97\times10^{15}$. The zeros are located at $z_1 = 0$,and $z_{2,3} = -1406.3\pm2435.9i$. The damping in the system is $\zeta = 1$ and the natural frequency $\omega_n = 2812.5$ rad/s, and $\omega_n = 7.97\times10^{15}$ rad/s.

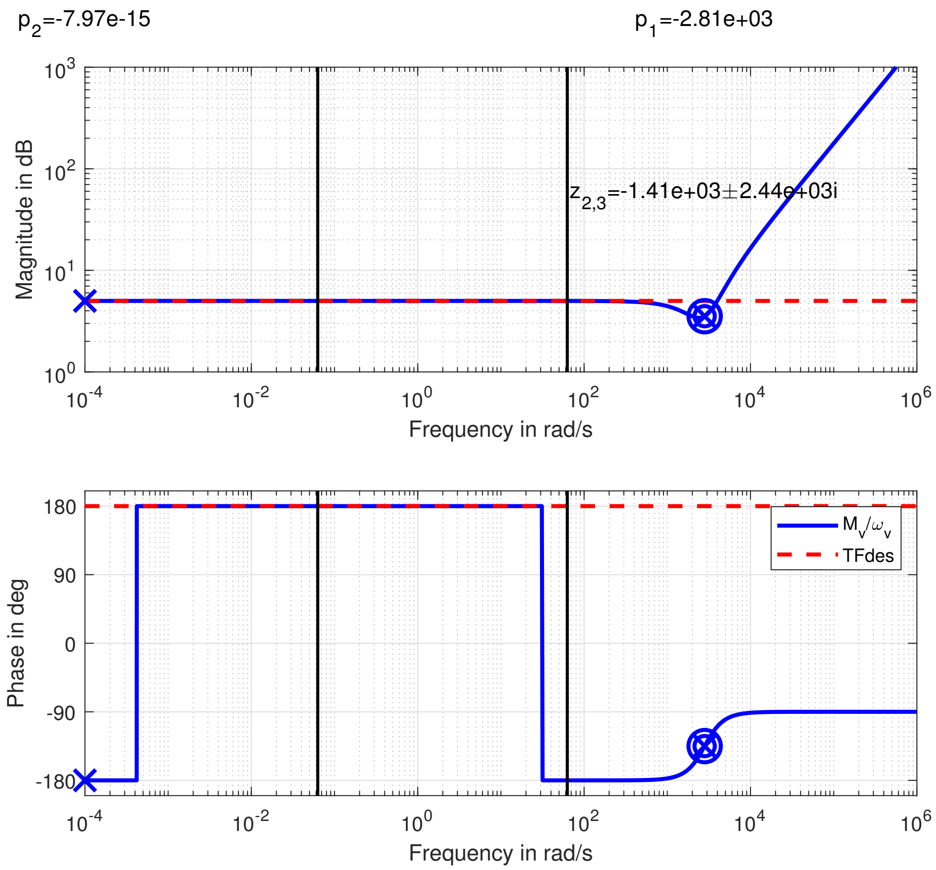


Figure 4.2: Bode plot of both TFdes, a damper, (red dotted) and the optimized impedance (blue). The area between the vertical lines are the optimized frequencies. Zeros are indicated with a circle, poles are indicated with a cross.

4.1.3. Optimization to imitate a mass

The optimization was done one-hundred times for $\frac{M_v}{\omega_v}$. The bounds on the initial guess were not changed for the optimizations. The resnorm of the best optimization of the cost function was 1.88×10^{-12}. The optimized parameters are shown in Table 4.1. Figure 4.3 shows both the transfer function of a mass and $\frac{M_v}{\omega_v}$. The resulting transfer function has two poles and three zeros. The poles are located at $p_1 = -2.27 \times 10^5$, and $p_2 = -2.15$. The zeros are located at $z_1 = 0$, $z_2 = -2.27 \times 10^5$ and $z_3 = -2.15$. The damping in the system is $\zeta = 1$ and the natural frequency $\omega_n = 2.27 \times 10^5$ rad/s and $\omega_n = 2.15$ rad/s.

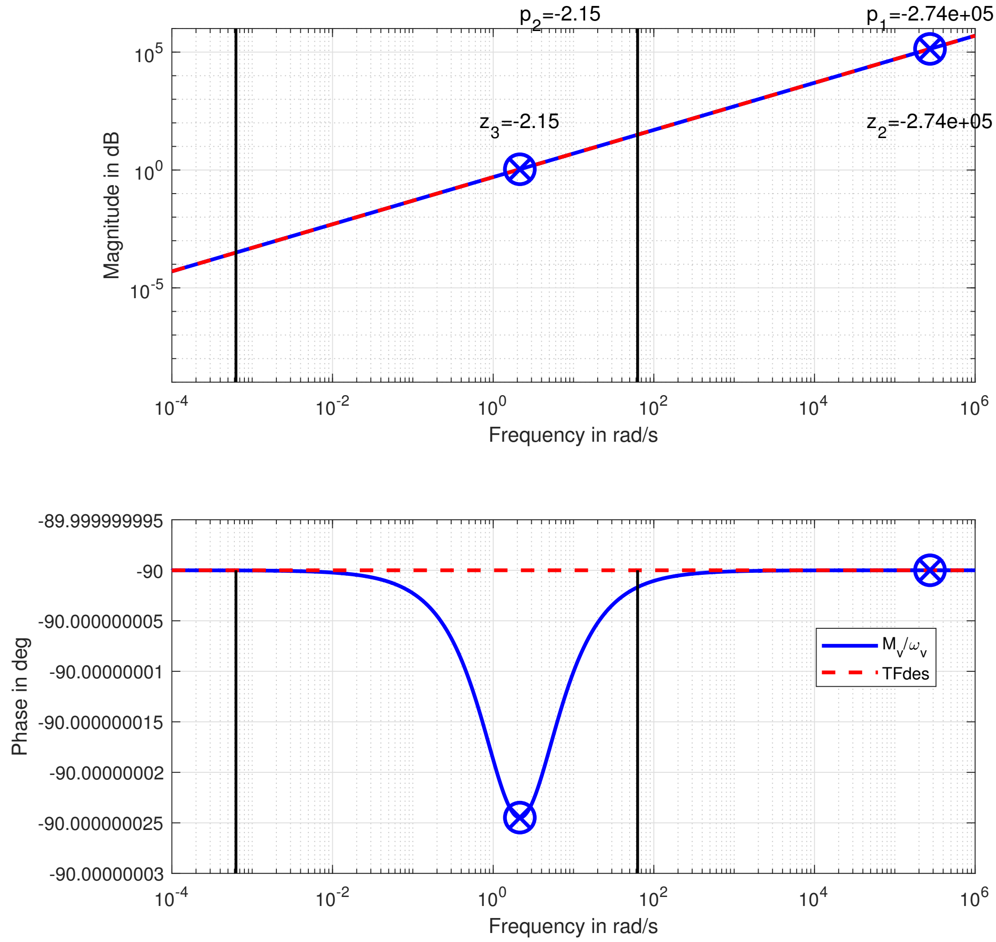


Figure 4.3: Bode plot of both TFdes, a mass, (red) and the optimized impedance (blue). The area between the vertical lines are the optimized frequencies. Zeros are indicated with a circle, poles are indicated with a cross.

4.1.4. Optimization to imitate a mass-spring-damper System

The optimization was done one-hundred times for $\frac{M_v}{\omega_v}$. The resnorm of the best optimization of the cost function was 1.7×10^3. The upper bound of the initial guess for the spring was changed to 0.1 Nm/rad. The optimized parameters are shown in Table 4.1. Figure 4.4 shows both the transfer function of a mass-spring-damper system and $\frac{M_v}{\omega_v}$. The resulting transfer function has two poles and three zeros. The poles are located at $p_{1,2} = -0.00 \pm 0.0012i$. The zeros are located at $z_1 = 0$,and $z_{2,3} = -0.00 \pm 7.73i$. The damping in the system is $\zeta = 0.18 \times 10^{-7}$ and the natural frequency $\omega_n = 0.0012\,\text{rad/s}$.

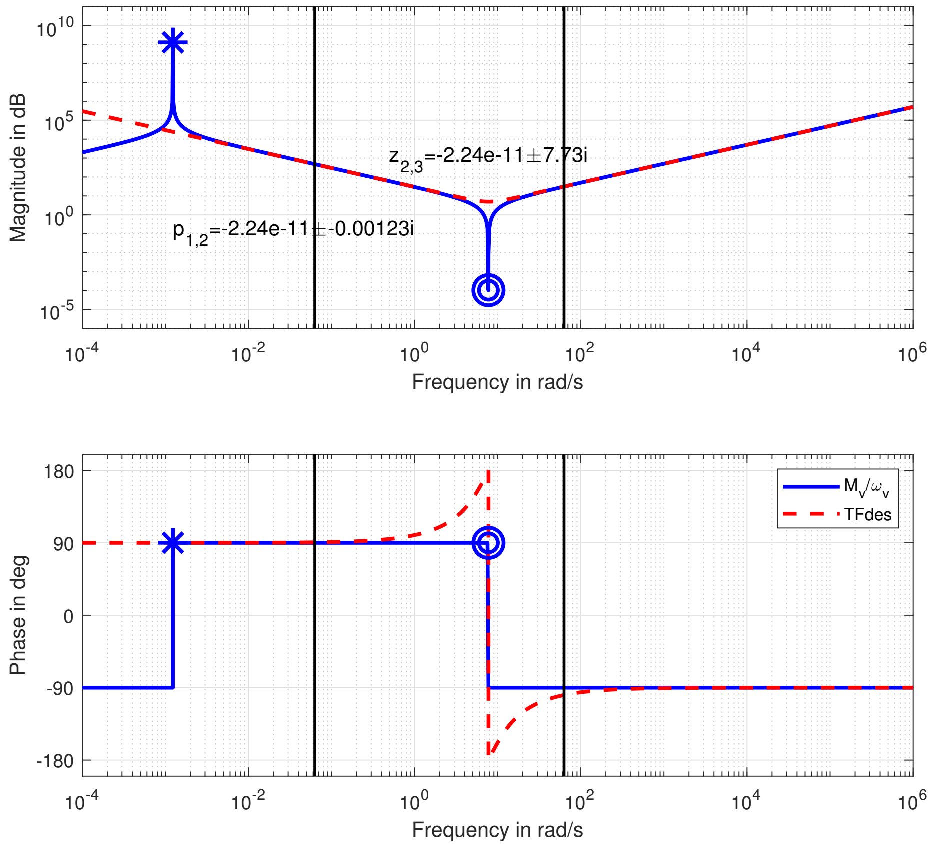


Figure 4.4: Bode plot of both TFdes, a mass, (red) and the optimized impedance (blue). The area between the vertical lines are the optimized frequencies. Zeros are indicated with a circle, poles are indicated with a cross.

4.1.5. Optimization of PDXCoM

The optimization was done one-hundred times for $\frac{M_v}{\omega_v}$. The resnorm of the best optimization of the cost function was 2.6×10^9. The optimized parameters are shown in Table 4.1. Figure 4.5 shows both the transfer function of the PD controller and $\frac{M_v}{\omega_v}$. The resulting transfer function has two poles located at $p_1 = -176.0$, and $p_2 = -0.022$ and three zeros located at $z_1 = 0$, and $z_{2,3} = -88.0 \pm 164.9i$. The damping in the system is $\zeta = 1$ and the natural frequency $\omega_n = 0.022\,\text{rad/s}$, and $\omega_n = 176.0\,\text{rad/s}$.

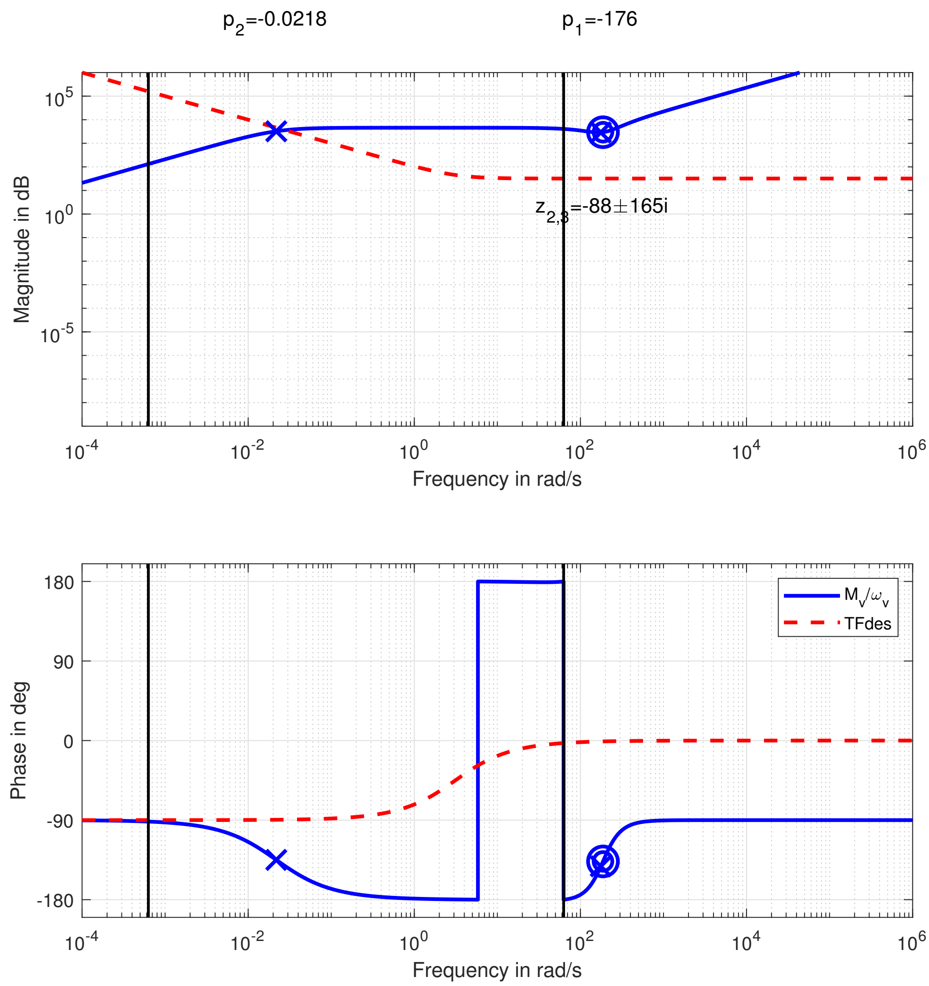


Figure 4.5: Bode plot of both TFdes, PDXCoM, (red dotted) and the optimized impedance (blue). The area between the vertical lines are the optimized frequencies. Zeros are indicated with a circle, poles are indicated with a cross.

Table 4.1: The optimized parameters of the CMG. Both the best possible parameters and the realistic parameters (RP) are shown.

Parameter	**Spring**	**Damper**	**Mass**	**Mass-Spring-Damper**	**PDXCoM**	**Unit**
Resnorm	2.1	1.6×10^{-5}	1.88×10^{-12}	1.7×10^{3}	2.63×10^{9}	
Resnorm RP	112.1	2.2×10^{-5}	0.84	6.3×10^{8}	1.1×10^{11}	
k	2.7×10^{-12}	4.2×10^{-14}	5888.6	1.5×10^{-6}	169.0	Nm/rad
k RP	7.5×10^{-8}	4.0×10^{-14}	3633.3	3.1×10^{-5}	0.23	Nm/rad
b	5.3×10^{-9}	5.21	2.74×10^{5}	4.5×10^{-11}	7756.7	Nm/rad/s
b RP	6.9×10^{-6}	5.36	7.31	9.2×10^{-4}	2.76	Nm/rad/s
I_{ws}	3.69×10^{-5}	0.0034	1.51×10^{-4}	0.0037	3.99	kgm^2
I_{ws}RP	1.0×10^{-4}	0.0035	0.028	5.6×10^{-4}	0.040	kgm^2
I_{wt}	1.85×10^{-5}	0.0017	7.55×10^{-5}	0.0018	1.99	kgm^2
I_{wt}RP	0.5×10^{-5}	0.0017	0.014	2.8×10^{-4}	0.02	kgm^2
I_{gs}	4.14×10^{-5}	7.6×10^{-5}	0.50	0.50	21.04	kgm^2
I_{gs}RP	3.17×10^{-4}	1.3×10^{-4}	6.5×10^{-6}	0.01	1.2×10^{-14}	kgm^2
I_{gg}	8.28×10^{-5}	1.5×10^{-4}	1.00	1.00	42.07	kgm^2
I_{gg}RP	6.34×10^{-4}	2.7×10^{-4}	1.3×10^{-5}	0.02	2.4×10^{-14}	kgm^2
$\gamma^\star$	0.01	7.1×10^{-5}	1.50	0.092	3.05	rad
$\gamma^\star$RP	0.30	3.86×10^{-4}	0.001	0.10	-2.5×10^{-4}	rad

4.1.6. Walk simulation

The moment that were applied on the human by the CMG are shown in this subsection. The parameters used for the CMG are the parameters that were found when the CMG was optimized a damper. The walking simulations with the other parameters can be seen in Appendix D. In Fig. 4.6 it can be seen that the maximum moment of 1.99 N m is applied before the first left foot off. Furthermore, the angle γ stays between 0.05 rad and -0.03 rad.

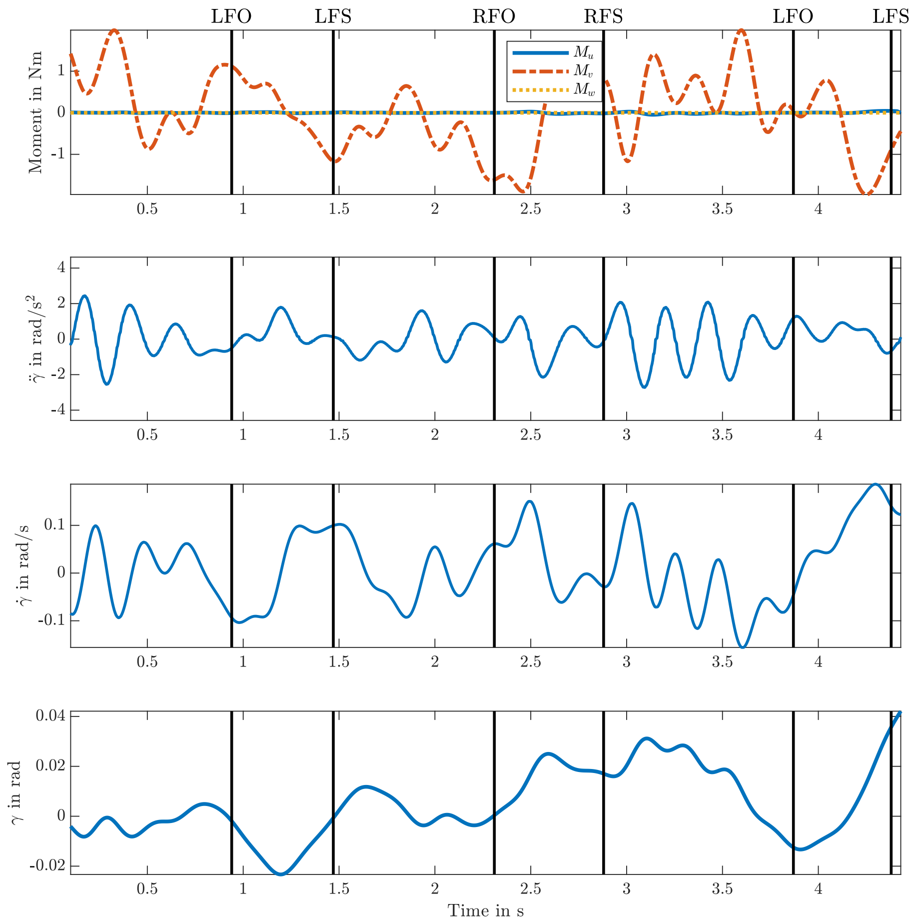


Figure 4.6: Forward simulation of the moments exerted on the human by the optimized CMG. Also $\ddot{\gamma}$, $\dot{\gamma}$, and γ are shown. The walking speed was between 0-0.4 m/s. LFO = left foot off, LFS = left foot strike, RFO = right foot strike, RFS = right foot strike.

In Fig. 4.7 it can be seen that the maximum moment of -6.66 N m is applied between the left foot strike and the right foot off. Furthermore, the angle γ stays between -0.07 rad and 0.05 rad.

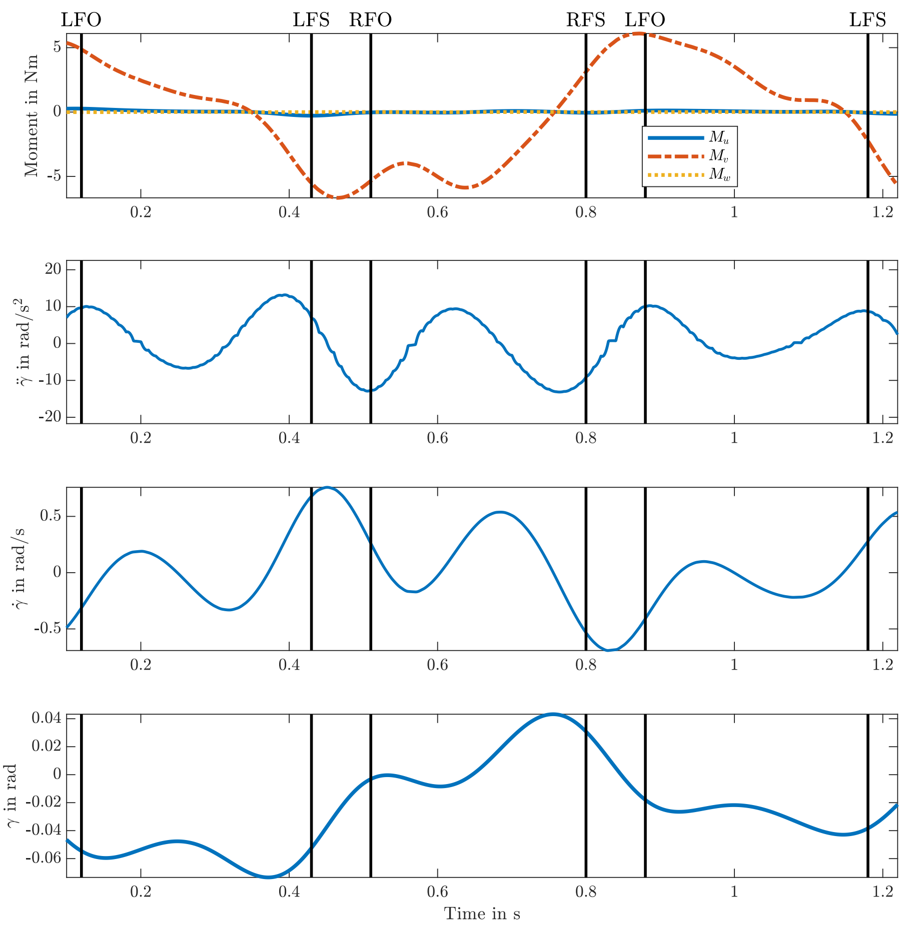


Figure 4.7: Forward simulation of the moments exerted on the human by the optimized CMG. Also $\ddot{\gamma}$, $\dot{\gamma}$, and γ are shown. The walking speed was a self selected fast speed which was between 1.9-2.2 m/s. LFO = left foot off, LFS = left foot strike, RFO = right foot strike, RFS = right foot strike.

4.2. Scissored pair CMG

For the single SPCMG, one-hundred optimizations were performed for each target. In the following sections, only the bode plots of the optimized impedance without bounds is shown. The bode plots of the impedance when there was optimized bounds, is shown in Appendix C. The parameters of both sets of optimizations are shown in this section. Furthermore, the time response of the SPCMG is shown with the found optimized parameters when the SPCMG was optimized to be simulate PDXCoM with realistic values.

4.2.1. Optimization to imitate a spring

The resnorm of the best optimization was 2.4. The upper bounds of the initial guess were changed for the spring stiffness and the damping to 0.01 Nm/rad and 0.001 Nm/rad/s respectively. The optimized parameters can be seen in table 4.2. The bode plot of the found impedance function can be seen in Fig. 4.8 together with the desired transfer function. The resulting impedance function has two poles located at $p_{1,2} = -0.027 \pm 0.16 \times 10^{-3} i$ and three zeros located at $z_1 = 0$, $z_{2,3} = -2.6 \times 10^{-5} \pm 682.7i$. The damping in the system is $\zeta = 0.16$ and the natural frequency $\omega_n = 0.16 \times 10^{-3}$ rad/s.

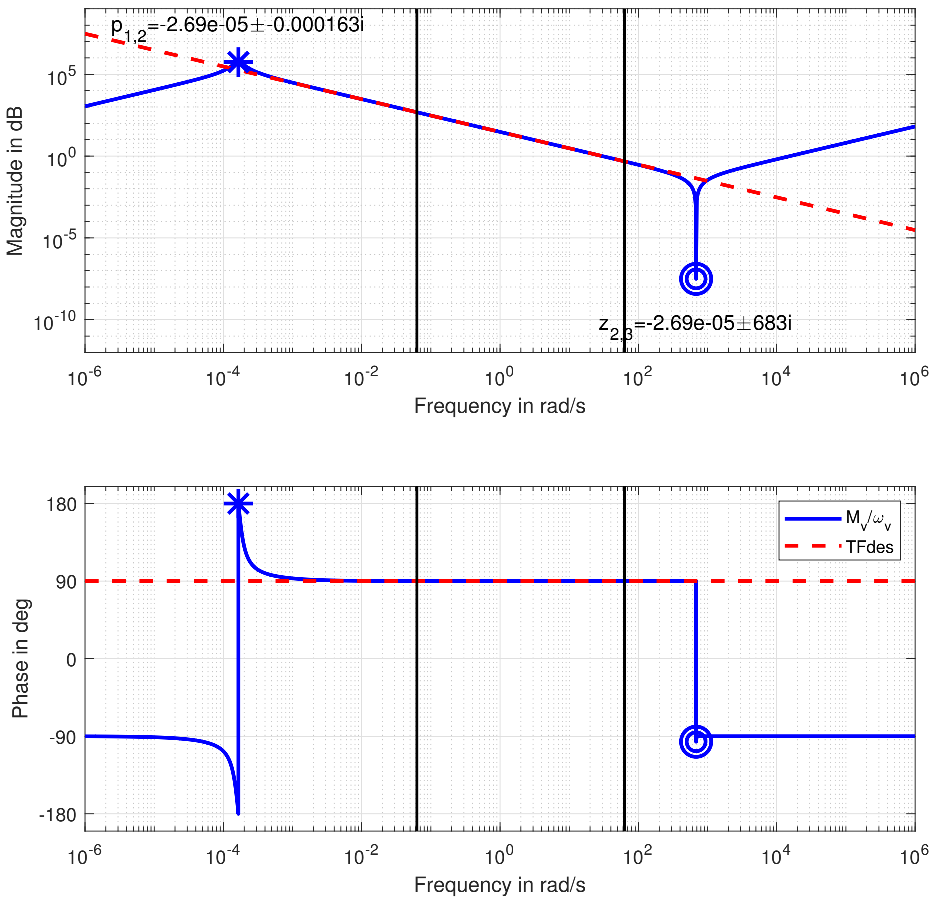


Figure 4.8: Impedance of a SPCMG when optimized to mimic a spring. The area between the vertical lines are the optimized frequencies. Zeros are indicated with a circle, poles are indicated with a cross.

4.2.2. Optimization to imitate a damper

The resnorm of the best optimization was 4.4×10^{-5}. The optimized parameters can be seen in table 4.2. The bode plot of the found impedance function can be seen in Fig. 4.9 together with the desired transfer function. The found impedance function from has two poles located at $p_1 = -2428.3$, $p_2 = -1.12e - 14$ and three zeros located at $z_1 = 0$, and $z_{2,3} = -1214.1 \pm 2103.2i$. The damping in the system is $\zeta = 1$ and the natural frequency

$\omega_n = 2428.3\,\text{rad/s}$ and $\omega_n = 1.12 \times 10^{-14}\,\text{rad/s}$.

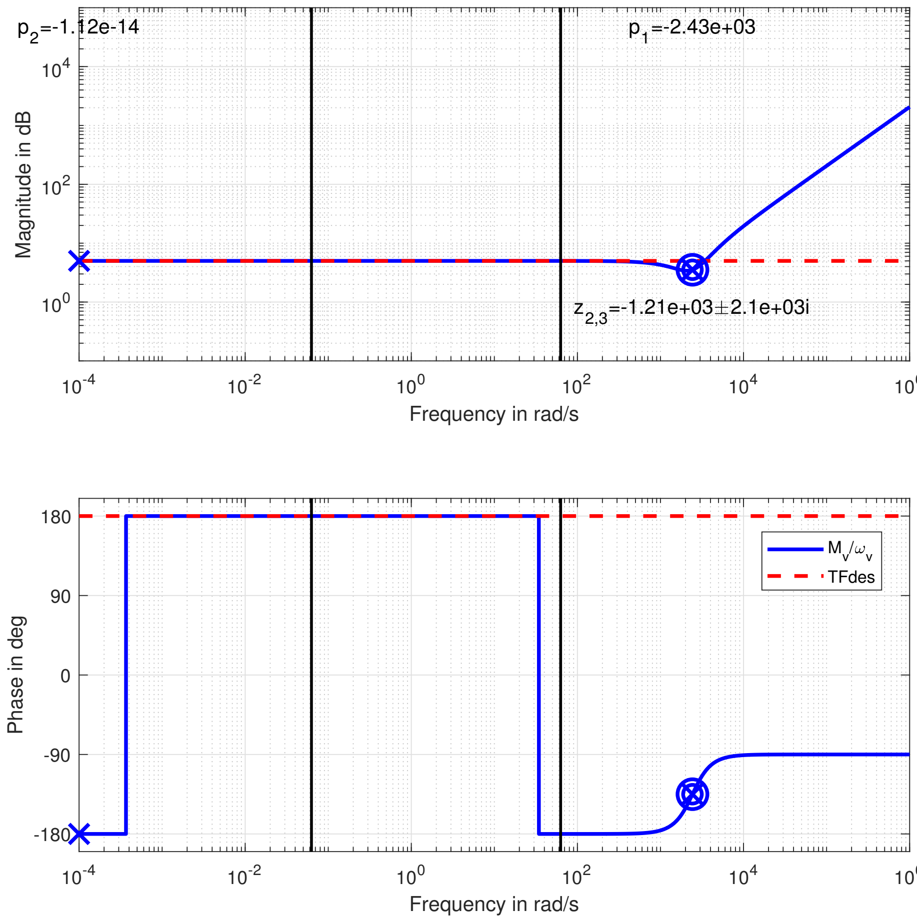


Figure 4.9: Impedance of a SPCMG when optimized to mimic a damper. The area between the vertical lines are the optimized frequencies. Zeros are indicated with a circle, poles are indicated with a cross.

4.2.3. Optimization to imitate a mass

The resnorm of the best optimization was 1.5×10^{-9}. The optimized parameters can be seen in table 4.2. The bode plot of the found impedance function can be seen in Fig. 4.10 together with the desired transfer function. The found impedance function has two poles located at $p_1 = -5615.8$, $p_2 = -1.1$ and three zeros located at $z_1 = 0$, $z_2 = -5615.8$, and $z_3 = -1.1$. The damping in the system is $\zeta = 1$ and the natural frequency $\omega_n = 5615.8$ and $\omega_n = 1.1\,\text{rad/s}$.

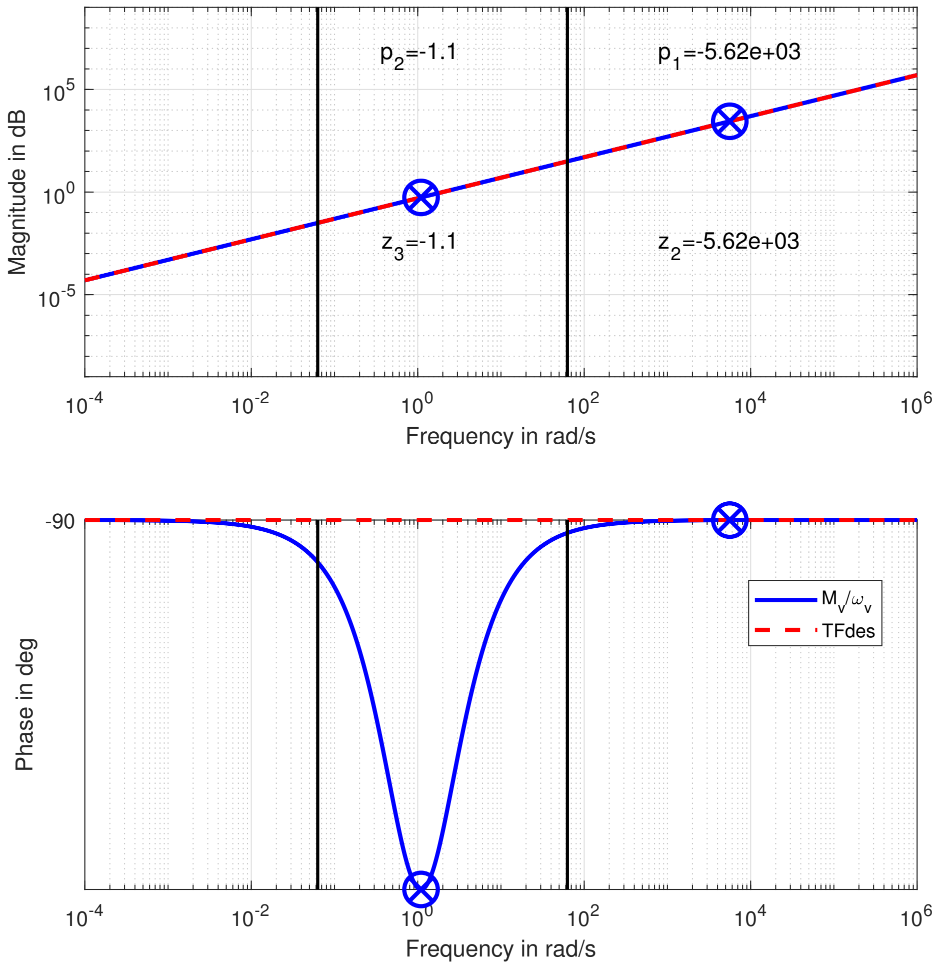


Figure 4.10: Impedance of a SPCMG when optimized to mimic a mass. The area between the vertical lines are the optimized frequencies. Zeros are indicated with a circle, poles are indicated with a cross.

4.2.4. Optimization to imitate a mass-spring-damper system

The resnorm of the best optimization was 1.7×10^3. The resnorm of the best optimization of the cost function was 1.7×10^3. The upper bound of the initial guess for the spring was changed to 0.1 Nm/rad. The optimized parameters can be seen in table 4.2. The bode plot of the found impedance function can be seen in Fig. 4.10 together with the desired transfer function. Two zeros and one pole are not shown because they exist at very high frequencies. The found impedance function has two poles located at $p_{1,2} = -0.000 \pm 0.0012i$ and three zeros located at $z_1 = 0$, and $z_{2,3} = -0.000 \pm 7.7i$. The damping in the system is $\zeta = 0.32 \times 10^{-7}$ and the natural frequency $\omega_n = 0.0012\,\text{rad/s}$.

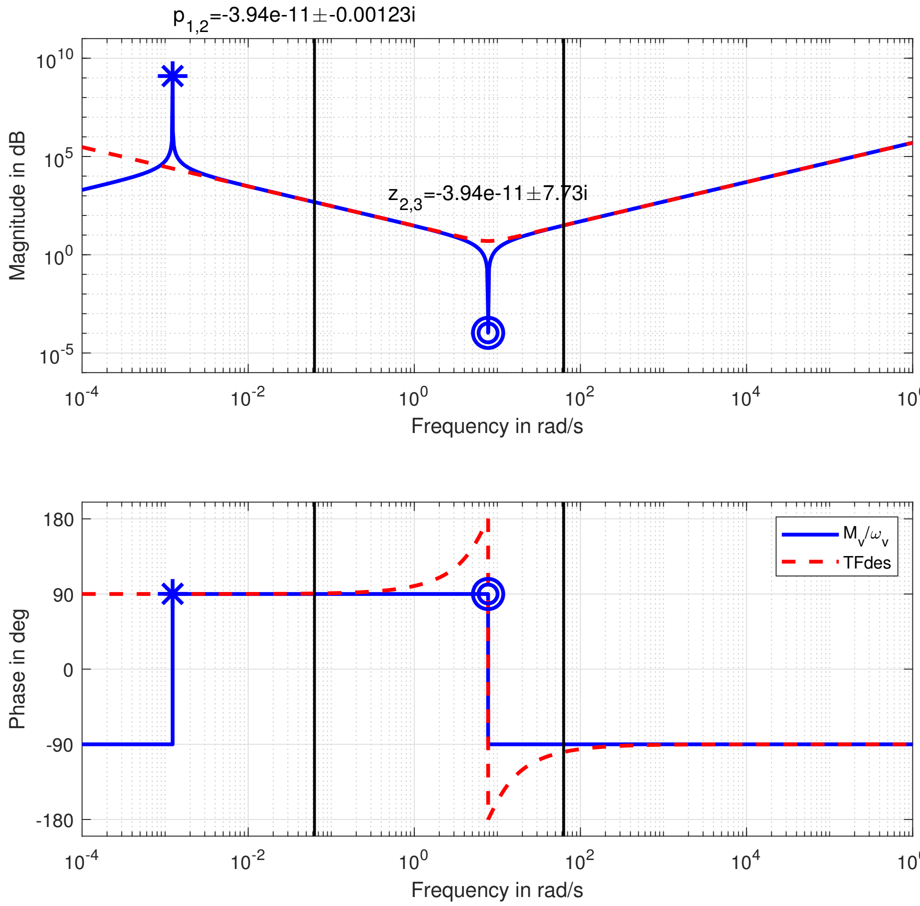


Figure 4.11: Impedance of a SPCMG when optimized to mimic a mass-spring-damper system. The area between the vertical lines are the optimized frequencies. Zeros are indicated with a circle, poles are indicated with a cross.

4.2.5. Optimization of PDXCoM

The resnorm of the cost function for the scissored pair gyros was 2.63×10^9. The optimized parameters can be seen in table 4.2. The bode plot of the found impedance function can be seen in Fig. 4.12 together with the desired transfer function. One pole and one zero are not shown because they exist at very high frequencies. The found impedance function has two poles located at $p_1 = -176.00$, $p_2 = -0.02$ and three zeros located at $z_1 = 0$ and, $z_{2,3} = -0.88 + 1.65i$. The damping in the system is $\zeta = \pm 1$ and the natural frequency $\omega_n = 0.02\,\text{rad/s}$ and $\omega_n = 176.00\,\text{rad/s}$.

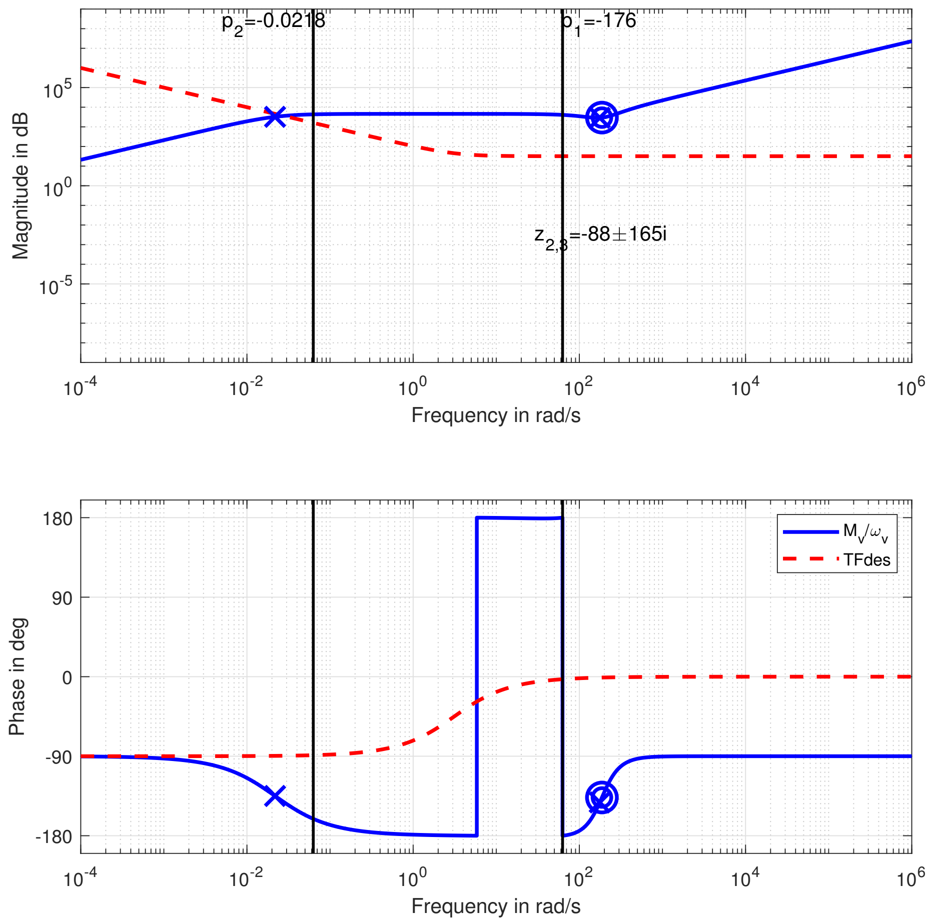


Figure 4.12: Impedance of a SPCMG when optimized to mimic XCoM. One pole and one zero are not shown because they exist at very high frequencies. The area between the vertical lines are the optimized frequencies. Zeros are indicated with a circle, poles are indicated with a cross.

Table 4.2: The optimized parameters of the SPCMG. Both the best possible parameters and the realistic parameters (RP) are shown.

Parameter	**Spring**	**Damper**	**Mass**	**Mass-Spring-Damper**	**PDXCoM**	**Unit**
Resnorm	2.4	4.4×10^{-5}	1.5×10^{-9}	1.7×10^{3}	2.63×10^{9}	
Resnorm RP	1.3	4.5×10^{-5}	0.71	1.7×10^{3}	6.4×10^{8}	
k	1.4×10^{-9}	3.2×10^{-14}	2858.0	7.6×10^{-7}	84.6	Nm/rad
k RP	6.7×10^{-13}	4.0×10^{-14}	2243.6	3.1×10^{-5}	0.31	Nm/rad
b	2.8×10^{-9}	2.8	2609.3	3.9×10^{-11}	3881.1	Nm/rad/s
b RP	1.4×10^{-9}	2.9	4.3	9.2×10^{-4}	4.33	Nm/rad/s
I_{ws}	2.0×10^{-6}	0.0018	0.024	0.0019	1.99	kgm^2
I_{ws}RP	1.6×10^{-5}	0.0018	0.016	4.2×10^{-4}	0.040	kgm^2
I_{wt}	9.8×10^{-6}	8.9×10^{-4}	0.012	9.4×10^{-4}	0.99	kgm^2
I_{wt}RP	8.2×10^{-6}	8.9×10^{-4}	0.0078	2.1×10^{-4}	0.020	kgm^2
I_{gs}	2.2×10^{-5}	1.4×10^{-4}	0.23	0.25	10.53	kgm^2
I_{gs}RP	1.4×10^{-5}	1.5×10^{-4}	1.2×10^{-5}	0.010	1.4×10^{-14}	kgm^2
I_{gg}	4.3×10^{-5}	2.8×10^{-4}	0.45	0.5	21.05	kgm^2
I_{gg} RP	2.9×10^{-5}	2.9×10^{-4}	2.4×10^{-5}	0.02	2.8×10^{-14}	kgm^2
$\gamma^\star$	-0.30	-1.6×10^{-4}	1.55	0.25	-0.05	rad
$\gamma^\star$ RP	0.30	-1.7×10^{-4}	0.073	1.88	-0.3	rad

4.2.6. Walking simulation

The moment that were applied on the human by the SPCMG are shown in this subsection. The parameters used for the CMG are the parameters that were found when the CMG was optimized to simulate a damper. The walking simulations with the other parameters can be seen in Appendix D. In Fig. 4.13 it can be seen that the maximum moment of $-0.97\,\mathrm{N\,m}$ is applied between the second left foot off and the second left foot strike. Furthermore, the angle γ stays between $-0.025\,\mathrm{rad}$ and $-0.01\,\mathrm{rad}$.

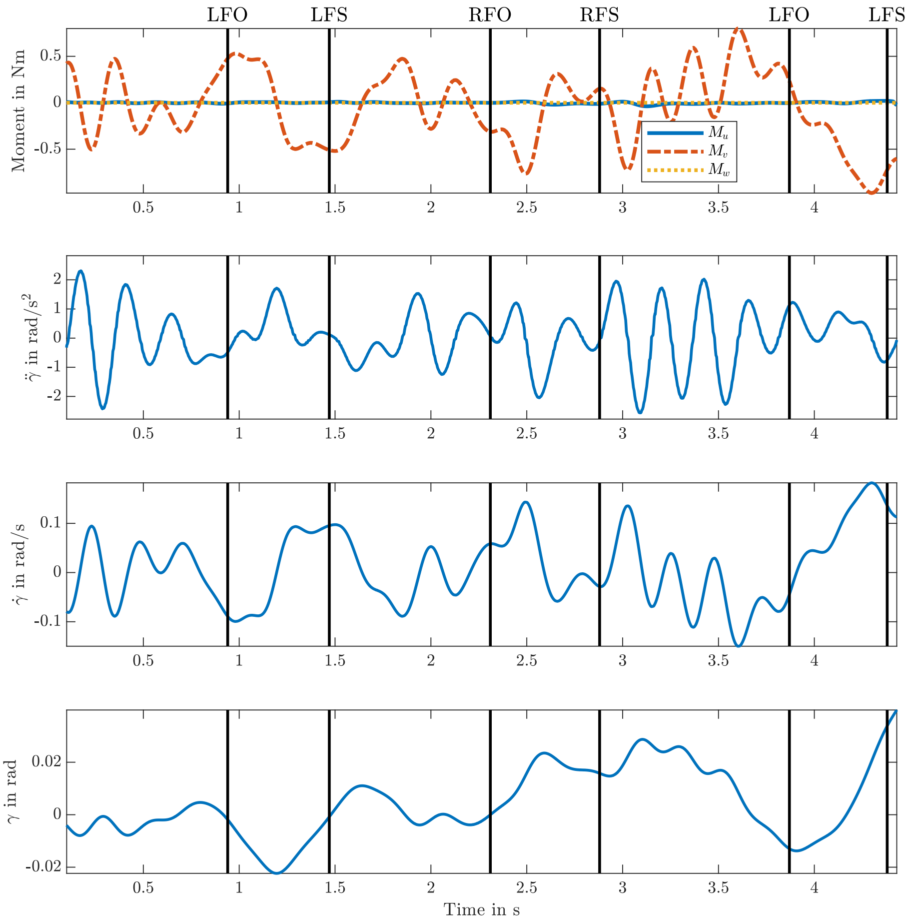


Figure 4.13: Forward simulation of the moments exerted on the human by the SPCMG. Also $\ddot{\gamma}$, $\dot{\gamma}$, and γ are shown. The walking speed was between 0-0.4 m/s. LFO = left foot off, LFS = left foot strike, RFO = right foot strike, RFS = right foot strike.

In Fig. 4.14it can be seen that the maximum moment of $-3.88\,\mathrm{Nm}$ is applied between the first left foot strike and the right foot off. Furthermore, the angle γ stays between -0.07 rad and 0.05 rad.

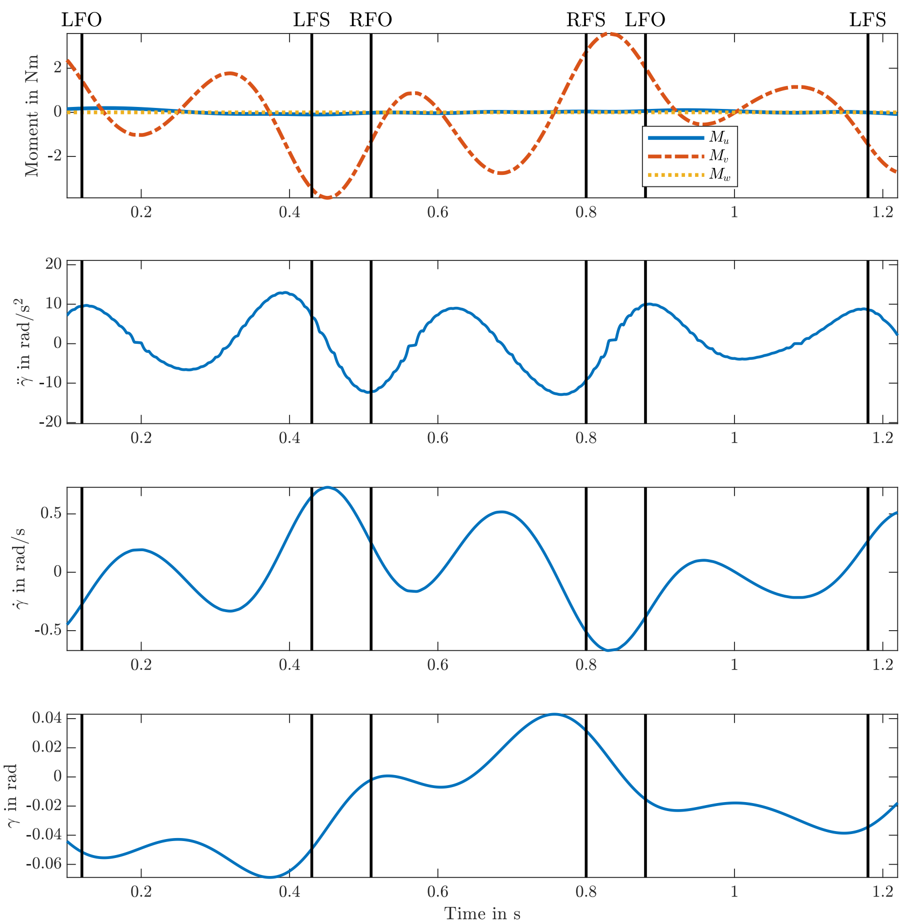


Figure 4.14: Forward simulation of the moments exerted on the human by the SPCMG. Also $\ddot{\gamma}$, $\dot{\gamma}$, and γ are shown. The walking speed was a self selected fast speed between 1.9-2.2 m/s. LFO = left foot off, LFS = left foot strike, RFO = right foot strike, RFS = right foot strike.

5
Discussion

Passively exploiting gyroscopic dynamics is a new concept as well as parameter optimization in frequency domain for CMGs. In the next chapter, the most important findings are discussed.

5.1. Discussion of CMG and SPCMG optimization

Since the results of the CMG and SPCMG are very similar, this section applies to both the CMG and SPCMG. It was possible to mimic the impedance of a spring, a damper, a mass, and a mass-spring-damper system with an (SP)CMG. However it was not possible to simulate the dynamcis of the PDXCoM.

A complex pole pair was placed at low frequencies when the (SP)CMG was optimized to simulate a spring. This, in combination with the zero at the origin, gives a magnitude slope of −20 dB/dec and a phase of 90°. This is the same as the desired impedance. Furthermore, a complex zero pair is placed outside the optimized frequency range. This initially gives a dip in the magnitude after which there is a magnitude slope of 20 dB/dec. It was possible to find a good fit for the damper when the (SP)CMG was optimized with and without bounds on the parameters. One zero exists at the origin, and therefore there is a magnitude slope of 20 dB/dec. One pole was placed at low frequencies to create a slope of 0 dB/dec. This also resulted into a phase of 180°. At frequencies outside the optimized frequency range, a complex zero pair and one pole are placed at the same frequency. This creates a small dip in the magnitude response and then creates a slope of 20 dB/dec and a phase of −90°. The algorithm found a good result for when the (SP)CMG was optimized to simulate a mass for both the optimization without bounds and with bounds. However, the strategy to find this fit were very different. The optimization without bounds found a result were $\gamma = 1.50$ rad. Combined with a flywheel with very small inertia, the gyroscopic effect is negligible. Furthermore, the inertia of the gimbal in the $\boldsymbol{\hat{e}}_v$ direction is 0.5 kgm^2, which was the desired inertia. The optimization with bounds on the parameters found a result where the gyroscopic effect had an effect on the impedance. The inertia of the gimbal is now very small and the combination of $\gamma = 0$ rad/ and large inertia for the flywheel create an impedance which is similar to the desired impedance of a mass. It was possible to simulate the impedance of a mass-spring-damper system with the (SP)CMG. One complex pole pair was placed at low frequencies to give the impedance a −20 dB/dec magnitude slope and a phase of 90°. Right where the desired impedance has two zeros, a complex zero pair is placed for the (SP)CMG impedance. Unlike for the desired impedance, this causes a dip in magnitude. However, at frequencies higher than the dip, the (SP)CMG impedance follows the desired impedance perfectly. It was not possible to get a good fit on the PDXCoM. One zero was placed at the origin which results in a −90° phase and a magnitude slope of 20 dB/dec. One pole is placed at 0.02 rad/s which gives a phase of −180° and a magnitude slope of 0 dB/dec. However, the desired impedance has a zero around 5 rad/s which gives a phase shift to 0°.

5.1.1. Explanation of the fit

For the impedance optimization, some assumptions were made. The angular velocity around which the equations of motion were optimized was 0 rad/s. A γ of 0 rad is used to simplify the equations even further. This is done because, in this configuration, the flywheel generated the highest torque in the $\hat{e}_v$ direction. This leads to the following impedance function for the CMG:

$$\frac{M_v}{\omega_v} = \frac{s(-J_t J_g s^2 - J_t bs - I_{ws}^2 \Omega^2 - J_t k)}{(J_g s^2 + bs + k)} \tag{5.1}$$

And the following impedance function for the SPCMG:

$$\frac{M_v}{\omega_v} = \frac{2s(-J_t J_g s^2 - J_t bs - I_{ws}^2 \Omega^2 - J_t k)}{(J_g s^2 + bs + k)} \tag{5.2}$$

From this, it is clear that the impedance of an SPCMG is two times the impedance of a CMG. The equation to solve squared equations is very well known. This equation can be used to compute the poles of the system. This leads to the following equation for both the CMG and SPCMG.

$$p_{1,2} = \frac{-b \pm \sqrt{b^2 - 4J_g k}}{2J_g} \tag{5.3}$$

From this, it can be derived that if two single poles are needed to fit the impedance, high damping is needed. Furthermore, the poles are independent of the parameters, I_{ws}, I_{gs}, and Ω. There is also a general equation to find the roots of cubic equations in the form of $As^3 + Bs^2 + Cs + D$ [42]. In this case however, the equation can be simplified to $s(As^2 + Bs + C)$. In this case, there is always one zero at the origin, and the other zeros can be computed using the following equation for both the CMG and the SPCMG.

$$z_{2,3} = \frac{J_t b \pm \sqrt{(-J_t b)^2 - 4(-J_t J_g)(-I_{ws}^2 \Omega^2 - J_t k)}}{-2J_t J_g} \tag{5.4}$$

It can be seen that the equation to compute the zeros is very similar to the equation to compute the poles. This equation is, however, dependant on all parameters. This means that the parameters I_{ws}, I_{gs}, and Ω can be used to change the zeros independently from the poles. However, it is only possible to change the discriminant with these parameters. With this knowledge, we can try to explain why the algorithm was able to find the found results.

A pure spring has a magnitude slope of −20 dB/dec and a phase of 90°. Since one zero always exists at 0 rad/s, two poles have to be placed at low frequencies. The damping has to be very low to accomplish this. However, if the damping is too low, the two poles become a complex pole pair. A complex pole pair has its influence around the natural frequency of the system. The natural frequency of the system can be calculated with: $s^2 + qs + r = s^2 + 2\zeta\omega_n s + \omega_n^2$. From this it follows that the natural frequency of the system is $\omega_\mathrm{n} = \sqrt{\frac{k}{J_\mathrm{g}}}$. Hence, J_g must be much larger than the spring stiffness k to place the poles at a low frequency. Therefore a low value for the spring stiffness and the damping was used for the initial guess.

A pure damper has a magnitude slope of 0 dB/dec and a phase of 180°. Because of the zero at the origin, one pole needs to be placed at frequencies lower than the optimized frequency range, and one pole needs to be placed at higher frequencies than the optimized frequency range. This is achieved by a high damping and low stiffness. Because of the low stiffness, the equation to compute the pole can be approximated with: $p_{1,2} = \frac{-b \pm b}{2J_g}$. From this, it is clear that with high damping, one pole is placed close to zero and one pole far outside the frequency range.

A pure mass has a magnitude slope of 20 dB/dec and a phase of −90. One strategy to match this was to have a γ that is close to $\frac{\pi}{2}$. This way, the system behaves like a mass. Furthermore, I_{ws} is very low to decrease the gyroscopic effect further. This strategy, however, cannot work for the optimization with realistic bounds on the parameters since, with this optimization, it is not possible to get the required inertia. Therefore, a high damping and very small inertia J_g were used to place the poles at frequencies higher than the optimized frequencies.

The PDXCoM has a phase of −90° and a magnitude slope of −20 dB/dec. Two poles would have to placed at low frequencies to get the same magnitude slope. This would, however, give a phase of 90°. This difference occurs because of the opposite sign for the PDXCoM and the impedance of the CMG. Since the optimized parameters cannot be negative, it is not possible to get a good fit on the PDXCoM with the CMG impedance.

5.1.2. Pole zero placement

The poles of both the CMG and SPCMG do not depend on the I_{ws}, I_{gs} and Ω. However, the zeros do depend on these parameters. This means that the zeros can be placed independently from the poles using these parameters. Basically, by changing the angular momentum in $\hat{\boldsymbol{e}}_v$ direction, the location of the zeros can be changed independently from the poles. In Fig. 5.1, it can be seen that the location of the zeros change when I_{ws} is changed. One zero always exist in the origin. The two other poles can be complex or real depending on the value of I_{ws}. The zeros are always be mirrored around $\frac{-b}{2J_g}$. Another way to change the angular

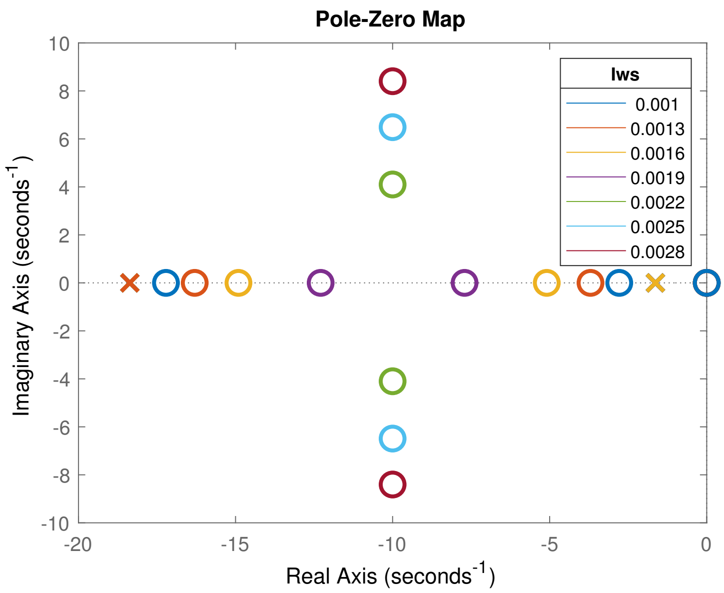


Figure 5.1: Plot which shows the effect of an changing I_{ws} on the location of the poles and zeros.

momentum in $\hat{\boldsymbol{e}}_v$ direction is to change γ. Changing γ gives similar results as changing I_{ws}. The effect of a changing γ on the zeros can be seen in Fig. 5.2. When $\gamma = \frac{\pi}{2}$, there is no angular momentum of the flywheel in $\hat{\boldsymbol{e}}_v$ direction. Therefore, the system behaves like a mass. Hence, there exists only one zero.

5.2. Discussion of walking simulation

5.2.1. Walking simulation of the CMG

The set of parameters that was used was the set for when the CMG was optimized to simulate a damper. The time response plots with different parameters can be found in Appendix D. Because of the damping, γ changed very little. This makes sure that the moments are mainly generated in the $\hat{e}_v$ direction. The generated moments are in the opposite direction, with respect to the angular velocity of the body. Therefore it would reduce the angular velocity and therefore, the CMG could help to maintain balance.

5.2.2. Walking simulation of the SPCMG

The set of parameters that was used was the set for when the CMG was optimized to simulate a damper. The time response plots with different parameters can be found in Appendix D. Because of the scissored pairing, the moments were mainly generated in the $\hat{e}_v$ direction. The moments were generated in the opposite direction compared to the angular velocity. Therefore, the SPCMG could be used for balance assistance.

5.3. Virtual stiffness, damping, and mass

With a reaction wheel, it should also be possible to simulate the dynamic behaviour of a spring, a damper, and a mass. Since in reaction wheels, there is no torque amplification, the impedance of a spring can just be simulated by adding that spring to the reaction wheel. This research shows that a CMG is capable of generating a virtual spring, damper and mass. The (SP)CMG was optimized to simulate a spring with a spring stiffness of 30.0 Nm/rad. To match this impedance, the (SP)CMG had to use a very low spring stiffness and damping co-

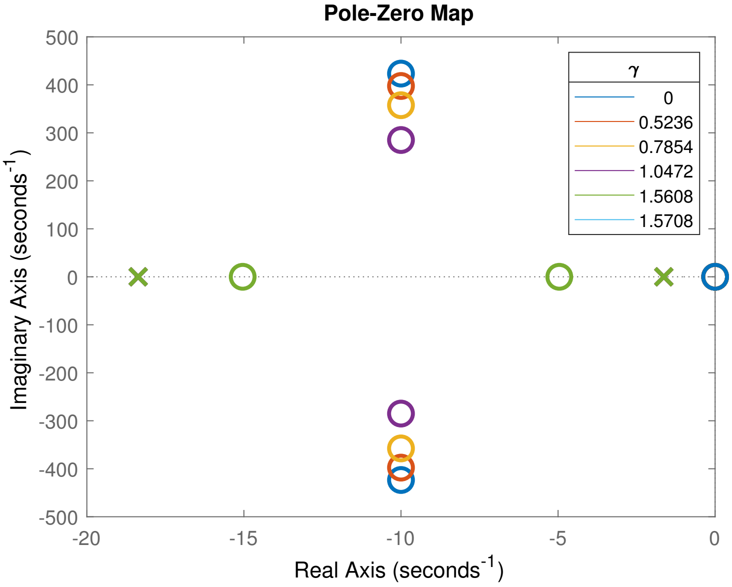


Figure 5.2: Plot which shows the effect of an changing γ on the location of the poles and zeros.

efficient. The spring stiffness comes from the inertia of the system. To create a damping, however, a damper was needed. A damper coefficient of 5.2 Nm/rad/s was needed to create the impedance of a damper with damping coefficient of 5.0 Nm/rad/s. This damping, however, does have an effect about another axis than to which the damper is applied. The angular momentum of the flywheel is needed to realize this coupling. It was also possible to simulate the impedance of inertia that was higher than the inertia of the actual system. Reaction wheel can also be used for balance assistance [44]. That the actual stiffness and mass are lower than the virtual stiffness shows that it is possible to generate a high stiffness or mass with a CMG without a high stiffness or mass.

5.4. Comparison between CMG and SPCMG

The impedance functions of the CMG and SPCMG are very similar when $\gamma = 0$, and all the angular velocities of the human are considered zero. The impedance for the SPCMG is two times the impedance for the CMG. However, the general impedance, Eq. (2.32) and Eq. (2.20), are very different. Both the CMG and SPCMG were able to simulate the desired damper between the optimized frequencies. The difference in dynamics can be seen in the walking simulation plots Fig. 4.6, Fig. 4.7, Fig. 4.13, and Fig. 4.14. From these plots, it can be seen that the moments generated by the CMG have about two times the magnitude of the moments generated by the SPCMG. This discrepancy occurs because with a single CMG, $\omega_u^\star$ contributes much more to the impedance than with the SPCMG. With the optimizations, $\omega_u^\star$ was considered zero, while with the walking simulation it ranged from -1 rad/s to 1 rad/s. Therefore, during the walking simulation, there are generated moments that were not accounted for with the optimization. Since $\omega_u^\star$ does not contribute as much to the impedance for the SPCMG, the impedance used during the optimization is a much better representation of the actual dynamics than the impedance for the CMG.

5.5. Optimization in frequency domain

The goal of the optimizations was to find a set of parameters with which a specific impedance could be achieved. One of the parameters that was optimized was the initial orientation of the flywheel, $\gamma^\star$. This parameter might be redundant since the optimization was performed for one impedance, $\frac{M_v}{\omega_v}$. Therefore, $\gamma^\star$ only has an influence on the angular momentum in the $\hat{\boldsymbol{e}}_v$ direction. The angular momentum can also be changed by altering the moment of inertia, I_{ws}. It would, however, be very useful to use $\gamma^\star$ when the impedance in multiple directions was optimized. In that case, $\gamma^\star$ would influence how the angular momentum is divided in each direction. It is also possible to optimize in time domain. In time domain, a specific desired moment would be given. The parameters would be adjusted to fit the desired moment as closely as possible. This is done, for example, in [21].

5.6. Cost function design

The cost function that was used for the optimization was:

$$C = w_1(imag(TF_{\text{des}} - imag(TF))) + real(TF_{\text{des}} - TF) \quad (5.5)$$

This cost function was able to perform twenty optimizations in 197.7 s. The best resnorm was 2.5×10^{-5}. It would have been possible to use a different cost function for the optimization. Another cost function that was tried can be seen in Eq. (5.6). A potential benefit of this cost function is that the punishment for de distance above the desired impedance and below the desired impedance is the same.

$$C = w_1(\angle TF_{\text{des}} - \angle TF) + (\ln|(TF_{\text{des}}) - (TF)|) \quad (5.6)$$

Performing twenty optimizations with realistic bounds took 790 s, which is over 13 minutes. Furthermore, the resnorm of the best optimization of the cost function from Eq. (5.6) was 1.01×10^3.

5.7. Parameter Design

When the optimization was successful in finding a set of parameters to simulate the desired impedance, the parameters are applicable. For example, when the CMG was optimized to simulate a damper, a damper with a damping coefficient of 5.2 Nm/rad/s is needed. A damper with this damping coefficient can be found and has a mass of 0.522 kg [26]. Furthermore, the flywheel has an inertia of 0.0034 kgm^2. Assuming the flywheel has a mass of 1 kg, the flywheel must have a radius of 0.082 m. The inertia of the damper would add to the inertia of the gimbal. When it is assumed that the damper with the right damping coefficient can be approximated as a solid cylinder, the approximate moments of inertia are $I_{gs} = 2.35 \times 10^{-4}\,\text{kgm}^2$ and $I_{gg} = 1.66 \times 10^{-4}\,\text{kgm}^2$. This is only slightly more than the inertia of the gimbal that was found with the optimization. The same damper can be used to simulate a mass. The main difference is that now also a spring is needed. Springs with a spring stiffness of 3633.3 Nm/rad are commercially available [14].

5.8. Future Directions

It would be useful to focus more on performing the optimization for multiple impedances to improve on current results. This way, a desired behaviour in multiple directions could be obtained. It can also be tried to fit the (SP)CMG impedance to new impedances. The impedances that were used in this study were arbitrarily chosen. Other measures of stability could be used. One popular measure of stability is "the maximum Lyapunov exponent", see **??**, firstly used by Dingwell et al. [8] in the context of gait stability. Other measures of stability that could be used are, "Foot Placement Estimator" by Millard et al. [28], a measure of stability in the sagittal plane. Or a similar measure in 3D, by Millard et al. [29]. Also, more complicated design features could be explored like end stops, which prevent the gimbal from rotating beyond a specific angle. Secondly, a passive mechanism with magnets could be explored. Magnets can be used to create an anti-spring. These have already been used to tune the natural frequency in passive-vibration isolators [17]. Anti-springs can also be used to create a bistable system [17]. The two stable equilibrium points could be used to rotate the gimbal between the two equilibrium points quickly. Lastly, nonlinear springs and dampers could be implemented in the design. This will, however, make the impedance optimization harder since the system has to be linearized to convert it into frequency domain.

Moreover, a prototype could be made. This way, it can be studied how people react to wearing a passively controlled (SP)CMG. The gait of the wearer will change due to the moments that are applied to the body. It is, however, also likely that the wearer would adapt to the new moments and therefore, might change their gait in unexpected ways.

6 Conclusion

By modelling a CMG and an SPCMG and optimizing their impedance, it was possible to replicate the dynamics of a spring, a damper, a mass, and a mass-spring-damper system. It was not possible to replicate the dynamics of the PDXCoM. When the found parameters were used in a walking simulation, it showed that the generated moments were in the opposite direction to the angular velocity of the walking person. This shows that a CMG and an SPCMG could be able to generate stabilizing moments for balance. A CMG generates higher moments than an SPCMGs when they have the same impedance. However, the moments generated by the SPCMG are easier to model and therefore, easier to predict than the CMG. This study lays the groundwork for impedance optimization of (SP)CMGs. Insight is gained in what the influence of the design parameters is on the behaviour of the (SP)CMG. Furthermore, it should now be easy to match new desired impedances to the impedance of an (SP)CMG.

Bibliography

[1] Sakineh B Akram, James S Frank, Aftab E Patla, and John HJ Allum. Balance control during continuous rotational perturbations of the support surface. *Gait & posture*, 27(3):393–398, 2008.

[2] MJ Ashley, CI Gryfe, and A Amies. A longitudinal study of falls in an elderly population ii. some circumstances of falling. *Age and ageing*, 6(4):211–220, 1977.

[3] AV Beznos, AM Formal'Sky, EV Gurfinkel, DN Jicharev, AV Lensky, KV Savitsky, and LS Tchesalin. Control of autonomous motion of two-wheel bicycle with gyroscopic stabilisation. In *Proceedings. 1998 IEEE International Conference on Robotics and Automation (Cat. No. 98CH36146)*, volume 3, pages 2670–2675. IEEE, 1998.

[4] Tjitske Anke Boonstra, Alfred C Schouten, and Herman Van der Kooij. Identification of the contribution of the ankle and hip joints to multi-segmental balance control. *Journal of neuroengineering and rehabilitation*, 10(1):23, 2013.

[5] H Benjamin Brown and Yangsheng Xu. A single-wheel, gyroscopically stabilized robot. In *Proceedings of IEEE international conference on robotics and automation*, volume 4, pages 3658–3663. IEEE, 1996.

[6] Jimmy Chiu and Ambarish Goswami. Design of a wearable scissored-pair control moment gyroscope (sp-cmg) for human balance assist. In *ASME 2014 international design engineering technical conferences and computers and information in engineering conference*, pages V05AT08A023–V05AT08A023. American Society of Mechanical Engineers, 2014.

[7] Pei Di, Jian Huang, Shotaro Nakagawa, Kosuke Sekiyama, and Toshio Fukuda. Real-time fall and overturn prevention control for human-cane robotic system. In *IEEE ISR 2013*, pages 1–6. IEEE, 2013.

[8] Jonathan B Dingwell, Joseph Paul Cusumano, D Sternad, and PR Cavanagh. Slower speeds in patients with diabetic neuropathy lead to improved local dynamic stability of continuous overground walking. *Journal of biomechanics*, 33(10):1269–1277, 2000.

[9] Denise Engelhart, Jantsje H Pasma, Alfred C Schouten, Ronald GKM Aarts, Carel GM Meskers, Andrea B Maier, and Herman van der Kooij. Adaptation of multijoint coordination during standing balance in healthy young and healthy old individuals. *Journal of neurophysiology*, 115(3):1422–1435, 2015.

[10] Bernard Friedland. *Control system design: an introduction to state-space methods.* Courier Corporation, 2012.

[11] George F Fuller. Falls in the elderly. *American family physician*, 61(7):2159–68, 2000.

[12] Milad Geravand, Wolfgang Rampeltshammer, and Angelika Peer. Control of mobility assistive robot for human fall prevention. In *2015 IEEE International Conference on Rehabilitation Robotics (ICORR)*, pages 882–887. IEEE, 2015.

[13] Torkel Glad. *Step responses of nonlinear non-minimum phase systems.* Linköping University Electronic Press, 2004.

[14] LESJÖFORS GROUP. The Spring Catalogue 15. Technical report, Lesjöfors, 2017.

[15] AL Hof, MGJ Gazendam, and WE Sinke. The condition for dynamic stability. *Journal of biomechanics*, 38 (1):1–8, 2005.

[16] Sang-Ho Hyon, Jun Morimoto, Takamitsu Matsubara, Tomoyuki Noda, and Mitsuo Kawato. Xor: Hybrid drive exoskeleton robot that can balance. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3975–3981. IEEE, 2011.

[17] RA Ibrahim. Recent advances in nonlinear passive vibration isolators. *Journal of sound and vibration*, 314(3-5):371–452, 2008.

[18] Jane Jensen, Lars Nyberg, Yngve Gustafson, and Lillemor Lundin-Olsson. Fall and injury prevention in residential care—effects in residents with higher and lower levels of cognition. *Journal of the American geriatrics society*, 51(5):627–635, 2003.

[19] Shuuji Kajita, Tomio Yamaura, and Akira Kobayashi. Dynamic walking control of a biped robot along a potential energy conserving orbit. *IEEE Transactions on robotics and automation*, 8(4):431–438, 1992.

[20] Pekka Kannus, Jari Parkkari, Seppo Niemi, and Mika Palvanen. Fall-induced deaths among elderly people. *American journal of public health*, 95(3):422–424, 2005.

[21] Poya Khalaf, Hanz Richter, Antonie J Van Den Bogert, and Dan Simon. Multi-objective optimization of impedance parameters in a prosthesis test robot. In *ASME 2015 Dynamic Systems and Control Conference*. American Society of Mechanical Engineers Digital Collection, 2015.

[22] Young-Gyu Ko, John H Challis, and Karl M Newell. Postural coordination patterns as a function of dynamics of the support surface. *Human movement science*, 20(6):737–764, 2001.

[23] VJ Lappas, WH Steyn, and C Underwood. Design and testing of a control moment gyroscope cluster for small satellites. *Journal of Spacecraft and Rockets*, 42(4):729–739, 2005.

[24] Daniel Lemus, Jan van Frankenhuyzen, and Heike Vallery. Design and evaluation of a balance assistance control moment gyroscope. *Journal of Mechanisms and Robotics*, 9(5):051007, 2017.

[25] Dustin Li and Heike Vallery. Gyroscopic assistance for human balance. In *2012 12th IEEE International Workshop on Advanced Motion Control (AMC)*, pages 1–6. IEEE, 2012.

[26] Kinetrol Ltd. Kinetrol Dashpots, Rotary Dampers. Technical report, Kinetrol, 2015.

[27] Duane T McRuer and Henry R Jex. A review of quasi-linear pilot models. *IEEE transactions on human factors in electronics*, (3):231–249, 1967.

[28] Matthew Millard, Derek Wight, John McPhee, Eric Kubica, and David Wang. Human foot placement and balance in the sagittal plane. *Journal of biomechanical engineering*, 131(12):121001, 2009.

[29] Matthew Millard, John McPhee, and Eric Kubica. Foot placement and balance in 3d. *Journal of Computational and Nonlinear Dynamics*, 7(2):021015, 2012.

[30] Osamu Nishihara and Hiroshi Matsuhisa. Design optimization of passive gyroscopic damper: Stability degree maximimation. *JSME International Journal Series C Mechanical Systems, Machine Elements and Manufacturing*, 40(4):643–651, 1997.

[31] Federico Parietti, Kameron C Chan, Banks Hunter, and H Harry Asada. Design and control of supernumerary robotic limbs for balance augmentation. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5010–5017. IEEE, 2015.

[32] Tristan Perez and Paul Steinmann. Advances in gyro-stabilisation of vessel roll motion. In *Proceedings of the International Maritime Conference, Pacific 2008*, pages 682–692. Royal Institution of Naval Architects (RINA), 2008.

[33] Michael Peshkin, David A Brown, Julio J Santos-Munné, Alex Makhlin, Ela Lewis, J Edward Colgate, James Patton, and Doug Schwandt. Kineassist: A robotic overground gait and balance training device. In *9th International Conference on Rehabilitation Robotics, 2005. ICORR 2005.*, pages 241–246. IEEE, 2005.

[34] Stephen N Robinovitch, Fabio Feldman, Yijian Yang, Rebecca Schonnop, Pet Ming Leung, Thiago Sarraf, Joanie Sims-Gould, and Marie Loughin. Video capture of the circumstances of falls in elderly people residing in long-term care: an observational study. *The Lancet*, 381(9860):47–54, 2013.

[35] Mark W Rogers and Marie-Laure Mille. Lateral stability and falls in older people. *Exercise and sport sciences reviews*, 31(4):182–187, 2003.

[36] Parineak Romtrairat, Chanyaphan Virulsri, and Pairat Tangpornprasert. An application of scissored-pair control moment gyroscopes in a design of wearable balance assistance device for the elderly. *Journal of biomechanics*, 2019.

[37] Laurence Z Rubenstein. Falls in older people: epidemiology, risk factors and strategies for prevention. *Age and ageing*, 35(suppl_2):ii37–ii41, 2006.

[38] Alice C Scheffer, Marieke J Schuurmans, Nynke Van Dijk, Truus Van Der Hooft, and Sophia E De Rooij. Fear of falling: measurement strategy, prevalence, risk factors and consequences among older persons. *Age and ageing*, 37(1):19–24, 2008.

[39] Céline Schreiber and Florent Moissenet. A multimodal dataset of human gait at different walking speeds established on injury-free adult participants. *Scientific data*, 6(1):111, 2019.

[40] Antonie J van den Bogert, MJ Pavol, and Mark D Grabiner. Response time is more important than walking speed for the ability of older adults to avoid a fall after a trip. *Journal of biomechanics*, 35(2):199–205, 2002.

[41] Bruno J Vellas, Sharon J Wayne, Linda J Romero, Richard N Baumgartner, and Philip J Garry. Fear of falling and restriction of mobility in elderly fallers. *Age and ageing*, 26(3):189–193, 1997.

[42] Eric W Weisstein. Cubic formula. *omega*, 86:87, 2002.

[43] David A Winter. Human balance and posture control during standing and walking. *Gait & posture*, 3(4): 193–214, 1995.

[44] Tytus Wojtara, Makoto Sasaki, Hitoshi Konosu, Masashi Yamashita, Shingo Shimoda, Fady Alnajjar, and Hidenori Kimura. Artificial balancer–supporting device for postural reflex. *Gait & posture*, 35(2):316–321, 2012.

A

Appendix A

A.1. Written out equations of motion

The equations of motion found for the CMG can be seen in Eq. (A.1). To reduce the lenght of the equations $\sin\gamma$ is written as $s\gamma$ and $\cos\gamma$ is written as $c\gamma$.

$$
{}^{\mathcal{B}}(\dot{\boldsymbol{H}})_{\mathcal{N}} = \begin{pmatrix}
c\gamma(I_{gs}(c\gamma(\dot{\omega}_u+\dot{\gamma}\omega_v)+s\gamma(\dot{\omega}_v-\dot{\gamma}\omega_u))+I_{ws}(c\gamma(\dot{\omega}_u+\dot{\gamma}\omega_v)+s\gamma(\dot{\omega}_v-\dot{\gamma}\omega_u))+I_{gg}(\dot{\gamma}+\omega_w)(\omega_v c\gamma-\omega_u s\gamma)\ldots \\
-I_{gt}(\dot{\gamma}+\omega_w)(\omega_v c\gamma-\omega_u s\gamma))-s\gamma(I_{gt}(c\gamma(\dot{\omega}_v-\dot{\gamma}\omega_u)-s\gamma(\dot{\omega}_u+\dot{\gamma}\omega_v))+I_{wt}(c\gamma(\dot{\omega}_v-\dot{\gamma}\omega_u)-s\gamma(\dot{\omega}_u+\dot{\gamma}\omega_v))\ldots \\
+I_{ws}(\dot{\gamma}+\omega_w)(\Omega+\omega_u c\gamma+\omega_v s\gamma)-I_{gg}(\dot{\gamma}+\omega_w)(\omega_u c\gamma+\omega_v s\gamma)+I_{gs}(\dot{\gamma}+\omega_w)(\omega_u c\gamma+\omega_v s\gamma)\ldots \\
-I_{wt}(\dot{\gamma}+\omega_w)(\omega_u c\gamma+\omega_v s\gamma)); \\
\\
c\gamma(I_{gt}(c\gamma(\dot{\omega}_v-\dot{\gamma}\omega_u)-s\gamma(\dot{\omega}_u+\dot{\gamma}\omega_v))+I_{wt}(c\gamma(\dot{\omega}_v-\dot{\gamma}\omega_u)-s\gamma(\dot{\omega}_u+\dot{\gamma}\omega_v))+I_{ws}(\dot{\gamma}+\omega_w)(\Omega+\omega_u c\gamma+\omega_v s\gamma)\ldots \\
-I_{gg}(\dot{\gamma}+\omega_w)(\omega_u c\gamma+\omega_v s\gamma)+I_{gs}(\dot{\gamma}+\omega_w)(\omega_u c\gamma+\omega_v s\gamma)-I_{wt}(\dot{\gamma}+\omega_w)(\omega_u c\gamma+\omega_v s\gamma))+s\gamma(I_{gs}(c\gamma(\dot{\omega}_u+\dot{\gamma}\omega_v)\ldots \\
+s\gamma(\dot{\omega}_v-\dot{\gamma}\omega_u))+I_{ws}(c\gamma(\dot{\omega}_u+\dot{\gamma}\omega_v)+s\gamma(\dot{\omega}_v-\dot{\gamma}\omega_u))+I_{gg}(\dot{\gamma}+\omega_w)(\omega_v c\gamma-\omega_u s\gamma)-I_{gt}(\dot{\gamma}+\omega_w)(\omega_v c\gamma-\omega_u s\gamma)); \\
\\
I_{gg}(\ddot{\gamma}+\dot{\omega}_w)+I_{wt}(\ddot{\gamma}+\dot{\omega}_w)-I_{gs}(\omega_u c\gamma+\omega_v s\gamma)(\omega_v c\gamma-\omega_u s\gamma)+I_{gt}(\omega_u c\gamma+\omega_v s\gamma)(\omega_v c\gamma-\omega_u s\gamma)\ldots \\
+I_{wt}(\omega_u c\gamma+\omega_v s\gamma)(\omega_v c\gamma-\omega_u s\gamma)-I_{ws}(\omega_v c\gamma-\omega_u s\gamma)(\Omega+\omega_u c\gamma+\omega_v s\gamma)
\end{pmatrix} \tag{A.1}
$$

The equations of motion found for the CMG can be seen in Eq. (A.2).

$$
{}^{\mathcal{B}}M = \begin{pmatrix}
2I_{gs}\dot{\omega}_u(s\gamma^2-1)-2I_{wt}\dot{\omega}_u s\gamma^2-2I_{gg}\omega_v\omega_w-2I_{gt}\dot{\omega}_u s\gamma^2+2I_{ws}\dot{\omega}_u(s\gamma^2-1)\ldots \\
-2I_{gt}\omega_v\omega_w(s\gamma^2-1)+2I_{ws}\Omega\omega_w s\gamma+2I_{gs}\dot{\gamma}\omega_u s2\gamma-2I_{gt}\dot{\gamma}\omega_u s2\gamma+2I_{ws}\dot{\gamma}\omega_u s2\gamma\ldots \\
-2I_{wt}\dot{\gamma}\omega_u s2\gamma+2I_{gs}\omega_v\omega_w s\gamma^2+2I_{ws}\omega_v\omega_w s\gamma^2-2I_{wt}\omega_v\omega_w s\gamma^2; \\
\\
2I_{gs}\dot{\omega}_v(c\gamma^2-1)-2I_{wt}\dot{\omega}_v c\gamma^2-2I_{gt}\dot{\omega}_v c\gamma^2+2I_{ws}\dot{\omega}_v(c\gamma^2-1)\ldots \\
+2I_{gg}\omega_u\omega_w+2I_{gt}\omega_u\omega_w(c\gamma^2-1)-2I_{ws}\dot{\gamma}\Omega c\gamma-2I_{gs}\dot{\gamma}\omega_v s2\gamma\ldots \\
+2I_{gt}\dot{\gamma}\omega_v s2\gamma-2I_{ws}\dot{\gamma}\omega_v s2\gamma+2I_{wt}\dot{\gamma}\omega_v s2\gamma-2I_{gs}\omega_u\omega_w c\gamma^2-2I_{ws}\omega_u\omega_w c\gamma^2+2I_{wt}\omega_u\omega_w c\gamma^2; \\
\\
2\gamma_0 k-2I_{wt}\ddot{\gamma}-2b\dot{\gamma}-2I_{gg}\ddot{\gamma}-2\gamma_k-I_{gs}\omega_u^2 s2\gamma+I_{gt}\omega_u^2 s2\gamma+I_{gs}\omega_v^2 s2\gamma\ldots \\
-I_{gt}\omega_v^2 s2\gamma-I_{ws}\omega_u^2 s2\gamma+I_{wt}\omega_u^2 s2\gamma+I_{ws}\omega_v^2 s2\gamma-I_{wt}\omega_v^2 s2\gamma+2I_{ws}\Omega\omega_v c\gamma
\end{pmatrix} \tag{A.2}
$$

A.2. Lagrange approach for a single CMG in the body-fixed Frame

To check if the equations of motion are correct, also the Lagrange method was used to compute the equations of motion. For the generalized coordinates, γ was used. The kinetic energy used for the this method was:

$$
T = \frac{1}{2}\left((\Omega\boldsymbol{g}_s+\dot{\gamma}\boldsymbol{g}_g+{}^{\mathcal{G}}\mathbf{R}(\gamma)^{\mathcal{B}}_{\mathcal{B}}\boldsymbol{\omega}_{\mathcal{B}/\mathcal{N}})^T\mathbf{I}_w(\Omega\boldsymbol{g}_s+\dot{\gamma}\boldsymbol{g}_g+{}^{\mathcal{G}}\mathbf{R}(\gamma)^{\mathcal{B}}_{\mathcal{B}}\boldsymbol{\omega}_{\mathcal{B}/\mathcal{N}})+(\dot{\gamma}\boldsymbol{g}_g+{}^{\mathcal{G}}\mathbf{R}(\gamma)^{\mathcal{B}}_{\mathcal{B}}\boldsymbol{\omega}_{\mathcal{B}/\mathcal{N}})^T\mathbf{I}_g(\dot{\gamma}\boldsymbol{g}_g+{}^{\mathcal{G}}\mathbf{R}(\gamma)^{\mathcal{B}}_{\mathcal{B}}\boldsymbol{\omega}_{\mathcal{B}/\mathcal{N}})\right) \tag{A.3}
$$

The potential energy used was:

$$V = \frac{1}{2}(k(\gamma - \gamma_0)^2) \tag{A.4}$$

The Lagrangian, L, of the system is:

$$L = T - V \tag{A.5}$$

The non conservative generalized forces will be in Q:

$$Q = -b\dot{\gamma} \tag{A.6}$$

To compute the equations of motion the following equation was used:

$$\frac{\mathrm{d}}{\mathrm{d}t}(\frac{\partial L}{\partial \dot{\gamma}}) - \frac{\partial L}{\partial \gamma} = Q \tag{A.7}$$

This can be solved for $\ddot{q}$ which leads to:

$$\begin{gathered}\ddot{\gamma} = -[b\dot{\gamma} - k(\gamma_0 - \gamma) + \dot{\omega}_w(I_{gg} + I_{wt}) - I_{gs}(\omega_u \cos\gamma + \omega_v \sin\gamma)(\omega_v \cos\gamma - \omega_u \sin\gamma) + I_{gt}(\omega_u \cos\gamma + \omega_v \sin\gamma) \\ (\omega_v \cos\gamma - \omega_u \sin\gamma) + I_{wt}(\omega_u \cos\gamma + \omega_v \sin\gamma)(\omega_v \cos\gamma - \omega_u \sin\gamma) - I_{ws}(\omega_v \cos\gamma - \omega_u \sin\gamma) \\ (\Omega + \omega_u \cos\gamma + \omega_v \sin\gamma)]/(I_{gg} + I_{wt})\end{gathered} \tag{A.8}$$

Which is the same as Eq. (2.11)

A.3. Lagrange approach for scissored pair gyro

To generate the Lagrange equations of motion, one generalized coordinate was used, $q = \gamma$. The kinetic energy ,T, of the system are defined as:

$$\begin{gathered}T1 = \tfrac{1}{2}\left((\Omega \boldsymbol{g}_s + \dot{\gamma}\boldsymbol{g}_g + {}^{\mathcal{G}_1}\mathbf{R}^{\mathcal{B}}_{\mathcal{B}}\boldsymbol{\omega}_{\mathcal{B}/\mathcal{N}})^T \mathbf{I}_w (\Omega \boldsymbol{g}_s + \dot{\gamma}\boldsymbol{g}_g + {}^{\mathcal{G}_1}\mathbf{R}^{\mathcal{B}}_{\mathcal{B}}\boldsymbol{\omega}_{\mathcal{B}/\mathcal{N}}) + (\dot{\gamma}\boldsymbol{g}_g + {}^{\mathcal{G}_1}\mathbf{R}^{\mathcal{B}}_{\mathcal{B}}\boldsymbol{\omega}_{\mathcal{B}/\mathcal{N}})^T \mathbf{I}_g (\dot{\gamma}\boldsymbol{g}_g + {}^{\mathcal{G}_1}\mathbf{R}^{\mathcal{B}}_{\mathcal{B}}\boldsymbol{\omega}_{\mathcal{B}/\mathcal{N}})\right) \\ T2 = \tfrac{1}{2}\left((\Omega \boldsymbol{g}_s - \dot{\gamma}\boldsymbol{g}_g + {}^{\mathcal{G}_2}\mathbf{R}^{\mathcal{B}}_{\mathcal{B}}\boldsymbol{\omega}_{\mathcal{B}/\mathcal{N}})^T \mathbf{I}_w (\Omega \boldsymbol{g}_s - \dot{\gamma}\boldsymbol{g}_g + {}^{\mathcal{G}_2}\mathbf{R}^{\mathcal{B}}_{\mathcal{B}}\boldsymbol{\omega}_{\mathcal{B}/\mathcal{N}}) + (-\dot{\gamma}\boldsymbol{g}_g + {}^{\mathcal{G}_2}\mathbf{R}^{\mathcal{B}}_{\mathcal{B}}\boldsymbol{\omega}_{\mathcal{B}/\mathcal{N}})^T \mathbf{I}_g (-\dot{\gamma}\boldsymbol{g}_g + {}^{\mathcal{G}_2}\mathbf{R}^{\mathcal{B}}_{\mathcal{B}}\boldsymbol{\omega}_{\mathcal{B}/\mathcal{N}})\right) \\ T = T1 + T2\end{gathered} \tag{A.9}$$

Where T1is the kinetic energy of the first CMG respectively and T2, is the kinetic energy of the second CMG respectively. The potential energy is twice the potential energy of a single CMG, which was given in Eq. (A.4). The Lagrangian of the system is then is done in the same manner as Eq. (A.5). There are no external forces applied to the system so Q consists only of non conservative forces, which are only the two dampers.

$$Q = -2b\dot{\gamma} \tag{A.10}$$

To compute the equations of motion, equation Eq. (A.7) was used. When this is solved for $\ddot{q}$, this results in:

$$\begin{gathered}\ddot{\gamma} = -[2b\dot{\gamma} - 2(\gamma_0 - \gamma)k + I_{gs}\omega_u^2 \sin(2\gamma) - I_{gt}\omega_u^2 \sin(2\gamma) - I_{gs}\omega_v^2 \sin(2\gamma) \\ + I_{gt}\omega_v^2 \sin(2\gamma) + I_{ws}\omega_u^2 \sin(2\gamma) - I_{wt}\omega_u^2 \sin(2\gamma) - I_{ws}\omega_v^2 \sin(2\gamma) \\ + I_{wt}\omega_v^2 \sin(2\gamma) + 2I_{ws}\Omega\omega_u \sin(\gamma)]/[2(I_{gg} + I_{wt})]\end{gathered} \tag{A.11}$$

Which is equivalent to Eq. (2.27).

A.4. Numerical differentiation

Numerical differentiation was used to validate ${}^{\mathcal{B}}(\dot{\boldsymbol{H}})_{\mathcal{N}}$. To do this, first ${}^{\mathcal{G}}\boldsymbol{H}$ had to be transformed to the natural frame. This was done by first transforming it to the body fixed frame and then to the natural frame. The rotation matrix from the gimbal fixed frame to the body fixed frame is explained in Section 2.1. The rotation matrix from the body fixed frame to the natural frame is:

$$\mathbf{R}_\phi = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \sin\phi \end{pmatrix}, \quad \mathbf{R}_\theta = \begin{pmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{pmatrix}, \quad \mathbf{R}_\psi = \begin{pmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{pmatrix}, \tag{A.12}$$

$${}^{\mathcal{N}}\mathbf{R}_{\mathcal{B}} = \mathbf{R}_\phi \mathbf{R}_\theta \mathbf{R}_\psi \tag{A.13}$$

This leads to:

$${}^{\mathcal{N}}\mathbf{H} = {}^{\mathcal{N}}\mathbf{R}^{\mathcal{B}}_{\mathcal{B}}\mathbf{R}^{\mathcal{G}}_{\mathcal{G}}\mathbf{H} \tag{A.14}$$

Next, values are given to all the variables and the difference between the time step is taken. This difference should now equal ${}^{\mathcal{B}}(\dot{\boldsymbol{H}})_{\mathcal{N}}$ when it is rotated to $\mathcal{N}$. The plot of both can be seen in

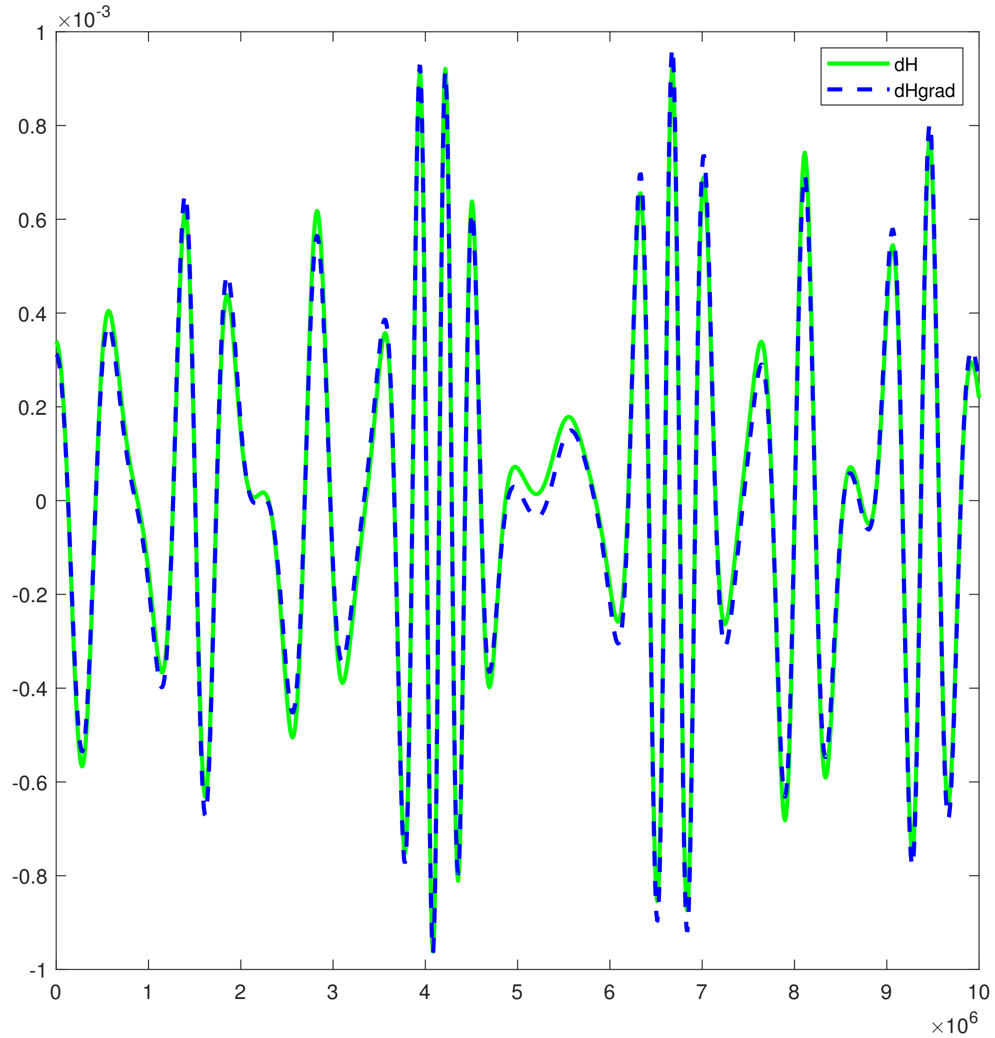


Figure A.1: Plot of the numerical value of $\dot{H}_{\mathcal{N}}$ and the gradient of $H_{\mathcal{N}}$

B

Appendix B

B.1. Impedance of a Single CMG

$$\frac{M_u}{\omega_u} = [\omega_w^\star sin(2\gamma^\star)(I_{ws} - I_{wt})]/2 - s(J_s - I_{ws}sin(\gamma^\star)^2 + I_{wt}sin(\gamma^\star)^2)$$

$$+\frac{s\omega_v^\star[(I_{wt}-I_{ws})\omega_v^\star cos(2\gamma^\star)+(I_{ws}-I_{wt})\omega_u^\star sin(2\gamma^\star)+I_{ws}\Omega sin(\gamma^\star)]}{k+bs+J_g s^2+(I_{ws}-I_{wt})\omega_u^{\star 2}cos(2\gamma^\star)+(I_{wt}-I_{ws})\omega_v^{\star 2}cos(2\gamma^\star)+I_{ws}\Omega\omega_u^\star cos(\gamma^\star)+I_{ws}\Omega\omega_v^\star sin(\gamma^\star)+2(I_{ws}-I_{wt})\omega_u^\star\omega_v^\star sin(2\gamma^\star)}$$

$$-\frac{\omega_w^\star[(I_{ws}-I_{wt})\omega_u^\star cos(2\gamma^\star)+(I_{ws}-I_{wt})\omega_v^\star sin(2\gamma^\star)+I_{ws}\Omega cos(\gamma^\star)][(I_{wt}-I_{ws})\omega_v^\star cos(2\gamma^\star)+(I_{ws}-I_{wt})\omega_u^\star sin(2\gamma^\star)+I_{ws}\Omega sin(\gamma^\star)]}{k+bs+J_g s^2+(I_{ws}-I_{wt})\omega_u^{\star 2}cos(2\gamma^\star)+(I_{wt}-I_{ws})\omega_v^{\star 2}cos(2\gamma^\star)+I_{ws}\Omega\omega_u^\star cos(\gamma^\star)+I_{ws}\Omega\omega_v^\star sin(\gamma^\star)+2(I_{ws}-I_{wt})\omega_u^\star\omega_v^\star sin(2\gamma^\star)}$$

(B.1)

$$\frac{M_v}{\omega_u} = \omega_w^\star(I_{gg} - I_{gs} - I_{ws} + I_{wt} + I_{ws}sin(\gamma^\star)^2 - I_{wt}sin(\gamma^\star)^2) - (ssin(2\gamma^\star)(I_{ws} - I_{wt}))/2$$

$$-\frac{s\omega_u^\star(I_{wt}\omega_v^\star cos(2\gamma^\star)-I_{ws}\omega_v^\star cos(2\gamma^\star)+I_{ws}\omega_u^\star sin(2\gamma^\star)-I_{wt}\omega_u^\star sin(2\gamma^\star)+I_{ws}Omegasin(\gamma^\star))}{k+bs+J_g s^2+(I_{ws}-I_{wt})\omega_u^{\star 2}cos(2\gamma^\star)+(I_{wt}-I_{ws})\omega_v^{\star 2}cos(2\gamma^\star)+I_{ws}\Omega\omega_u^\star cos(\gamma^\star)+I_{ws}\Omega\omega_v^\star sin(\gamma^\star)+2(I_{ws}-I_{wt})\omega_u^\star\omega_v^\star sin(2\gamma^\star}$$

(B.2)

$$-\frac{\omega_w^\star(I_{wt}\omega_v^\star cos(2\gamma^\star)-I_{ws}\omega_v^\star cos(2\gamma^\star)+I_{ws}\omega_u^\star sin(2\gamma^\star)-I_{wt}\omega_u^\star sin(2\gamma^\star)+I_{ws}Omegasin(\gamma^\star))^2}{k+bs+J_g s^2+(I_{ws}-I_{wt})\omega_u^{\star 2}cos(2\gamma^\star)+(I_{wt}-I_{ws})\omega_v^{\star 2}cos(2\gamma^\star)+I_{ws}\Omega\omega_u^\star cos(\gamma^\star)+I_{ws}\Omega\omega_v^\star sin(\gamma^\star)+2(I_{ws}-I_{wt})\omega_u^\star\omega_v^\star sin(2\gamma^\star}$$

$$\frac{M_w}{\omega_u} = -\frac{(k+bs)(I_{wt}\omega_v^\star cos(2\gamma^\star)-I_{ws}\omega_v^\star cos(2\gamma^\star)+I_{ws}\omega_u^\star sin(2\gamma^\star)-I_{wt}\omega_u^\star sin(2\gamma^\star)+I_{ws}\Omega sin(\gamma^\star))}{k+bs+J_g s^2+(I_{ws}-I_{wt})\omega_u^{\star 2}cos(2\gamma^\star)+(I_{wt}-I_{ws})\omega_v^{\star 2}cos(2\gamma^\star)+I_{ws}\Omega\omega_u^\star cos(\gamma^\star)+I_{ws}\Omega\omega_v^\star sin(\gamma^\star)+2(I_{ws}-I_{wt})\omega_u^\star\omega_v^\star sin(2\gamma^\star}$$

(B.3)

$$\frac{M_u}{\omega_v} = -\omega_w^\star(I_{gg} - I_{gs} - I_{ws}sin(\gamma^\star)^2 + I_{wt}sin(\gamma^\star)^2)$$

$$+\frac{\omega_w^\star(I_{ws}\omega_u^\star cos(2\gamma^\star)-I_{wt}\omega_u^\star cos(2\gamma^\star)+I_{ws}\omega_v^\star sin(2\gamma^\star)-I_{wt}\omega_v^\star sin(2\gamma^\star)+I_{ws}\Omega cos(\gamma^\star))^2}{k+bs+J_g s^2+(I_{ws}-I_{wt})\omega_u^{\star 2}cos(2\gamma^\star)+(I_{wt}-I_{ws})\omega_v^{\star 2}cos(2\gamma^\star)+I_{ws}\Omega\omega_u^\star cos(\gamma^\star)+I_{ws}\Omega\omega_v^\star sin(\gamma^\star)+2(I_{ws}-I_{wt})\omega_u^\star\omega_v^\star sin(2\gamma^\star}$$

(B.4)

$$-\frac{s\omega_v^\star(I_{ws}\omega_u^\star cos(2\gamma^\star)-I_{wt}\omega_u^\star cos(2\gamma^\star)+I_{ws}\omega_v^\star sin(2\gamma^\star)-I_{wt}\omega_v^\star sin(2\gamma^\star)+I_{ws}\Omega cos(\gamma^\star))}{k+bs+J_g s^2+(I_{ws}-I_{wt})\omega_u^{\star 2}cos(2\gamma^\star)+(I_{wt}-I_{ws})\omega_v^{\star 2}cos(2\gamma^\star)+I_{ws}\Omega\omega_u^\star cos(\gamma^\star)+I_{ws}\Omega\omega_v^\star sin(\gamma^\star)+2(I_{ws}-I_{wt})\omega_u^\star\omega_v^\star sin(2\gamma^\star)}$$

$$\frac{M_v}{\omega_v} = -\omega_w^\star sin(2\gamma^\star)(I_{ws} - I_{wt})/2 - s(J_t + I_{ws}sin(\gamma^\star)^2 - I_{wt}sin(\gamma^\star)^2)$$

$$\frac{s\omega_u^\star(I_{ws}-I_{wt})\omega_u^\star cos(2\gamma^\star)+(I_{ws}-I_{wt})\omega_v^\star sin(2\gamma^\star)+I_{ws}\Omega cos(\gamma^\star)}{k+bs+J_g s^2+(I_{ws}-I_{wt})\omega_u^{\star 2}cos(2\gamma^\star)+(I_{wt}-I_{ws})\omega_v^{\star 2}cos(2\gamma^\star)+I_{ws}\Omega\omega_u^\star cos(\gamma^\star)+I_{ws}\Omega\omega_v^\star sin(\gamma^\star)+2(I_{ws}-I_{wt})\omega_u^\star\omega_v^\star sin(2\gamma^\star)}$$

$$+\frac{\omega_w^\star[(I_{ws}-I_{wt})\omega_u^\star cos(2\gamma^\star)+(I_{ws}-I_{wt})\omega_v^\star sin(2\gamma^\star)+I_{ws}\Omega cos(\gamma^\star)][(I_{wt}-I_{ws})\omega_v^\star cos(2\gamma^\star)+(I_{ws}-I_{wt})\omega_u^\star sin(2\gamma^\star)+I_{ws}\Omega sin(\gamma^\star)]}{k+bs+J_g s^2+(I_{ws}-I_{wt})\omega_u^{\star 2}cos(2\gamma^\star)+(I_{wt}-I_{ws})\omega_v^{\star 2}cos(2\gamma^\star)+I_{ws}\Omega\omega_u^\star cos(\gamma^\star)+I_{ws}\Omega\omega_v^\star sin(\gamma^\star)+2(I_{ws}-I_{wt})\omega_u^\star\omega_v^\star sin(2\gamma^\star)}$$

(B.5)

$$\frac{M_w}{\omega_v} = \frac{(k+bs)(I_{ws}\omega_u^\star \cos(2\gamma^\star)-I_{wt}\omega_u^\star \cos(2\gamma^\star)+I_{ws}\omega_v^\star \sin(2\gamma^\star)-I_{wt}\omega_v^\star \sin(2\gamma^\star)+I_{ws}\Omega \cos(\gamma^\star))}{k+bs+J_g s^2+(I_{ws}-I_{wt})\omega_u^{\star 2}\cos(2\gamma^\star)+(I_{wt}-I_{ws})\omega_v^{\star 2}\cos(2\gamma^\star)+I_{ws}\Omega\omega_u^\star \cos(\gamma^\star)+I_{ws}\Omega\omega_v^\star \sin(\gamma^\star)+2(I_{ws}-I_{wt})\omega_u^\star\omega_v^\star \sin(2\gamma^\star)} \tag{B.6}$$

$$\frac{M_u}{\omega_w} = -\frac{2\omega_u^\star \sin(2\gamma^\star)(I_{ws}-I_{wt})(k+bs)}{k+bs+J_g s^2+(I_{ws}-I_{wt})\omega_u^{\star 2}\cos(2\gamma^\star)+(I_{wt}-I_{ws})\omega_v^{\star 2}\cos(2\gamma^\star)+I_{ws}\Omega\omega_u^\star \cos(\gamma^\star)+I_{ws}\Omega\omega_v^\star \sin(\gamma^\star)+2(I_{ws}-I_{wt})\omega_u^\star\omega_v^\star \sin(2\gamma^\star} \tag{B.7}$$

$$\begin{aligned}\frac{M_v}{\omega_w} &= I_{gg}\omega_u^\star - I_{gs}\omega_u^\star - I_{ws}\omega_u^\star \cos(\gamma^\star)^2 + I_{wt}\omega_u^\star \cos(\gamma^\star)^2 - (I_{ws}\omega_v^\star \sin(2\gamma^\star))/2 + (I_{wt}\omega_v^\star \sin(2\gamma^\star))/2 - I_{ws}\Omega \cos(\gamma^\star) \\ &- \frac{s\omega_w^\star(I_{wt}\omega_v^\star \cos(2\gamma^\star)-I_{ws}\omega_v^\star \cos(2\gamma^\star)+I_{ws}\omega_u^\star \sin(2\gamma^\star)-I_{wt}\omega_u^\star \sin(2\gamma^\star)+I_{ws}\Omega \sin(\gamma^\star))}{k+bs+J_g s^2+(I_{ws}-I_{wt})\omega_u^{\star 2}\cos(2\gamma^\star)+(I_{wt}-I_{ws})\omega_v^{\star 2}\cos(2\gamma^\star)+I_{ws}\Omega\omega_u^\star \cos(\gamma^\star)+I_{ws}\Omega\omega_v^\star \sin(\gamma^\star)+2(I_{ws}-I_{wt})\omega_u^\star\omega_v^\star \sin(2\gamma^\star} \\ &- \frac{s^2\omega_u^\star J_g}{k+bs+J_g s^2+(I_{ws}-I_{wt})\omega_u^{\star 2}\cos(2\gamma^\star)+(I_{wt}-I_{ws})\omega_v^{\star 2}\cos(2\gamma^\star)+I_{ws}\Omega\omega_u^\star \cos(\gamma^\star)+I_{ws}\Omega\omega_v^\star \sin(\gamma^\star)+2(I_{ws}-I_{wt})\omega_u^\star\omega_v^\star \sin(2\gamma^\star)}\end{aligned} \tag{B.8}$$

$$\frac{M_w}{\omega_w} = \frac{-sJ_g(k+bs)}{k+bs+J_g s^2+(I_{ws}-I_{wt})\omega_u^{\star 2}\cos(2\gamma^\star)+(I_{wt}-I_{ws})\omega_v^{\star 2}\cos(2\gamma^\star)+I_{ws}\Omega\omega_u^\star \cos(\gamma^\star)+I_{ws}\Omega\omega_v^\star \sin(\gamma^\star)+2(I_{ws}-I_{wt})\omega_u^\star\omega_v^\star \sin(2\gamma^\star)} \tag{B.9}$$

B.2. Transmissibility of a single CMG in the body-fixed Frame

$$\frac{\gamma}{\omega_u} = -\frac{(J_t-J_s)\omega_v^\star \cos(2\gamma^\star)+(J_s-J_t)\omega_u^\star \sin(2\gamma^\star)+I_{ws}\Omega\sin(\gamma^\star)}{(k+bs+J_g s^2+(J_s-J_t)\omega_u^{\star 2}\cos 2\gamma^\star+(J_t-J_s)\omega_v^{\star 2}\cos 2\gamma^\star+I_{ws}\Omega\omega_u^\star\cos\gamma^\star+I_{ws}\Omega\omega_v^\star\sin\gamma^\star+2(J_s-J_t)\omega_u^\star\omega_v^\star\sin 2\gamma^\star)} \tag{B.10}$$

$$\frac{\gamma}{\omega_v} = \frac{(J_s-J_t)\omega_u^\star \cos(2\gamma^\star)+(J_s-J_t)\omega_v^\star \sin(2\gamma^\star)+I_{ws}\Omega\cos(\gamma^\star)}{(k+bs+J_g s^2+(J_s-J_t)\omega_u^{\star 2}\cos 2\gamma^\star+(J_t-J_s)\omega_v^{\star 2}\cos 2\gamma^\star+I_{ws}\Omega\omega_u^\star\cos\gamma^\star+I_{ws}\Omega\omega_v^\star\sin\gamma^\star+2(J_s-J_t)\omega_u^\star\omega_v^\star\sin 2\gamma^\star)} \tag{B.11}$$

$$\frac{\gamma}{\omega_w} = -\frac{s}{(k+bs+J_g s^2+(J_s-J_t)\omega_u^{\star 2}\cos 2\gamma^\star+(J_t-J_s)\omega_v^{\star 2}\cos 2\gamma^\star+I_{ws}\Omega\omega_u^\star\cos\gamma^\star+I_{ws}\Omega\omega_v^\star\sin\gamma^\star+2(J_s-J_t)\omega_u^\star\omega_v^\star\sin 2\gamma^\star)} \tag{B.12}$$

B.3. Impedance of a Scissored Pair CMG

$$\begin{aligned}\frac{M_u}{\omega_u} &= -s\cos(\gamma^\star)2J_s \\ &- \frac{s\omega_u^\star\sin(2\gamma^\star)(I_{ws}-I_{wt})[2(I_{ws}-I_{wt})\omega_v^\star\cos(\gamma^\star)+2I_{gg}\omega_u^\star\sin(\gamma^\star)+2I_{ws}\omega_u^\star\sin(\gamma^\star)]}{k+bs+J_g s^2+(I_{ws}-I_{wt})\omega_u^{\star 2}\cos(2\gamma^\star)+(I_{wt}-I_{wt})\omega_v^{\star 2}\cos(2\gamma^\star)+I_{ws}\Omega\omega_v^\star\sin(\gamma^\star)} \\ &- \frac{2\omega_u^\star\omega_v^\star\omega_w^\star\sin(2\gamma^\star)\sin(\gamma^\star)(I_{gg}-I_{gs})(I_{ws}-I_{wt})}{k+bs+J_g s^2+(I_{ws}-I_{wt})\omega_u^{\star 2}\cos(2\gamma^\star)+(I_{wt}-I_{wt})\omega_v^{\star 2}\cos(2\gamma^\star)+I_{ws}\Omega\omega_v^\star\sin(\gamma^\star)}\end{aligned} \tag{B.13}$$

$$\begin{aligned}\frac{M_u}{\omega_v} &= -2\omega_w^\star\cos(\gamma^\star)(I_{gg}-I_{gs}) \\ &+ \frac{s(I_{ws}\omega_v^\star\sin(2\gamma^\star)-I_{wt}\omega_v^\star\sin(2\gamma^\star)+I_{ws}\Omega\cos(\gamma^\star))(2I_{ws}\omega_v^\star\cos(\gamma^\star)-2I_{wt}\omega_v^\star\cos(\gamma^\star)+2I_{gg}\omega_u^\star\sin(\gamma^\star)+2I_{ws}\omega_u^\star\sin(\gamma^\star))}{k+bs+J_g s^2+(I_{ws}-I_{wt})\omega_u^{\star 2}\cos(2\gamma^\star)+(I_{wt}-I_{wt})\omega_v^{\star 2}\cos(2\gamma^\star)+I_{ws}\Omega\omega_v^\star\sin(\gamma^\star)} \\ &+ \frac{2\omega_v^\star\omega_w^\star\sin(\gamma^\star)(I_{gg}-I_{gs})(I_{ws}\omega_v^\star\sin(2\gamma^\star)-I_{wt}\omega_v^\star\sin(2\gamma^\star)+I_{ws}\Omega\cos(\gamma^\star))}{k+bs+J_g s^2+(I_{ws}-I_{wt})\omega_u^{\star 2}\cos(2\gamma^\star)+(I_{wt}-I_{wt})\omega_v^{\star 2}\cos(2\gamma^\star)+I_{ws}\Omega\omega_v^\star\sin(\gamma^\star)}\end{aligned} \tag{B.14}$$

$$\frac{M_u}{\omega_w} = -2\omega_v^\star \cos\left(\gamma^\star\right)(I_{gg} - I_{gs}) \tag{B.15}$$

$$\begin{aligned}\frac{M_v}{\omega_u} &= 2\omega_w^\star \cos\left(\gamma^\star\right)(J_g - J_s)\\ &+ \frac{s\omega_u^\star \sin(2\gamma^\star)(I_{ws}-I_{wt})(2I_{ws}\Omega-2I_{ws}\omega_u^\star\cos(\gamma^\star)+2I_{wt}\omega_u^\star\cos(\gamma^\star)-2I_{gg}\omega_v^\star\sin(\gamma^\star)+2I_{ws}\omega_v^\star\sin(\gamma^\star)-4I_{wt}\omega_v^\star\sin(\gamma^\star))}{k+bs+J_g s^2+(I_{ws}-I_{wt})\omega_u^{\star 2}\cos(2\gamma^\star)+(I_{wt}-I_{wt})\omega_v^{\star 2}\cos(2\gamma^\star)+I_{ws}\Omega\omega_v^\star\sin(\gamma^\star)}\\ &+ \frac{2\omega_u^{\star 2}\omega_w^\star\sin(2\gamma^\star)\sin(\gamma^\star)(I_{ws}-I_{wt})(J_g-J_s)}{k+bs+J_g s^2+(I_{ws}-I_{wt})\omega_u^{\star 2}\cos(2\gamma^\star)+(I_{wt}-I_{wt})\omega_v^{\star 2}\cos(2\gamma^\star)+I_{ws}\Omega\omega_v^\star\sin(\gamma^\star)}\end{aligned} \tag{B.16}$$

$$\begin{aligned}\frac{M_v}{\omega_v} &= -s\cos\left(\gamma^\star\right)(2J_t)\\ &- \frac{s[(I_{ws}-I_{wt})\omega_v^\star\sin(2\gamma^\star)+I_{ws}\Omega\cos(\gamma^\star)][2I_{ws}\Omega+2(I_{wt}-I_{ws})\omega_u^\star\cos(\gamma^\star)+2(I_{ws}-I_{gg}-2I_{wt})\omega_v^\star\sin(\gamma^\star)]}{k+bs+J_g s^2+(I_{ws}-I_{wt})\omega_u^{\star 2}\cos(2\gamma^\star)+(I_{wt}-I_{wt})\omega_v^{\star 2}\cos(2\gamma^\star)+I_{ws}\Omega\omega_v^\star\sin(\gamma^\star)}\\ &- \frac{2\omega_u^\star\omega_w^\star\sin(\gamma^\star)[I_{ws}\omega_v^\star\sin(2\gamma^\star)-I_{wt}\omega_v^\star\sin(2\gamma^\star)+I_{ws}\Omega\cos(\gamma^\star)](J_g-J_s)}{k+bs+J_g s^2+(I_{ws}-I_{wt})\omega_u^{\star 2}\cos(2\gamma^\star)+(I_{wt}-I_{wt})\omega_v^{\star 2}\cos(2\gamma^\star)+I_{ws}\Omega\omega_v^\star\sin(\gamma^\star)}\end{aligned} \tag{B.17}$$

$$\frac{M_v}{\omega_w} = 2\omega_u^\star \cos\left(\gamma^\star\right)(J_g - J_s) \tag{B.18}$$

$$\frac{M_w}{\omega_u} = -\frac{2\omega_u^\star\sin(2\gamma^\star)(I_{ws}-I_{wt})(k+bs)}{k+bs+J_g s^2+(I_{ws}-I_{wt})\omega_u^{\star 2}\cos(2\gamma^\star)+(I_{wt}-I_{wt})\omega_v^{\star 2}\cos(2\gamma^\star)+I_{ws}\Omega\omega_v^\star\sin(\gamma^\star)} \tag{B.19}$$

$$\frac{M_w}{\omega_v} = \frac{(2k+2bs)(I_{ws}\omega_v^\star\sin(2\gamma^\star)-I_{wt}\omega_v^\star\sin(2\gamma^\star)+I_{ws}\Omega\cos(\gamma^\star))}{k+bs+J_g s^2+(I_{ws}-I_{wt})\omega_u^{\star 2}\cos(2\gamma^\star)+(I_{wt}-I_{wt})\omega_v^{\star 2}\cos(2\gamma^\star)+I_{ws}\Omega\omega_v^\star\sin(\gamma^\star)} \tag{B.20}$$

$$\frac{M_w}{\omega_w} = \nexists \tag{B.21}$$

B.4. Transmissability of Scissored Pair Gyros

The transmissability describes the response of γ with a perturbation

$$\frac{\gamma}{\omega_u} = -\frac{\omega_u^\star\sin(2\gamma^\star)(J_s-J_t)+I_{ws}\Omega\sin(\gamma^\star)}{k+bs+J_g s^2+J_s-J_t\omega_u^{\star 2}\cos 2\gamma^\star+(J_t-J_s)\omega_v^{\star 2}\cos 2\gamma^\star+I_{ws}\Omega\omega_u^\star\cos\gamma^\star} \tag{B.22}$$

$$\frac{\gamma}{\omega_v} = \frac{\omega_v^\star\sin(2\gamma^\star)(J_s-J_t)}{k+bs+J_g s^2+J_s-J_t\omega_u^{\star 2}\cos 2\gamma^\star+(J_t-J_s)\omega_v^{\star 2}\cos 2\gamma^\star+I_{ws}\Omega\omega_u^\star\cos\gamma^\star} \tag{B.23}$$

$$\frac{\gamma}{\omega_w} = \nexists \tag{B.24}$$

C

Appendix C

C.1. Frequency Response of a Single CMG with realistic parameters

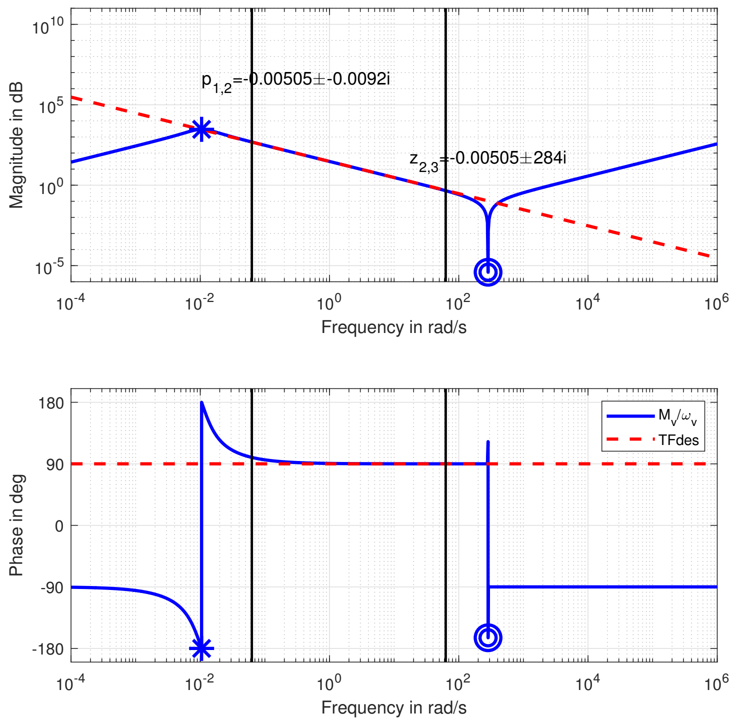


Figure C.1: Frequency response of a CMG when it was optimized to simulate a spring.

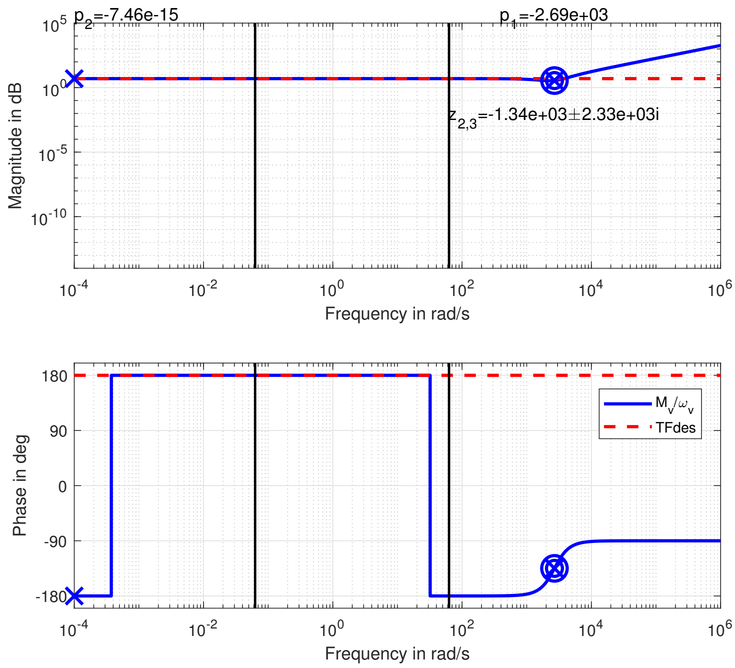


Figure C.2: Frequency response of a CMG when it was optimized to simulate a damper.

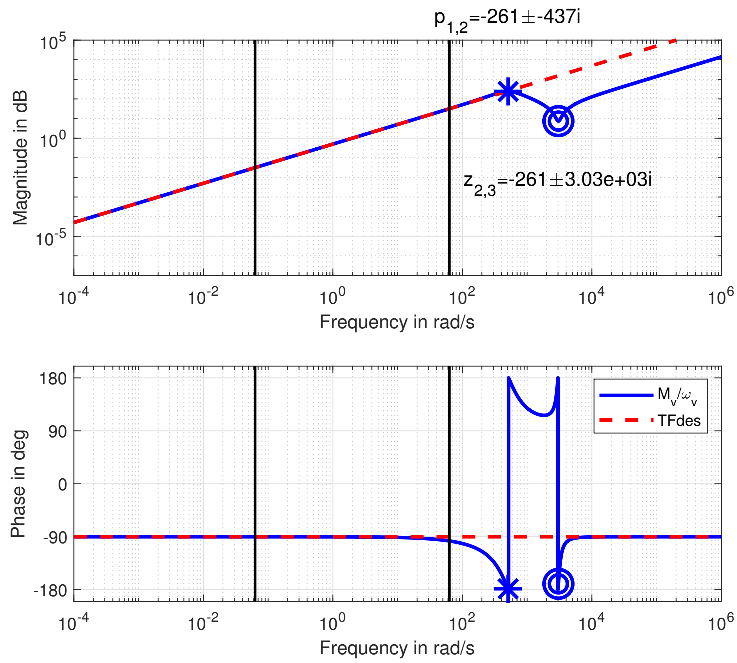


Figure C.3: Frequency response of a CMG when it was optimized to simulate a mass.

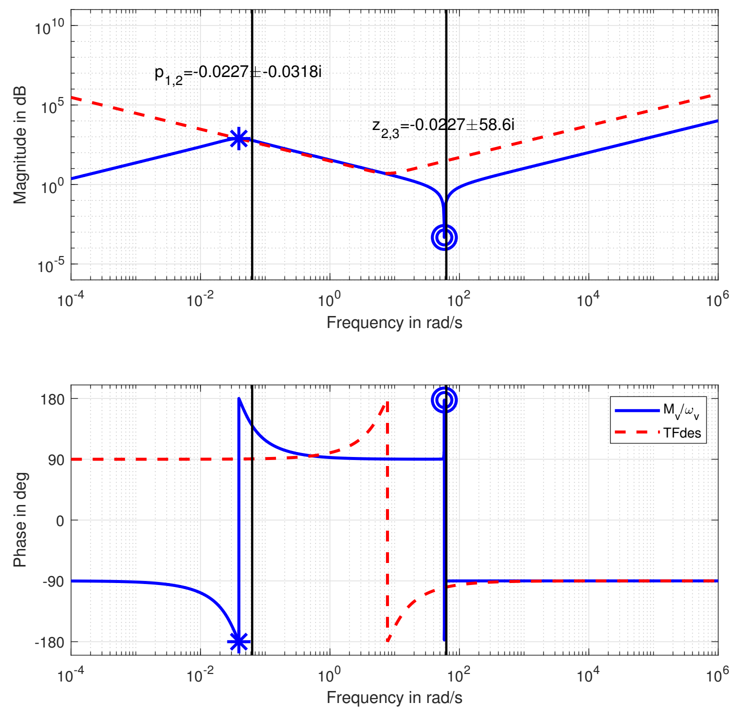


Figure C.4: Frequency response of a CMG when it was optimized to simulate a mass spring damper system.

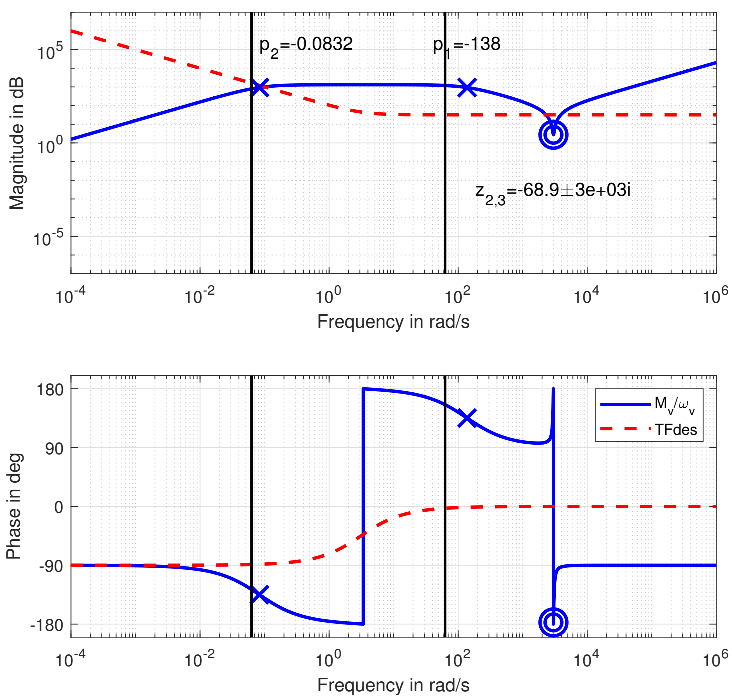


Figure C.5: Frequency response of a CMG when it was optimized to simulate the XCoM.

C.2. Frequency Response of a SPCMG with realistic parameters

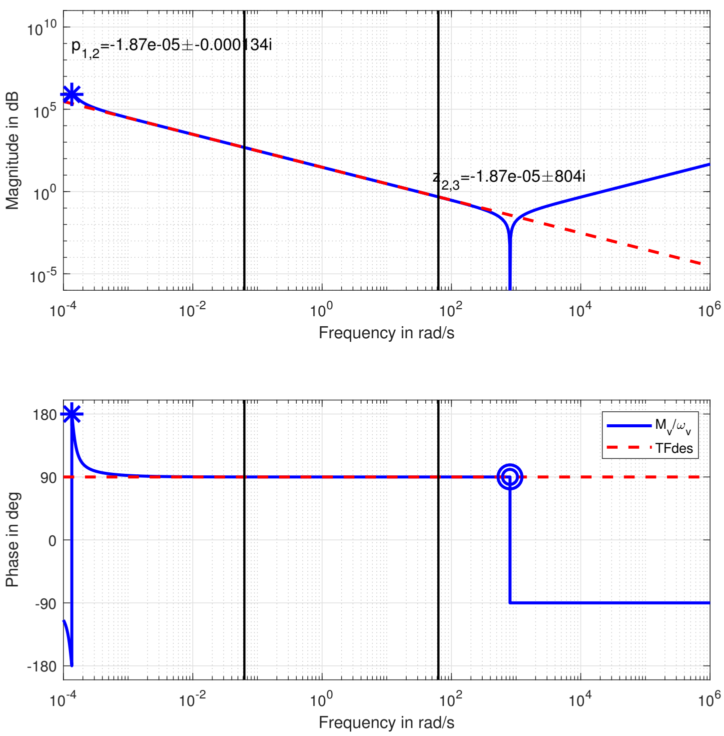


Figure C.6: Frequency response of a SPCMG when it was optimized to simulate a spring.

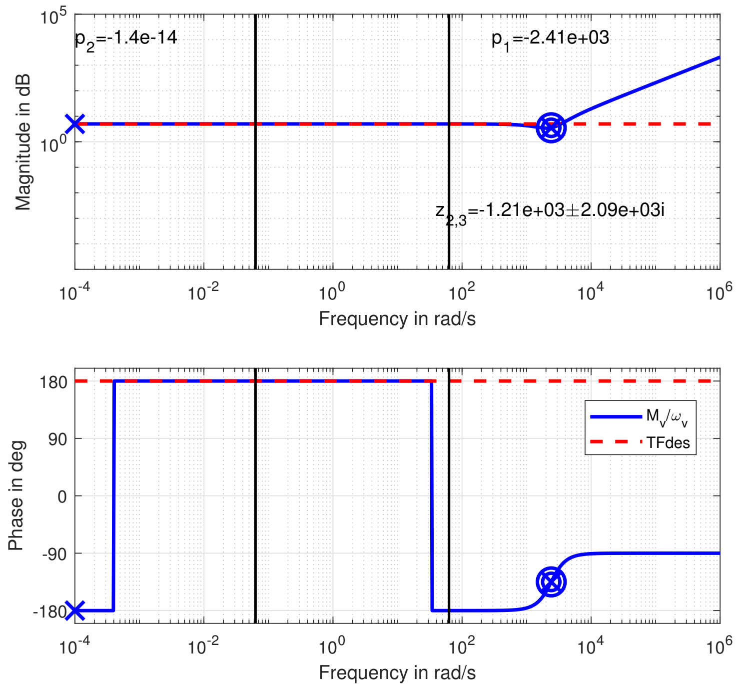


Figure C.7: Frequency response of a SPCMG when it was optimized to simulate a damper.

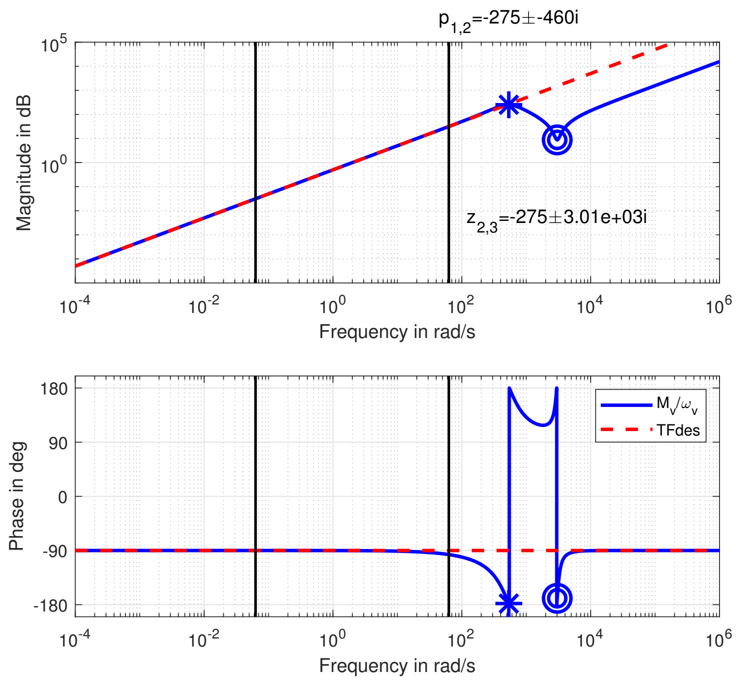


Figure C.8: Frequency response of a SPCMG when it was optimized to simulate a mass.

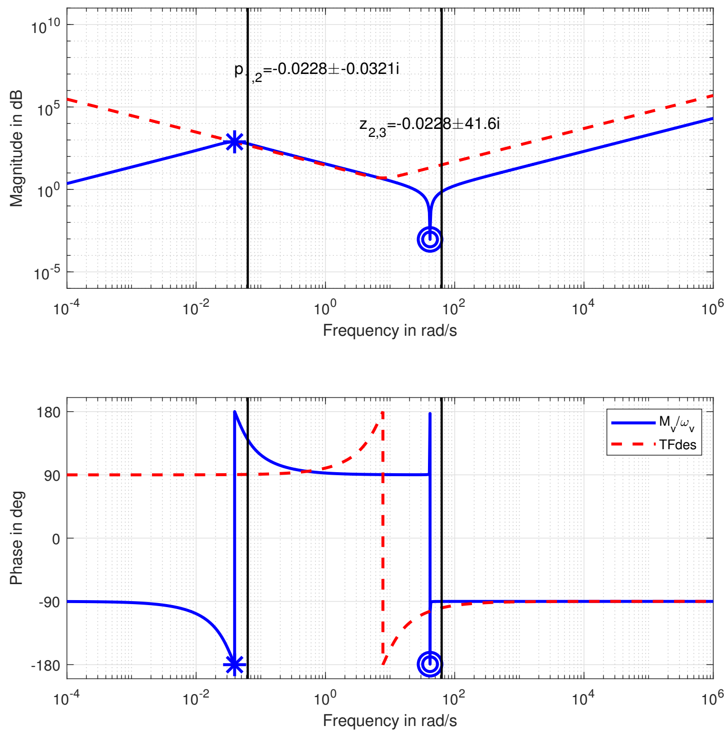


Figure C.9: Frequency response of a SPCMG when it was optimized to simulate a mass spring damper system.

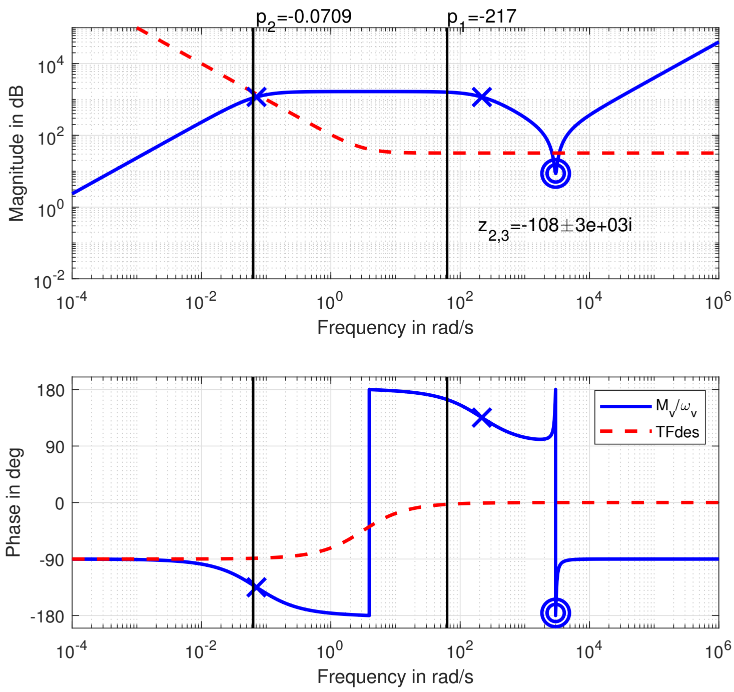


Figure C.10: Frequency response of a SPCMG when it was optimized to simulate the XCoM.

D

Appendix D

D.1. Time Response CMG

D.1.1. Spring

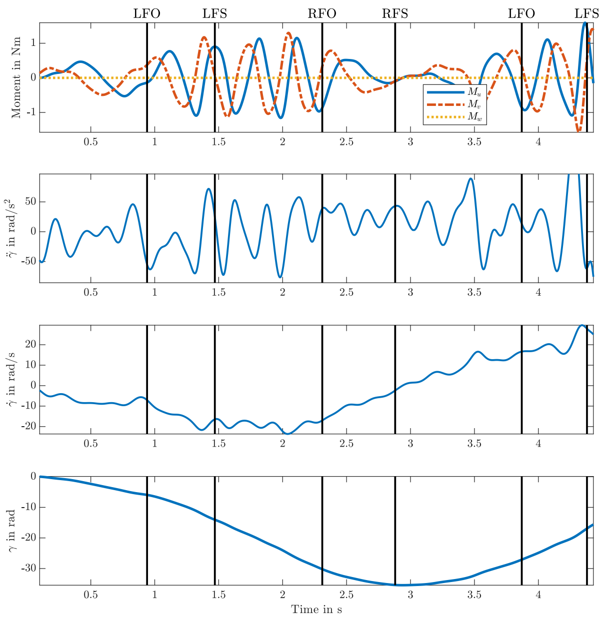


Figure D.1: Time response of a single CMG when optimized for a spring without bounds. The walking speed was between 0 - 0.4 m/s

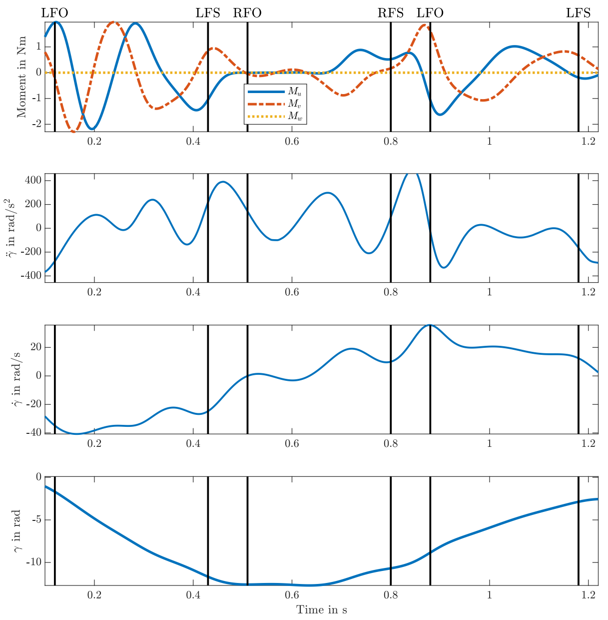


Figure D.2: Time response of a single CMG when optimized for a spring without bounds. The walking speed was a self selected fast walking speed between 1.9 - 2.2 m/s

D.1.2. Damper

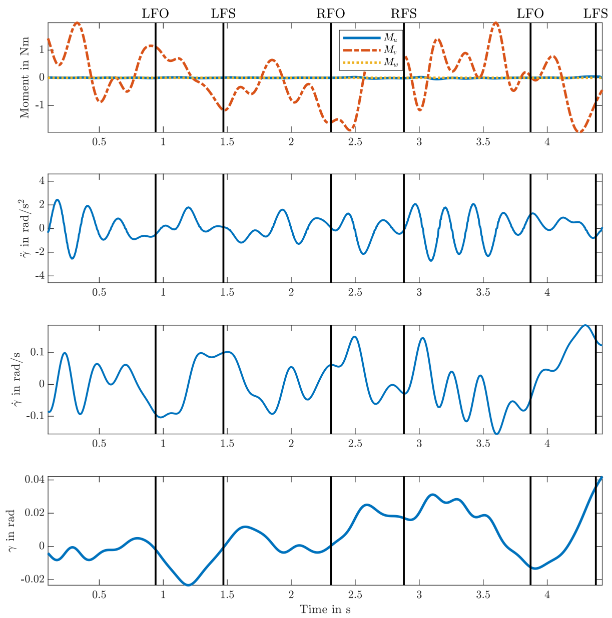


Figure D.3: Time response of a single CMG when optimized for a damper without bounds. The walking speed was between 0 - 0.4 m/s

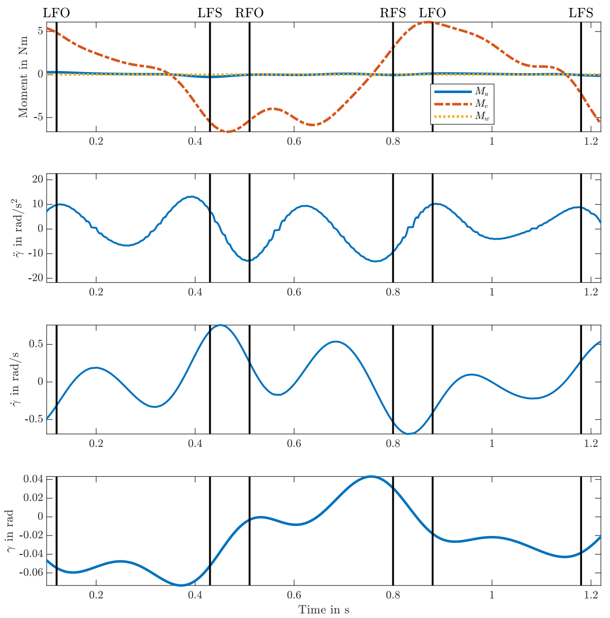


Figure D.4: Time response of a single CMG when optimized for a damper without bounds. The walking speed was a self selected fast walking speed between 1.9 - 2.2 m/s

D.1.3. Mass

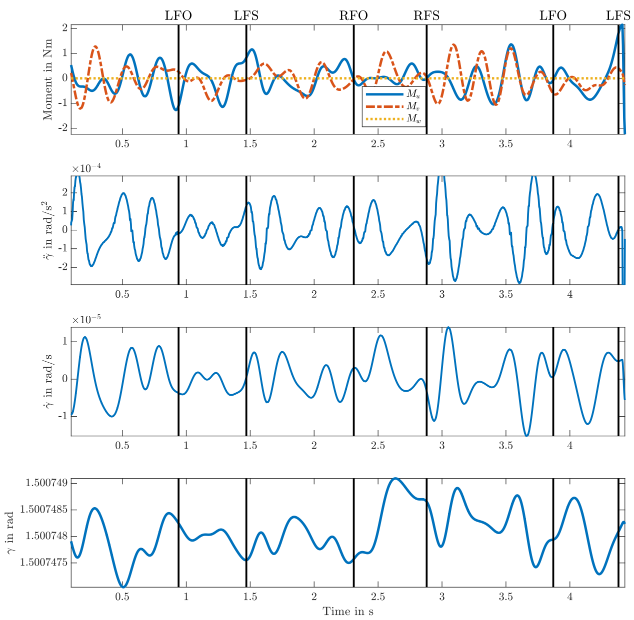


Figure D.5: Time response of a single CMG when optimized for a mass without bounds. The walking speed was between 0 - 0.4 m/s

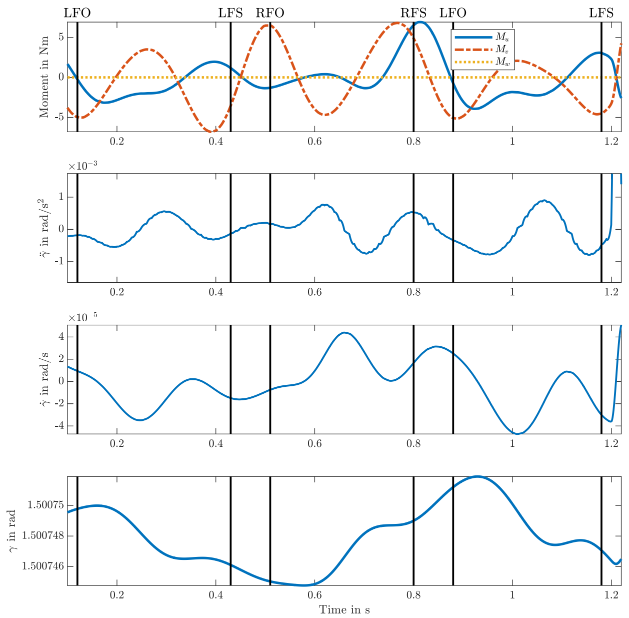


Figure D.6: Time response of a single CMG when optimized for a mass without bounds. The walking speed was a self selected fast walking speed between 1.9 - 2.2 m/s

D.1.4. Mass Spring Damper

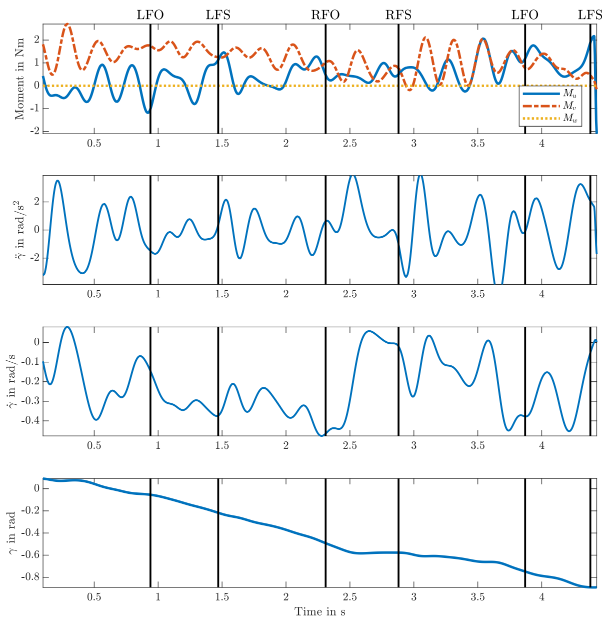


Figure D.7: Time response of a single CMG when optimized for a mass spring damper system without bounds. The walking speed was between 0 - 0.4 m/s

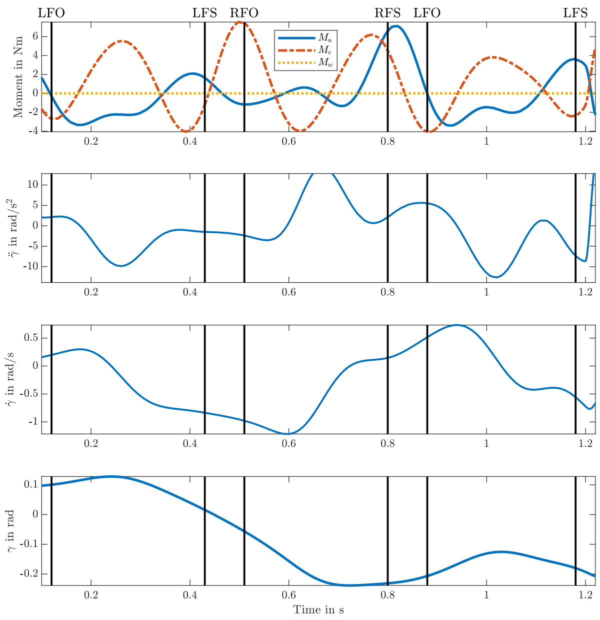


Figure D.8: Time response of a single CMG when optimized for a mass spring damper system without bounds. The walking speed was a self selected fast walking speed between 1.9 - 2.2 m/s

D.1.5. PDXCoM

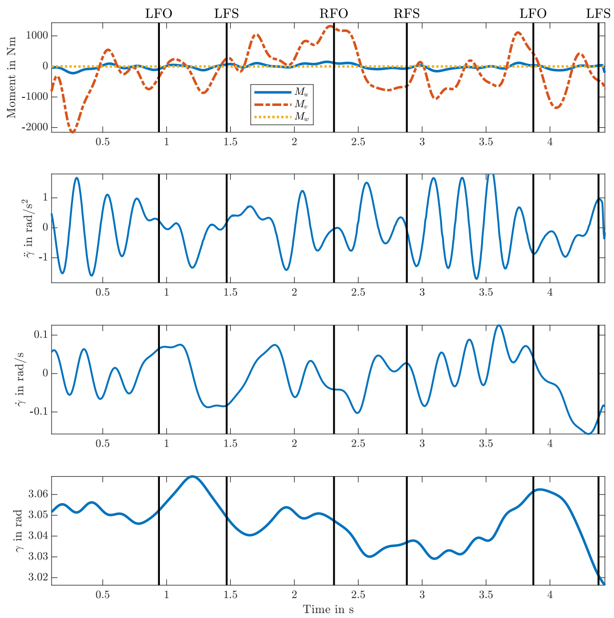


Figure D.9: Time response of a single CMG when optimized for PDXCoM without bounds. The walking speed was between 0 - 0.4 m/s

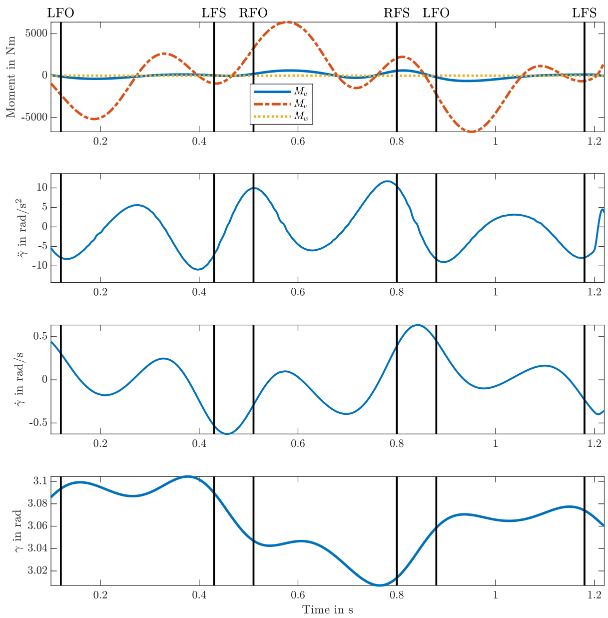


Figure D.10: Time response of a single CMG when optimized for PDXCoM without bounds. The walking speed was a self selected fast walking speed between 1.9 - 2.2 m/s

D.2. CMG with realistic parameters

D.2.1. Spring

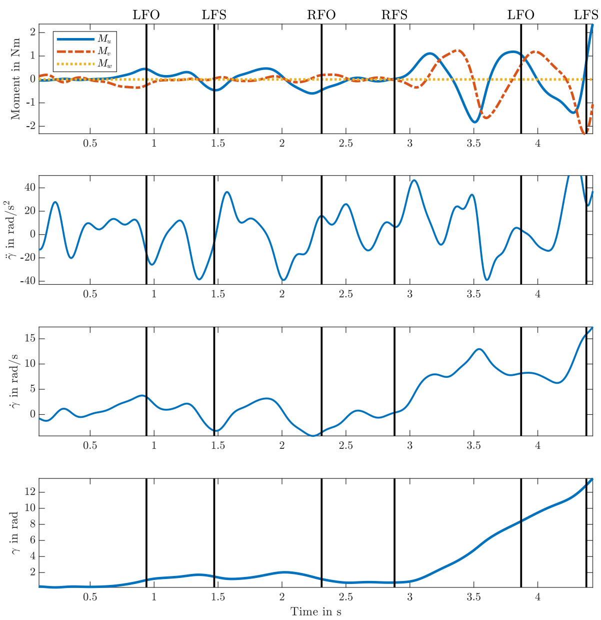


Figure D.11: Time response of a single CMG when optimized for a spring with realistic bounds. The walking speed was between 0 - 0.4 m/s

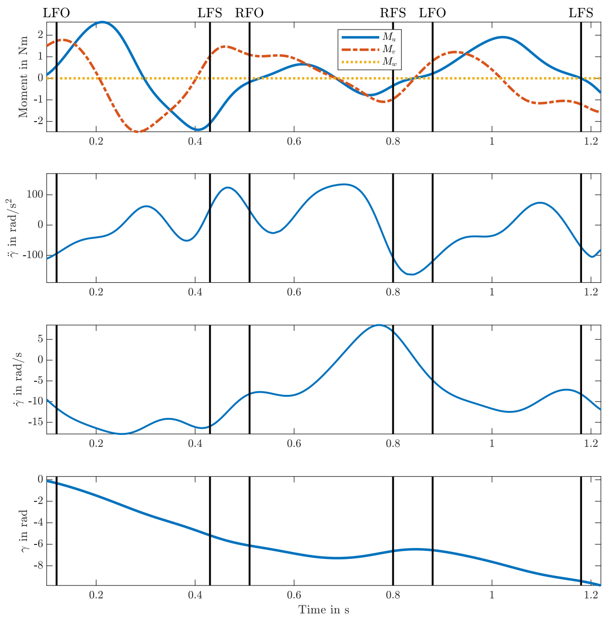


Figure D.12: Time response of a single CMG when optimized for a spring with realistic bounds. The walking speed a self selected fast speed between 1.9 - 2.2 m/s

D.2.2. Damper

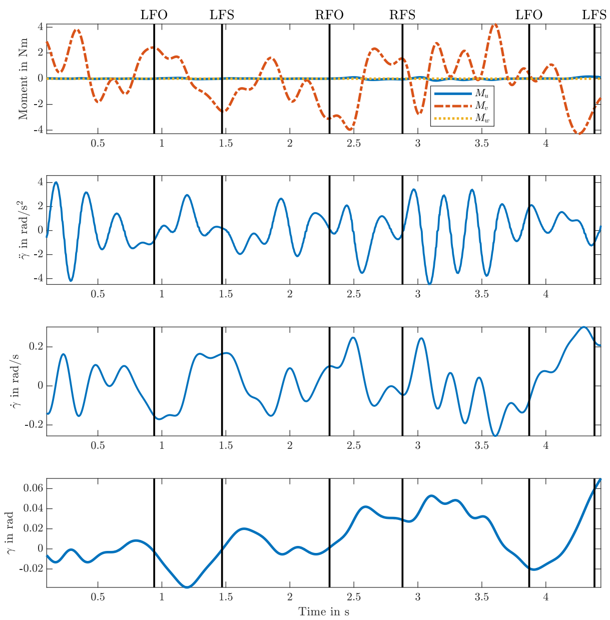


Figure D.13: Time response of a single CMG when optimized for a Damper with realistic bounds. The walking speed was between 0 - 0.4 m/s

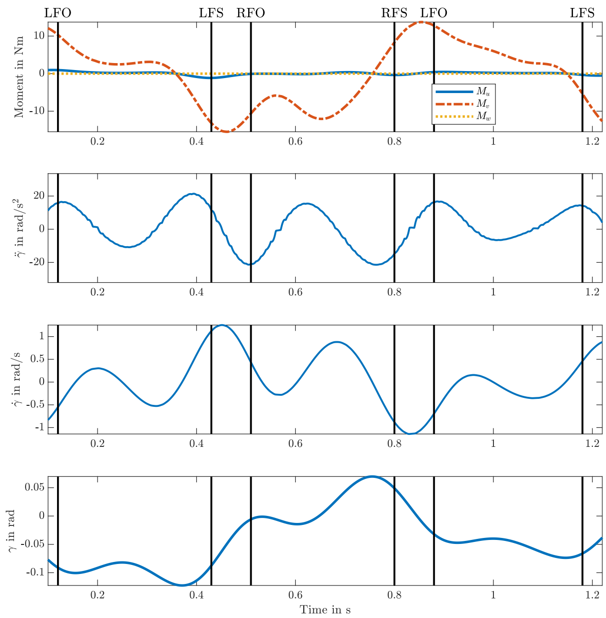


Figure D.14: Time response of a single CMG when optimized for a Damper with realistic bounds. The walking speed a self selected fast speed between 1.9 - 2.2 m/s

D.2.3. Mass

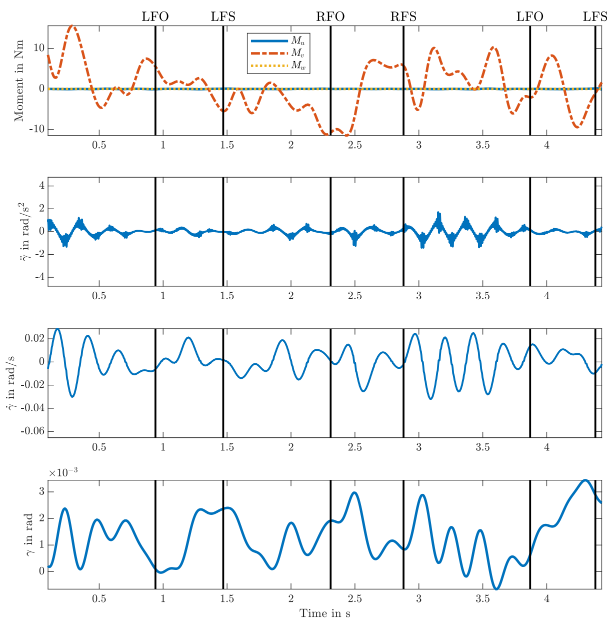


Figure D.15: Time response of a single CMG when optimized for a mass with realistic bounds. The walking speed was between 0 - 0.4 m/s

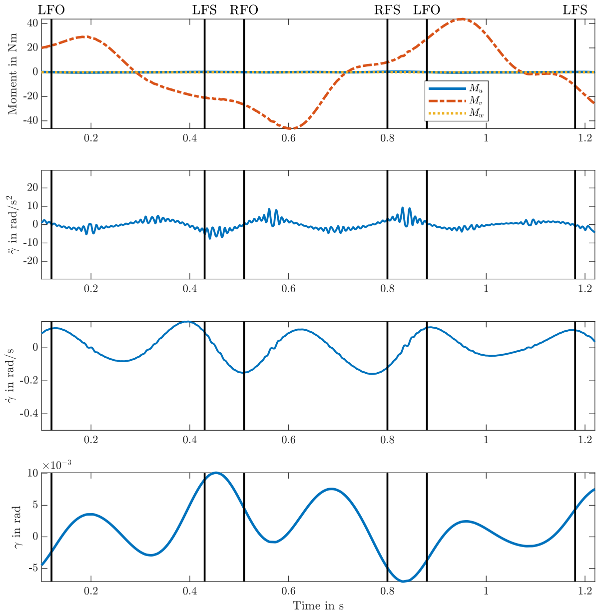


Figure D.16: Time response of a single CMG when optimized for a mass with realistic bounds. The walking speed a self selected fast speed between 1.9 - 2.2 m/s

D.2.4. Mass Spring Damper

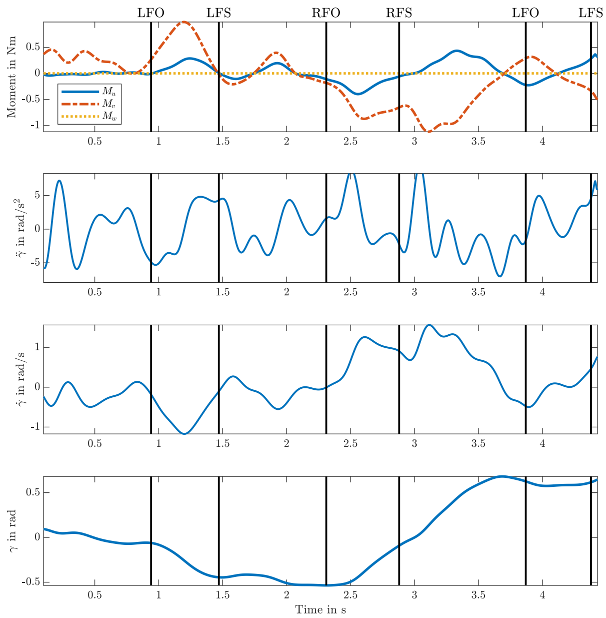


Figure D.17: Time response of a single CMG when optimized for a mass spring damper system with realistic bounds. The walking speed was between 0 - 0.4 m/s

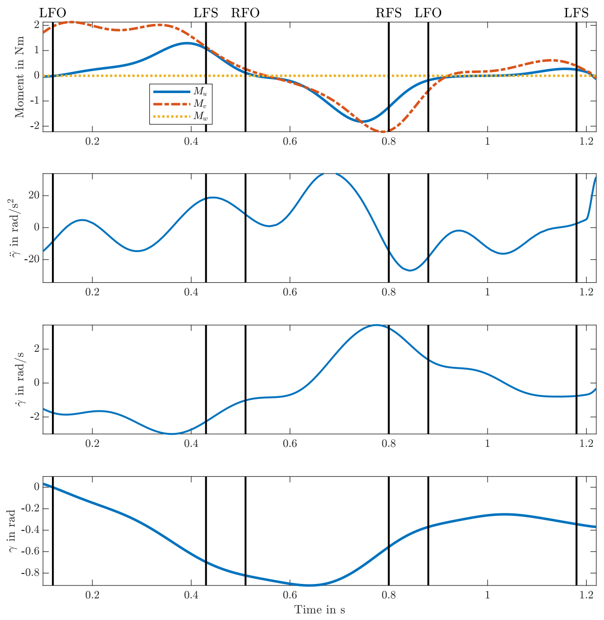


Figure D.18: Time response of a single CMG when optimized for a mass spring damper system with realistic bounds. The walking speed a self selected fast speed between 1.9 - 2.2 m/s

D.2.5. PDXCoM

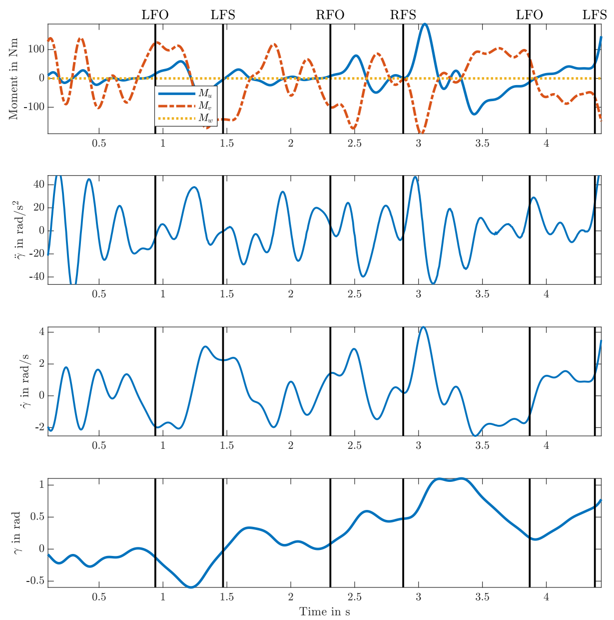


Figure D.19: Time response of a single CMG when optimized for the PDXCoM system with realistic bounds. The walking speed was between 0 - 0.4 m/s

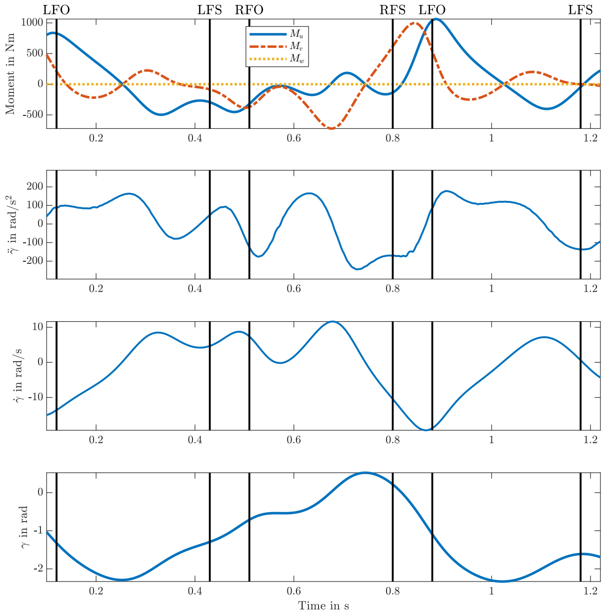


Figure D.20: Time response of a single CMG when optimized for the PDXCoM system with realistic bounds. The walking speed a self selected fast speed between 1.9 - 2.2 m/s

D.3. SPCMG without bounds

D.3.1. Mass

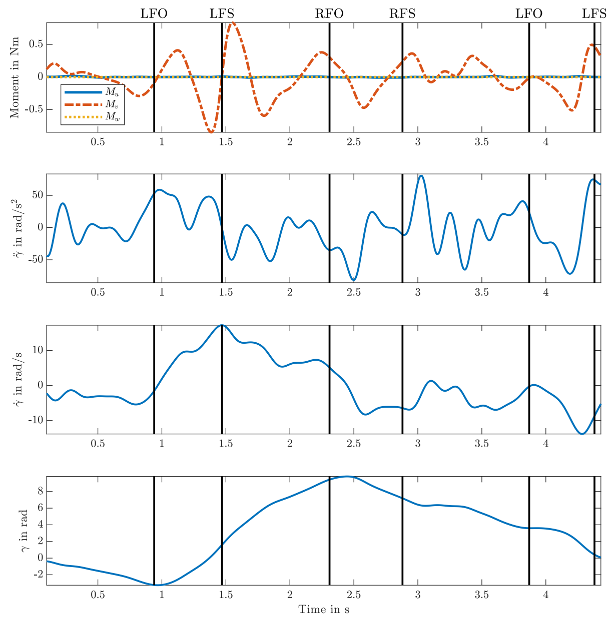


Figure D.21: Time response of a single SPCMG when optimized for a spring without bounds. The walking speed was between 0 - 0.4 m/s

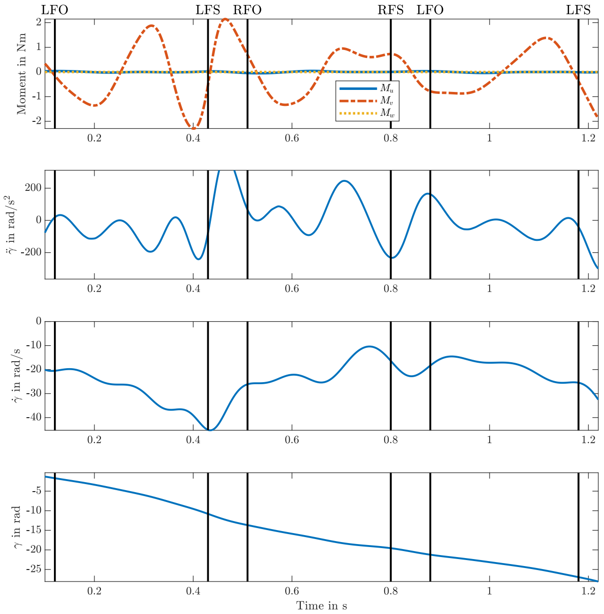


Figure D.22: Time response of a single SPCMG when optimized for a spring without bounds. The walking speed was a self selected fast walking speed between 1.9 - 2.2 m/s

D.3.2. Damper

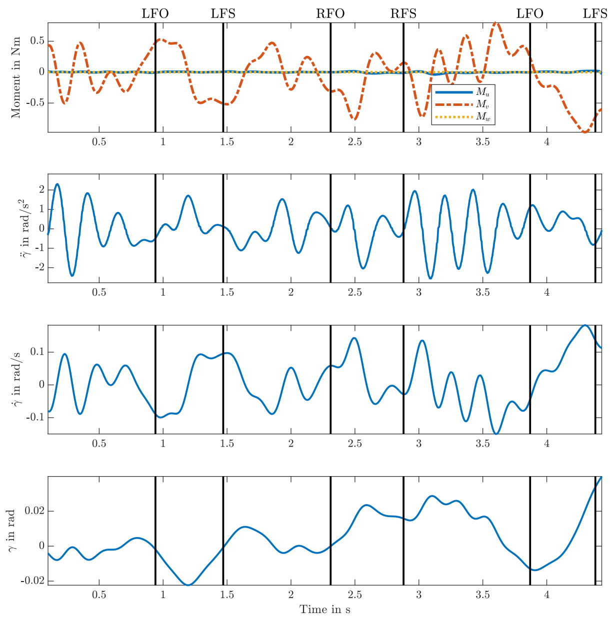


Figure D.23: Time response of a single SPCMG when optimized for a damper without bounds. The walking speed was between 0 - 0.4 m/s

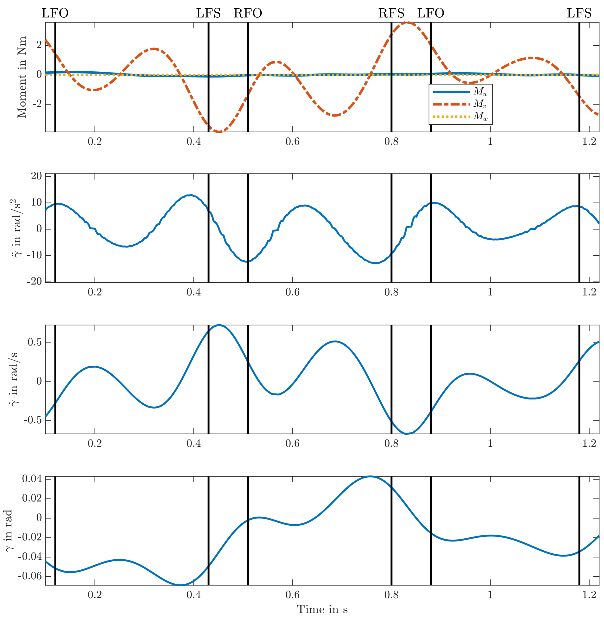


Figure D.24: Time response of a single SPCMG when optimized for a damper without bounds. The walking speed was a self selected fast walking speed between 1.9 - 2.2 m/s

D.3.3. Mass

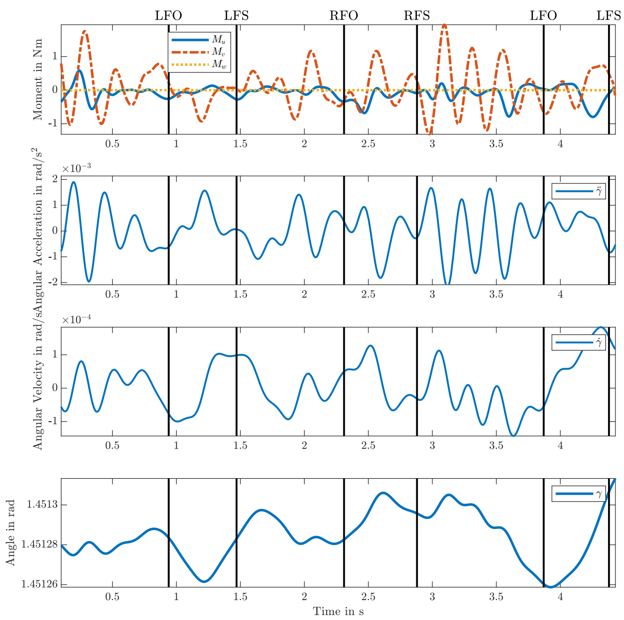


Figure D.25: Time response of a single SPCMG when optimized for a mass without bounds. The walking speed was between 0 - 0.4 m/s

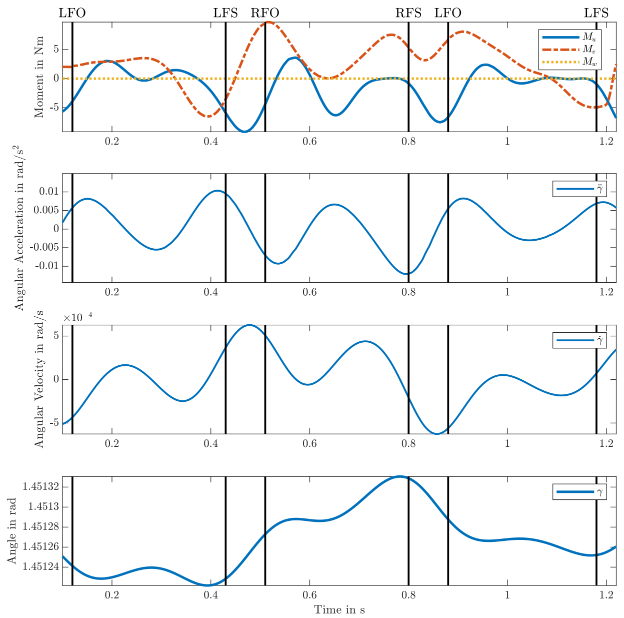


Figure D.26: Time response of a single SPCMG when optimized for a mass without bounds. The walking speed was a self selected fast walking speed between 1.9 - 2.2 m/s

D.3.4. Mass Spring Damper

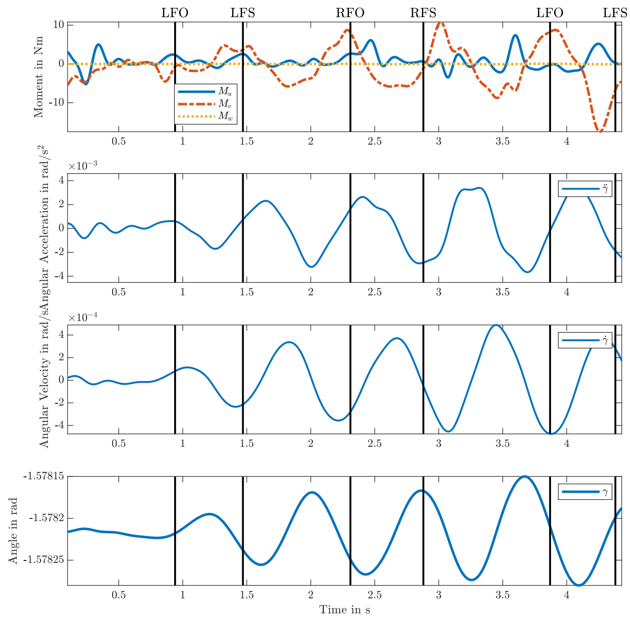


Figure D.27: Time response of a single SPCMG when optimized for a mass spring damper system without bounds. The walking speed was between 0 - 0.4 m/s

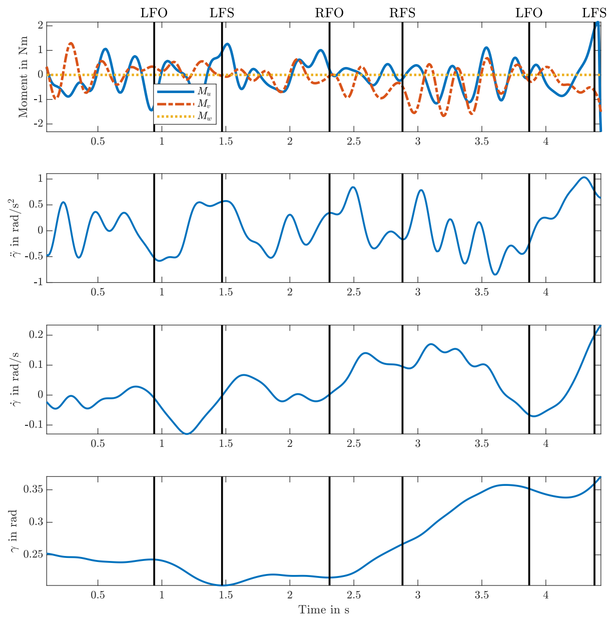


Figure D.28: Time response of a single SPCMG when optimized for a mass spring damper system without bounds. The walking speed was a self selected fast walking speed between 1.9 - 2.2 m/s

D.3.5. XCoM

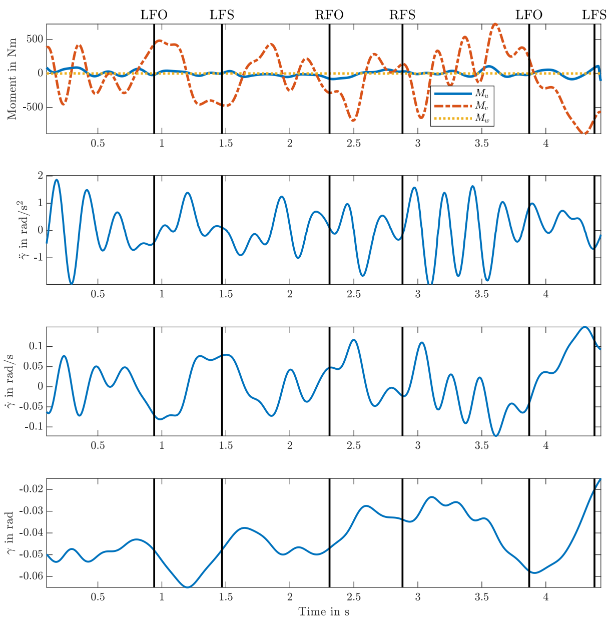


Figure D.29: Time response of a single SPCMG when optimized for the PDXCoM system without bounds. The walking speed was between 0 - 0.4 m/s

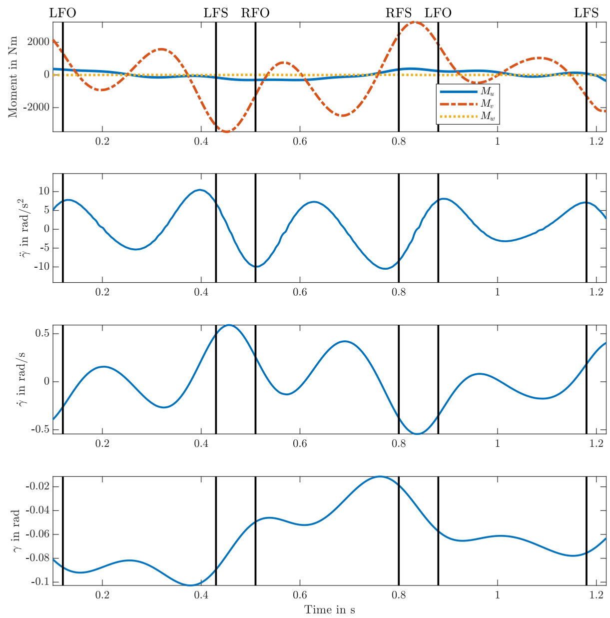


Figure D.30: Time response of a single SPCMG when optimized for the PDXCoM system without bounds. The walking speed was a self selected fast walking speed between 1.9 - 2.2 m/s

D.4. SPCMG with realistic bounds

D.4.1. Spring

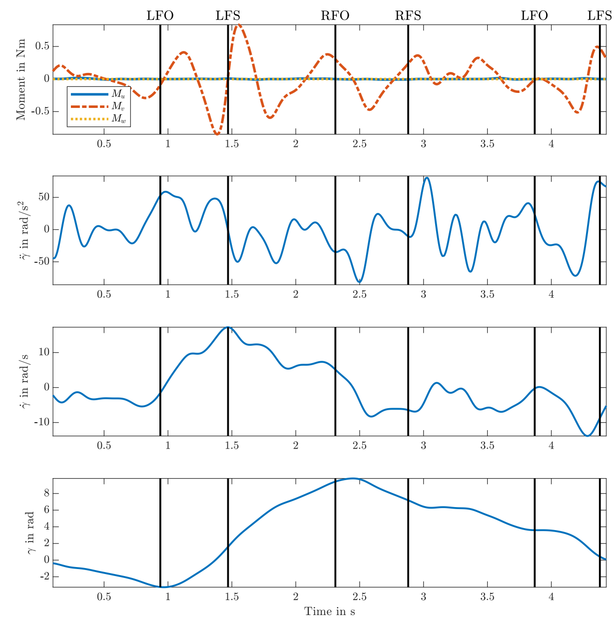


Figure D.31: Time response of a SPCMG when optimized for a spring with realistic bounds. The walking speed was between 0 - 0.4 m/s

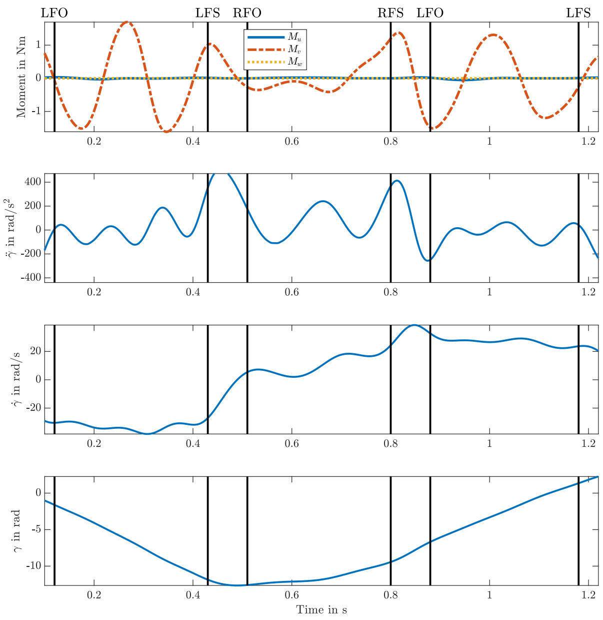


Figure D.32: Time response of a SPCMG when optimized for a spring with realistic bounds. The walking speed a self selected fast speed between 1.9 - 2.2 m/s

D.4.2. Damper

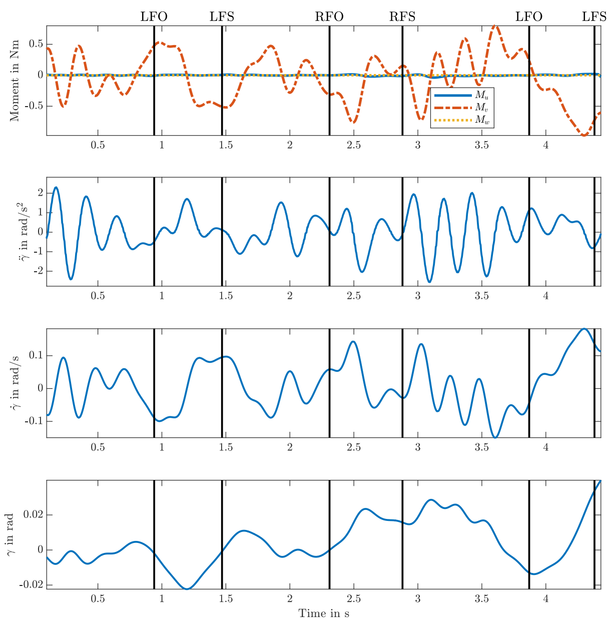


Figure D.33: Time response of a SPCMG when optimized for a damper with realistic bounds. The walking speed was between 0 - 0.4 m/s

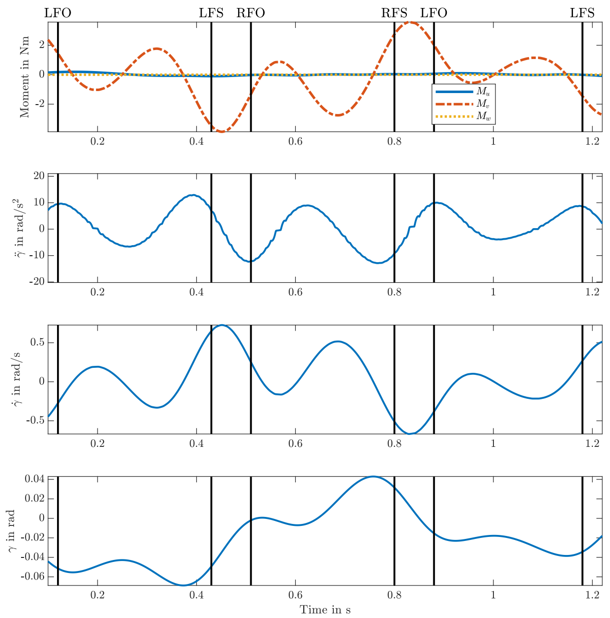


Figure D.34: Time response of a SPCMG when optimized for a damper with realistic bounds. The walking speed a self selected fast speed between 1.9 - 2.2 m/s

D.4.3. Mass

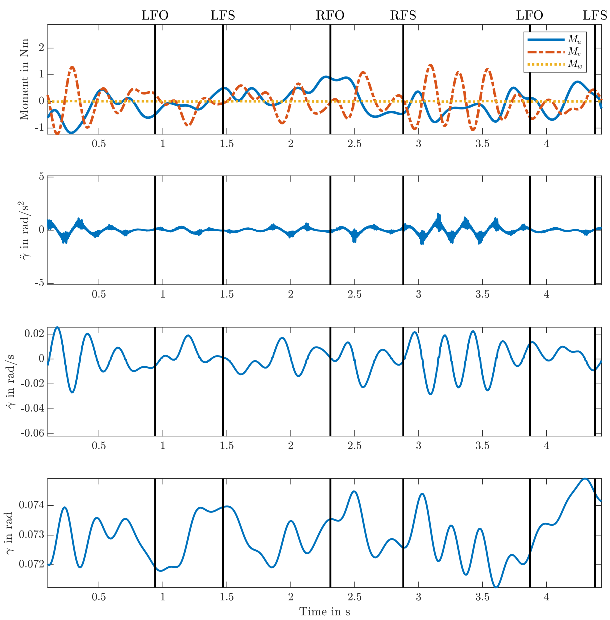


Figure D.35: Time response of a SPCMG when optimized for a mass with realistic bounds. The walking speed was between 0 - 0.4 m/s

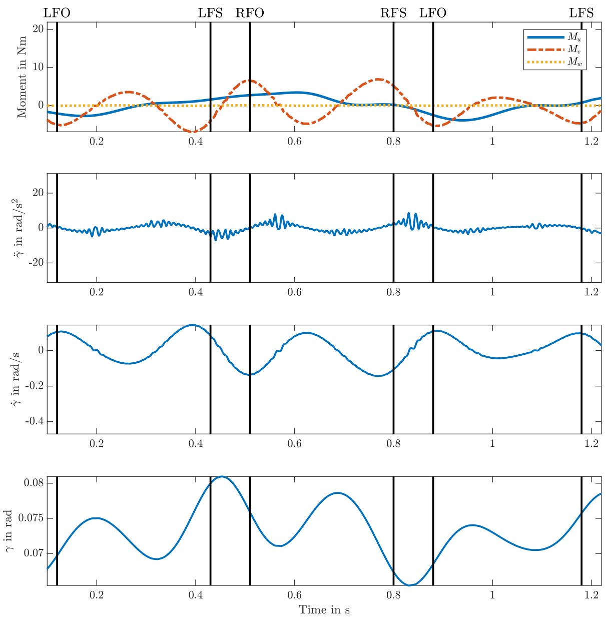


Figure D.36: Time response of a SPCMG when optimized for a mass with realistic bounds. The walking speed a self selected fast speed between 1.9 - 2.2 m/s

D.4.4. Mass Spring Damper

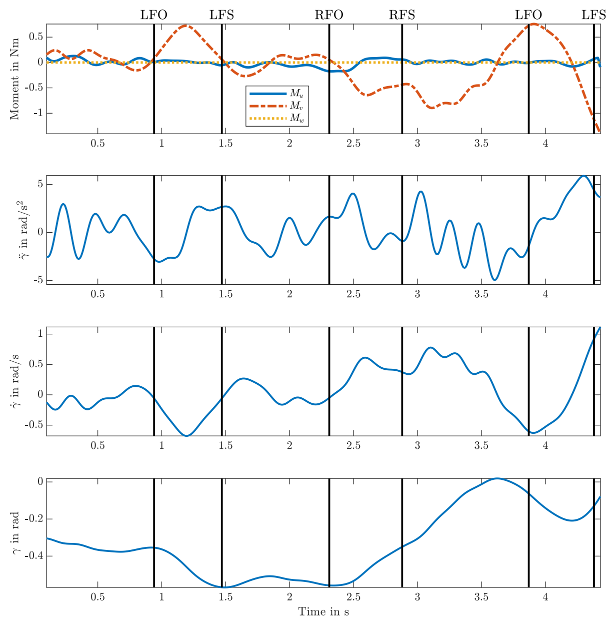


Figure D.37: Time response of a SPCMG when optimized for a mass spring damper system with realistic bounds. The walking speed was between 0 - 0.4 m/s

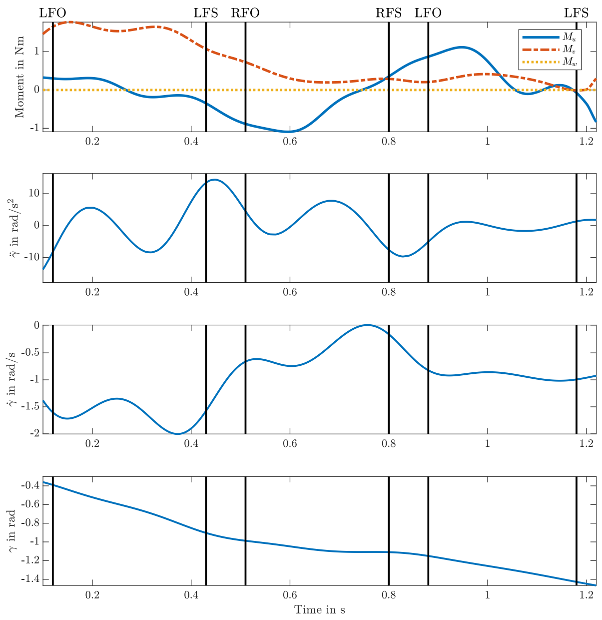


Figure D.38: Time response of a SPCMG when optimized for a mass spring damper system with realistic bounds. The walking speed a self selected fast speed between 1.9 - 2.2 m/s

D.4.5. PDXCoM

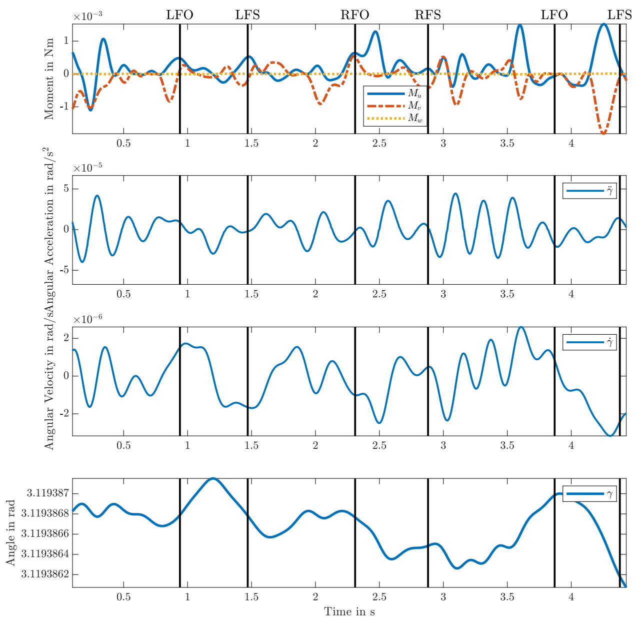


Figure D.39: Time response of a SPCMG when optimized for the PDXCoM with realistic bounds. The walking speed was between 0 - 0.4 m/s

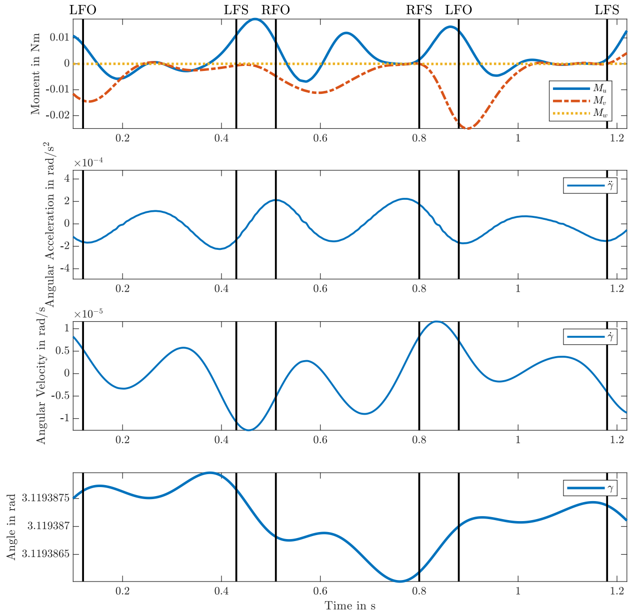


Figure D.40: Time response of a SPCMG when optimized for the PDXCoM with realistic bounds. The walking speed a self selected fast speed between 1.9 - 2.2 m/s

E

Appendix E

Matlab notation

Table E.1: Matlab notation list

Symbol	Matlab name
$\hat{\boldsymbol{e}}_s, \hat{\boldsymbol{e}}_t, \hat{\boldsymbol{e}}_g$	es, et, eg
${}^{\mathcal{B}}\mathbf{R}_{\mathcal{G}}$	bRg
${}^{\mathcal{A}}\boldsymbol{\omega}_{\mathcal{B}/\mathcal{G}}$	wbg_a
${}^{\mathcal{G}}\mathbf{I}_w, {}^{\mathcal{G}}\mathbf{I}_g$	Iwheel_g, Igimbal_g
${}^{\mathcal{G}}\boldsymbol{H}_w, {}^{\mathcal{G}}\boldsymbol{H}_g$	Hwheel_g, Hgimbal_g
${}^{\mathcal{A}}(\dot{\boldsymbol{\omega}}_{\mathcal{B}/\mathcal{N}})_{\mathcal{C}}$	dwbn_c_a
$\gamma, \dot{\gamma}, \ddot{\gamma}$	gamma, dgamma, ddgamma
Ω	omega
${}^{\mathcal{A}}(\dot{\boldsymbol{H}}_w)_{\mathcal{B}}$	dHwheel_b_a
${}^{\mathcal{A}}(\dot{\boldsymbol{H}}_w)_{\mathcal{N}}$	dHwheel_N_a

E.1. Main File: Single CMG

```
% This script is made by Roemer Helwig for his master thesis. It generates
% the equations of motion of a single CMG, the impedance of the CMG.
% Furhtermore, it can optimize the impedance to mimic an arbitrary transfer
% function. With the optimized parameters it can then compute the time
% response.
% Roemer Helwig, 11-12-2019

addpath('Necessary_functions')

clear
close all

% % Bode options
PP                 = bodeoptions;
PP.PhaseWrapping = 'on';
PP.FreqUnits  = 'Hz';
PP.XLim       = [1e-3 1e4];
PP.Grid       = 'on';

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Newton-Euler Equations of Motion
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Generate symbolic variables
syms omega domega gamma m r dgamma ddgamma k g t time r b Mu Mv Mw Js Jt Jg w phi theta psi
syms ws wt wg dwbn dws dwt dwg Igs Igt Igg Iws Iwt Iwg Ms Mt Mg s gamma0 wu wv ww dwu dwv dww wuS wvS wwS gammaS
  disp('EoM via Newton-Euler... ')

% Unit vectors of the gimbal fixed frame
es = [1; 0; 0];
et = [0; 1; 0];
eg = [ 0; 0;1];

% projection of the gimbal fixed frame on the body fixed frame
eu = [cos(-gamma); sin(-gamma); 0];
ev = [-sin(-gamma);cos(-gamma); 0];
ew = [0; 0; 1];

gRb= [eu ev ew];      % Rotation matrix from body to gimbal fixed frame
bRg= transpose(gRb); % Rotation matrix from gimbal to body fixed frame

% Angular velocities in the gimbal frame
wbg_g = [0 ; 0; -dgamma];
wwg_g = [omega; 0; 0];
wgb_g = [0;0;dgamma];

% Angular velocities in the body frame
wbn_b = [wu;wv;ww];
wbg_b = bRg*wbg_g;
wgn_b = wbn_b-wbg_b;

wbn_g = gRb*wbn_b;
wgn_g = gRb*wgn_b;

% Moment of inertia tensor in Gimbal frame
Iwheel_g = diag([Iws;Iwt;Iwt]);
Igimbal_g = diag([Igs;Igt;Igg]);

% Angular momentum in gimbal frame
Hwheel_g  = Iwheel_g*(wwg_g + wgb_g + wbn_g);
Hgimbal_g = Igimbal_g*(wgb_g + wbn_g);

% Angular acceleration of the body frame wrt the natural frame expressed in
% the body frame
dwbn_b_b = [dwu; dwv; dww];

% Angular acceleration of the gimbal frame wrt the natural frame expressed in
% the body frame
dwbn_g_b = dwbn_b_b + cross(wbg_b,wbn_b);

% Angular accelerations in the gimbal frame
dwbn_g_g = gRb*dwbn_g_b;
dwwg_g_g = [domega;0;0];
dwgb_g_g = [0;0;ddgamma];

```

```
domega      = 0;

% Take the time derivative with respect to the G frame
dHwheel_g_g = Iwheel_g*(dwgb_g_g + dwbn_g_g);
dHgimbal_g_g = Igimbal_g*(dwgb_g_g + dwbn_g_g);


% Use transport theorem to calculate derivatives with respect to N frame
dHwheel_N_g = dHwheel_g_g + cross(wgn_g,Hwheel_g);
dHgimbal_N_g = dHgimbal_g_g + cross(wgn_g,Hgimbal_g);
dH_g = dHwheel_N_g + dHgimbal_N_g;

dH_N_b = simplify(bRg*dH_g);

% Moment due to bearings, spring and dampers
Mpassive = [0; 0; 0-b*(dgamma)-k*(gamma-gamma0)];
M_b = dH_N_b - Mpassive;
MBODY = -M_b;

% equation of motion in body frame
[I_b, Mom_b]    = equationsToMatrix(MBODY(3) == 0,ddgamma);
[I2_b, Mom2_b]  = equationsToMatrix(MBODY == 0, ddgamma);
ddgamma_eq = simplify(I_b\Mom_b);


%% Validation
% Rotation Matrices to go to Natural frame
rotphi = [1 0 0;0 cos(phi) -sin(phi);0 sin(phi) cos(phi)];
rottheta = [cos(theta) 0 sin(theta);0 1 0;-sin(theta) 0 cos(theta)];
rotpsi = [cos(psi) sin(psi) 0;-sin(psi) cos(psi) 0;0 0 1];

% Change to natural frame
Hwheel_N = rotphi*rottheta*rotpsi*bRg*(Hwheel_g);
dHwheel_N = rotphi*rottheta*rotpsi*bRg*dHwheel_N_g;

% Validation(Hwheel_N,dHwheel_N);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Lagrange Equations of Motion
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    disp('EoM via Lagrange... ')

q = gamma;
dq = dgamma;
ddq = ddgamma;

% Kinetecs and Potential Engeries
T = 0.5 * ((omega*es + dgamma*eg + gRb*wbn_b).'* Iwheel_g * (omega*es + dgamma*eg +
    gRb*wbn_b) + (+dgamma*eg + gRb*wbn_b).'* Igimbal_g * (+dgamma*eg + gRb*wbn_b));
V = 0.5 * (k * (gamma-gamma0)^2);
% Lagrangian
L = T-V;

% Partial Derivatives
dLdq     = jacobian(L,q);
dLdqd    = jacobian(L,dq);
```

```
ddtdLdqd = jacobian(dLdqd,[q; dq; wbn_b])*[dq; ddq; dwbn_g_b];

% Non conservative forces
Qnc = -b*dgamma;

L_eq = simplify(ddtdLdqd - dLdq.' - Qnc);

[Inertia,Moment]    = equationsToMatrix(L_eq == 0, ddq);
ddq_eq   = simplify(Inertia\Moment);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Check if Newton-Euler and Lagrange are equivalent
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Error = simplify(ddgamma_eq - ddq_eq);



    if Error == 0
        disp('Newton-Euler and Lagrange are equivalent')
    else
        error('Formulations are not equivalent. Please check definitions')
    end

Igt         = Igs;
Mom_b   = subs(Mom_b);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Compute General Transfer functions of the system
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

CompAllTFs = yes_or_no('Compute all the Transfer Functions?'); % function by Daniel
    Lemus

if(CompAllTFs)
% Uncomment following lines to insert values to the impedance
k = 5;
b = 1;
Iws = 4.4e-4;
Iwt = 2.5e-4;
Igs = 8.8e-4;
Igg = 5.0e-4;
omega = 2513;

%linearize for different gammas
gammatemp = [0;pi/6;pi/3;pi/2];

% optional to use different spring stiffness or damping
ktemp       = [0.001;1;100;];
btemp       = [0.001;1;100];
% linearize for specific anglar velocity of the human
wbn_b0 = [0;0;0];


for i = 1:length(gammatemp)
    gammaS = gammatemp(i);
    A_H = linearization(-dH_N_b,[ddgamma;dgamma;gamma;wbn_b;dwbn_b_b],gammaS,wbn_b0)
        ; % Linearization of the Moments
```

```
    dH_lin = A_H*[ddgamma;dgamma;gamma-gammaS;wbn_b-wbn_b0;dwbn_b_b];

    ddgamma  = simplify(inv(I_b)*Mom_b); % recalculate ddgamma
    [A_gamma] = linearization(ddgamma,[dgamma;gamma;wbn_b;dwbn_b_b],gammaS,wbn_b0); % Linearization of ddgamma
    ddgamma_lin = A_gamma*[dgamma;gamma;wbn_b;dwbn_b_b];

    % Take the Laplace transforms
    dgamma = s*gamma;
    ddgamma = s^2*gamma;
    dwu = s*wu;
    dwv = s*wv;
    dww = s*ww;

    gamma = simplify(solve(subs(ddgamma_lin) - s^2*gamma == 0,gamma));  % solve for gamma
    sdH = simplify(subs(subs(dH_lin)));  % Fill in the Laplace transforms in the Linearized moments
    AA = linearization(sdH,wbn_b,[],wbn_b0); % Reduce so that equations are only dependant on wbn
    dH_reduced = AA * wbn_b;

    eq1 = dH_reduced - [Mu;Mv;Mw]; % Make equation: terms - M = 0
    % Compute transfer functions
    Gsuu = comptf(eq1,wu,1,Mu,1);   Gsvu = comptf(eq1,wu,1,Mv,2); Gswu = comptf(eq1,wu,1,Mw,3);
    Gsuv = comptf(eq1,wv,2,Mu,1);   Gsvv = comptf(eq1,wv,2,Mv,2); Gswv = comptf(eq1,wv,2,Mw,3);
    Gsuw = comptf(eq1,ww,3,Mu,1);   Gsvw = comptf(eq1,ww,3,Mv,2); Gsww = comptf(eq1,ww,3,Mw,3);

    Gs = syms2tf(subs(Gsvv));
    Gs.InputName        = '\omega_v';
    Gs.OutputName       = 'M_v';
    G = bodeplot(Gs,PP);
    hold on
    grid on

    clear gamma dgamma ddgamma dws dwt dwg
    syms gamma dgamma ddgamma

%     Compute Transmisability

%     eq2 = gamma2 - gamma;
%     H1 = comptf(eq2,wu,1,gamma,1);
%     H2 = comptf(eq2,wv,2,gamma,1);
%     H3 = comptf(eq2,ww,3,gamma,1);
%     Hs = [H1 H2 H3];
end
% Uncomment following lines to plot the impedance
legend(num2str(gammatemp))
fh = gcf;
lh = findall(fh,'Type','Line');
arrayfun(@(x) set(x,'LineWidth',2),lh)
leg = legend('show');
title(leg,'\gamma')
```

```
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Load Optimal Parameters
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
LoadPar = yes_or_no('Load the best paramters?');
if (LoadPar)
close all

num_opt = 100;
n_par = 5;
x = zeros(n_par,num_opt);
resnorm = ones(1,num_opt)* 1e10;
x0 = zeros(n_par,num_opt);

for j = 1:num_opt

 parameter(j) = load(['opt_parameter_' num2str(j) '.mat'],'x','resnorm','Gs','x0');
 x(:,j) = parameter(j).x;
 resnorm(j) = parameter(j).resnorm;
 Gs(:,j) = parameter(j).Gs;
 x0(:,j) = parameter(j).x0;

end
% Find the best parameters, Gs and initial guess
[~, col] = find(min(resnorm) == resnorm);
col = min(col);
x_best = x(:,col);
Gs = Gs(:,col);
x0 = x0(:,col);
% k = x_best(1); b = x_best(2); Iws = x_best(3); Iwt = x_best(4); Igs = x_best(5); Igt = Igs; Igg = x_best(6); gammaS = x_best(7);
% k = x_best(1); b = x_best(2); Iws = x_best(3); Igg = x_best(4); Iwt = 1/2*Iws; Igs = 1/2*Igg; Igt = Igs;  gammaS = x_best(5);

omega = 2500; k = 0.001; b = 50; Iws = 0.01; Iwt = 4; Igs = 0.1; Igg = 0.3; gammaS = 0;

gamma0 = gammaS;
sortRes = sort(resnorm,'descend');

% Create desired transfer function
kp = 100;
kd = 32;
Jdes = 0.5; bdes = 5; kdes= 30;
% TFdes = syms2tf(+(kp+kd*s)/s);
TFdes = syms2tf(-(kdes)/s);

% Find poles, zeros, damping and natural frequency
[wn,zeta] = damp(Gs);
Gpole = pole(Gs);
Gzero = zero(Gs)

wuS = 0; wvS = 0; wwS = 0;
omega = 1500;
GsvvTemp = syms2tf(subs(Gsvv));
```

```matlab

% Make bode plot of the optimized impedance and the desired transfer
% function

BodeGraph(GsvvTemp,TFdes)

% Make plot of the resnorm
% figure()
% semilogy(sortRes,'mo',...
%       'LineWidth',2,...
%       'MarkerEdgeColor','k',...
%       'MarkerFaceColor',[.49 1 .63],...
%       'MarkerSize',10)
% title('Optimizations Sorted by Resnorm')
% ylabel('resnorm')
% xlabel('Number of Iterations')

end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Optimization of the Transfer Functions
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if LoadPar == 0
    OptTF = yes_or_no('Optimize the Transfer Function?'); % function by Daniel Lemus
if (OptTF)

% Fill in unoptimizable parameters
Igt = Igs;
Igs = 1/2*Igg;
Iwt = 1/2*Iws;
wuS = 0; wvS = 0; wwS = 0;
omega = 1500;

% Create desired transfer function
kp = 100;
kd = 32;
Jdes = 0.5; bdes = 5; kdes= 30;

% Weights for the Cost function
w1 = 100; %best 100
w2 = 1;
% Parameters that will be optimized
par = [k b Iws Igg gammaS];
% Create frequency vector in Hz
wHz = logspace(-2,1,2e2);
% Create frequency vector in rad/s
w = wHz*2*pi;
num_opt = 100;

TFdes = -(Jdes*s^2 + bdes*s + kdes)/s;
% TFdes= +(kp+kd*s)/s;

s = 1j*w;                                                              %
    substitude s for jw
Gsn = subs(subs(Gsvv));
TFdes1 = subs(subs(TFdes));
```

```
C1 = w1*(imag(TFdes1-Gsn));                                             % Phase
     part of costfunction
C2 = w2*(real(TFdes1-Gsn));                                             %
     Magnitude part of costfunction
C = C1+C2;
errorfun = matlabFunction(C,'Vars',{par});


for j = 1:num_opt
[x,resnorm,Gs,~,x0] = optimization(Gsvv,TFdes,errorfun);
save(['RP_MSD_opt_parameter_' num2str(j) '.mat'],'x','resnorm','Gs','x0')
end


clear s
syms s

load gong.mat;
sound(y,Fs);


else
% k = 1.20; b = 8.13; omega = 2.513e+03; Iws = 0.1238; Iwt = 0.0116; Igg = 0.153;
     gammaS = 0; gamma0 = gammaS; Igs = 0.001; Igt = 0.001;
end
end



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Fill in Parameters and compute Frequency response
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if (CompAllTFs)
SubsTF = yes_or_no('Fill in parameters in TFs and compute Freq Response?');

% Compute all transfer functions
if (SubsTF)
wuS = 0.1; wvS = 0.1; wwS = 0.1;
Gsuu = zpk(syms2tf(subs(Gsuu)));    Gsvu = zpk(syms2tf(subs(Gsvu)));    Gswu = zpk(
     syms2tf(subs(Gswu)));
Gsuv = zpk(syms2tf(subs(Gsuv)));    Gsvv = zpk(syms2tf(subs(Gsvv)));    Gswv = zpk(
     syms2tf(subs(Gswv)));
Gsuw = zpk(syms2tf(subs(Gsuw)));    Gsvw = zpk(syms2tf(subs(Gsvw)));    Gsww = zpk(
     syms2tf(subs(Gsww)));

Gstot = [Gsuu Gsvu Gswu;Gsuv Gsvv Gswv; Gsuw Gsvw Gsww];
     Gstot.InputName        = 'Moment';
     Gstot.OutputName       = 'omega';
figure()
bodeP = bodeplot(Gstot,PP);
p=getoptions(bodeP);
% p.Ylim{1}= [-10 100]; %Setting the y-axis limits
% p.Ylim{2}= [-10 100]; %Setting the y-axis limits
% p.Ylim{3}= [-10 100]; %Setting the y-axis limits
setoptions(bodeP,p); %update your plot
fh = gcf;
```

```
lh = findall(fh,'Type','Line');
arrayfun(@(x) set(x,'LineWidth',1.5),lh)
end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Comp Time Response from Gait Data
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
CompTimeResp = yes_or_no('Compute time response from gait data?');

if (CompTimeResp)
close all
clear s dwu dwv dww gamma dgamma ddgamma wu wv ww
syms dwu dwv dww gamma dgamma ddgamma t time wu wv ww

FrameRate = 100; % per second
h = 1/FrameRate; % time step
h2 = 0.01*h;     % time step for interpolation
omega = 1500;
ddgamma_eq = subs(ddgamma_eq);
M_b_opt = subs(subs(MBODY));

Condition = 5; % Select which walking condition to use for input. Range from 1 to 5.

% Load gait data
addpath('Matlab Motion Data')

AngVel = load(['AngVel' num2str(Condition) '.txt']);
AngAcc_temp = load(['AngAcc' num2str(Condition) '.txt']);
AngAcc = zeros(length(AngVel),3);

AngAcc(2:end-1,:) = AngAcc_temp;
TrunkRot = wrapTo360(load(['TrunkRot' num2str(Condition) '.txt']));
EventData = xlsread(['Events' num2str(Condition) '.xlsx']);

[LFO,LFS,RFO,RFS,TimePoint] = RecEvent(EventData);
et = (0:length(AngVel)-1)'*h;

% Create function of the gait data
omega_func    = @(t_i) interp1(et,AngVel,t_i);
wv_func       = @(t_i) interp1(et,AngVel(:,2),t_i);
wu_func       = @(t_i) interp1(et,AngVel(:,1),t_i);
dww_func      = @(t_i) interp1(et,AngAcc(:,3),t_i);

% Create function of the moments and ddgamma
Mcmg_b        = matlabFunction(M_b_opt,'file','Mcmg_b');
ddgamma_fun_b = matlabFunction([ddgamma_eq],'file','ddgamma_fun_b');

% % Create function handle and use ode15s for numerical integration
% ddgamma_func = @(t,y) ddgamma_fun_b(y(2),dww_func(t),y(1),wu_func(t),wv_func(t));
% [t,y] = ode15s(ddgamma_func,[0 3],[0 1]);

wu1 = AngVel(:,1);

dwu1 = AngAcc(:,1);
dwv1 = AngAcc(:,2);
```

```
dwwl = AngAcc(:,3);

Time = length(wul)*h;
% Interpolate to improve integration
wul = interp1(0:h:(Time-h),wul,0:h2:(Time-h2),'PCHIP');
wvl = AngVel(:,2);
wvl = interp1(0:h:(Time-h),wvl,0:h2:(Time-h2),'PCHIP');
wwl = AngVel(:,3);
wwl = interp1(0:h:(Time-h),wwl,0:h2:(Time-h2),'PCHIP');

dwul = interp1(0:h:(Time-h),dwul,0:h2:(Time-h2),'PCHIP');
dwvl = interp1(0:h:(Time-h),dwvl,0:h2:(Time-h2),'PCHIP');
dwwl = interp1(0:h:(Time-h),dwwl,0:h2:(Time-h2),'PCHIP');

TrunkRot = interp1(0:h:(Time),TrunkRot,0:h2:(Time),'PCHIP');
% Create initial conditions
w = zeros(3,length(wul));
dw = zeros(3,length(wul));
wu = wul(1,1);     wv = wvl(1,1);     ww = wwl(1,1);
dwu= dwul(1,1);    dwv= dwvl(1,1);    dww= dwwl(1,1);
ddgammal = zeros(1,length(wul)); dgammal  = zeros(1,length(wul)); gammal   = zeros
    (1,length(wul));
gamma     = gammaS;
dgamma    = 0;
initial_conditions = [wu;wv;ww;gamma;dgamma];
M_b_optl = zeros(3,length(wul));

ddgammal(1,1) = ddgamma_fun_b(dgamma,dww,gamma,wu,wv);
ddgamma = ddgammal(1,1);
M_b_optl(1:3,1) = Mcmg_b(ddgamma,dgamma,dwu,dwv,dww,gamma,wu,wv,ww);
gammal(1,1) = gamma;
dgammal(1,1) = dgamma;
% Numerical integration
for i = 2:length(wvl)
    nC = rotx((TrunkRot(i,1)))*roty((TrunkRot(i,2)))*rotz((TrunkRot(i,3)+pi/2));
    w(1:3,i) = nC*[wul(i);wvl(i);wwl(i)];
    dw(1:3,i)  = nC*[dwul(i);dwvl(i);dwwl(i)];
    wu = w(1,i);     wv = w(2,i);     ww = w(3,i);
    dwu= dw(1,i);    dwv= dw(2,i);    dww= dw(3,i);
    ddgammal(i) = ddgamma_fun_b(dgamma,dww,gamma,wu,wv);
    ddgamma = ddgammal(i);
    dgammal(i) = dgammal(i-1) + double(ddgammal(i)*h2);
    dgamma = dgammal(i);
    gammal(i) = gammal(i-1) + double(dgammal(i)*h2 + 0.5*ddgamma*h2^2);
    gamma = gammal(i);

    M_b_optl(:,i) = Mcmg_b(ddgamma,dgamma,dwu,dwv,dww,gamma,wu,wv,ww);

    if isnan(ddgamma) == 1
        error('decrease time step')
    end
%     controle(i) = dgamma+wg;
%     check(i) = Mt/controle(i);
end
```

```
% Plot Generated Moments Due Walking %
GaitEvent = [LFO,LFS,RFO,RFS];
FirstEvent = find(GaitEvent(1,:) == 0);
if FirstEvent == 1
    Tag1 = 'LFO';
    Tag2 = 'LFS';
    Tag3 = 'RFO';
    Tag4 = 'RFS';
elseif FirstEvent == 2
    Tag1 = 'LFS';
    Tag2 = 'RFO';
    Tag3 = 'RFS';
    Tag4 = 'LFO';
elseif FirstEvent ==3
    Tag1 = 'RFO';
    Tag2 = 'RFS';
    Tag3 = 'LFO';
    Tag4 = 'LFS';
elseif FirstEvent == 4
    Tag1 = 'RFS';
    Tag2 = 'LFO';
    Tag3 = 'LFS';
    Tag4 = 'RFO';
end
Tag5 = Tag1;
Tag6 = Tag2;
Tag7 = Tag3;
if TimePoint(1) < 0.1
    Tag1 = ' ' ;
end


figure()
subplot(4,1,1)
plot(0:h2:(Time-h2),M_b_opt1(1,:),'Linewidth',2,'Linestyle','-')
hold on
plot(0:h2:(Time-h2),M_b_opt1(2,:),'Linewidth',2,'Linestyle','-.')
plot(0:h2:(Time-h2),M_b_opt1(3,:),'Linewidth',2,'Linestyle',':')
ylabel('Moment in Nm')
xlim([0.1 Time(end)])
ylim([min(M_b_opt1(:)) max(M_b_opt1(:))])
vline(TimePoint(1),'k');
text(TimePoint(1),max(M_b_opt1(:)),Tag1, 'HorizontalAlignment','center', '
    VerticalAlignment','bottom','FontSize',11)
vline(TimePoint(2),'k');
text(TimePoint(2),max(M_b_opt1(:)),Tag2, 'HorizontalAlignment','center', '
    VerticalAlignment','bottom','FontSize',11)
vline(TimePoint(3),'k');
text(TimePoint(3),max(M_b_opt1(:)),Tag3, 'HorizontalAlignment','center', '
    VerticalAlignment','bottom','FontSize',11)
vline(TimePoint(4),'k');
text(TimePoint(4),max(M_b_opt1(:)),Tag4, 'HorizontalAlignment','center', '
    VerticalAlignment','bottom','FontSize',11)
vline(TimePoint(5),'k');
text(TimePoint(5),max(M_b_opt1(:)),Tag5, 'HorizontalAlignment','center', '
    VerticalAlignment','bottom','FontSize',11)
```

```
vline(TimePoint(6),'k');
text(TimePoint(6),max(M_b_opt1(:)),Tag6, 'HorizontalAlignment','center', '
    VerticalAlignment','bottom','FontSize',11)
vline(TimePoint(7),'k');
text(TimePoint(7),max(M_b_opt1(:)),Tag7, 'HorizontalAlignment','center', '
    VerticalAlignment','bottom','FontSize',11)
legend('$M_u$' , '$M_v$' , '$M_w$','Location','best');

subplot(4,1,2)
plot(0:h2:(Time-h2),ddgamma1,'Linewidth',1.5);
ylabel('$\ddot{\gamma}$ in rad/s$^{2}$')
xlim([0.1 Time(end)])
ylim([mean(ddgamma1)-2.3*std(ddgamma1) mean(ddgamma1)+2.3*std(ddgamma1)])
vline(TimePoint(1),'k');
vline(TimePoint(2),'k');
vline(TimePoint(3),'k');
vline(TimePoint(4),'k');
vline(TimePoint(5),'k');
vline(TimePoint(6),'k');
vline(TimePoint(7),'k');

subplot(4,1,3)
plot(0:h2:(Time-h2),dgamma1,'Linewidth',1.5);
ylabel('$\dot{\gamma}$ in rad/s')
% ylim([mean(dgamma1)-1.5*std(dgamma1) mean(dgamma1)+1.5*std(dgamma1)])
xlim([0.1 Time(end)])
ylim([min(dgamma1(:)) max(dgamma1(:))])
vline(TimePoint(1),'k');
vline(TimePoint(2),'k');
vline(TimePoint(3),'k');
vline(TimePoint(4),'k');
vline(TimePoint(5),'k');
vline(TimePoint(6),'k');
vline(TimePoint(7),'k');

subplot(4,1,4)
plot(0:h2:(Time-h2),gamma1,'Linewidth',2);
ylabel('${\gamma}$ in rad')
xlabel('Time in s')
xlim([0.1 Time(end)])
ylim([min(gamma1(:)) max(gamma1(:))])
vline(TimePoint(1),'k');
vline(TimePoint(2),'k');
vline(TimePoint(3),'k');
vline(TimePoint(4),'k');
vline(TimePoint(5),'k');
vline(TimePoint(6),'k');
vline(TimePoint(7),'k');

setInterpreter(gcf,'latex');
% save_fig(gcf,'path','/Figures/','filename',{'GyRAB_contour_10Nms'},'extensions',{'
    matlabfrag'})

% Plot Angular Velocity Data %
% figure()
% plot(0:h2:(Time-h2),w,'Linewidth',2)
```

```
% xlim([0.1 Time-h2])
% ylim([min(w(:)) max(w(:))])
% hold on
% vline(TimePoint(1),'k');
% text(TimePoint(1),max(w(:)),Tag1, 'HorizontalAlignment','center', '
    VerticalAlignment','bottom','FontSize',11)
% vline(TimePoint(2),'k');
% text(TimePoint(2),max(w(:)),Tag2, 'HorizontalAlignment','center', '
    VerticalAlignment','bottom','FontSize',11)
% vline(TimePoint(3),'k');
% text(TimePoint(3),max(w(:)),Tag3, 'HorizontalAlignment','center', '
    VerticalAlignment','bottom','FontSize',11)
% vline(TimePoint(4),'k');
% text(TimePoint(4),max(w(:)),Tag4, 'HorizontalAlignment','center', '
    VerticalAlignment','bottom','FontSize',11)
% vline(TimePoint(5),'k');
% text(TimePoint(5),max(w(:)),Tag5, 'HorizontalAlignment','center', '
    VerticalAlignment','bottom','FontSize',11)
% vline(TimePoint(6),'k');
% text(TimePoint(6),max(w(:)),Tag6, 'HorizontalAlignment','center', '
    VerticalAlignment','bottom','FontSize',11)
% vline(TimePoint(7),'k');
% text(TimePoint(7),max(w(:)),Tag7, 'HorizontalAlignment','center', '
    VerticalAlignment','bottom','FontSize',11)
% legend('$\omega_u$','$\omega_v$','$\omega_w$')
% setInterpreter(gcf,'latex');

end

%% Functions
function [x1,resnorm,Gs,TFdes,x0] = optimization(Gs,TFdes,errorfun)
ub = [4500, 3800, 0.04, 0.02, 0.1];                              % Upper bounds

x0 = [rand(1)*0.01 , rand(1)*0.001 , rand(1)*ub(3), rand(1)*ub(4), rand(1)*pi + rand
    (1)*-pi]; % Initial guess

% XS = [4500, 3800, 0.3, 0.3, 0];                                      % Upper
    bounds
% x0 = [rand(1)*0.01 , rand(1)*0.001 , rand(1)*XS(3), rand(1)*XS(4), rand(1)*pi +
    rand(1)*-pi]; % Initial guess
%
% ub = [inf, inf, inf, inf, 0.1];

lb = [0 , 0, 0, 0, -0.1];                                            % Lower
    bounds

options = optimoptions(@lsqnonlin,'Algorithm','trust-region-reflective');
options.MaxFunctionEvaluations = 180000;
options.MaxIterations = 12000;
[x1, resnorm] = lsqnonlin(errorfun,x0,lb,ub,options);                    %
    Optimization function
k = x1(1); b = x1(2); Iws = x1(3); Igg = x1(4); gammaS = x1(5); Iwt = 1/2*Iws; Igs =
    1/2*Igg; Igt = Igs;



```

```
clear s
syms s
Gs = syms2tf(subs(Gs));

end
```

E.2. Main File: SPCMG

```
% This script is made by Roemer Helwig for his master thesis. It generates
% the equations of motion of a SPCMG, the impedance of the SPCMG.
% Furhtermore, it can optimize the impedance to mimic an arbitrary transfer
% function. With the optimized parameters it can then compute the time
% response.
% Roemer Helwig, 11-12-2019

addpath('Necessary_functions')

clear
close all

% Bode options
PP                  = bodeoptions;
PP.PhaseWrapping = 'on';
PP.FreqUnits = 'Hz';
PP.XLim      = [1e-4 2e2];
PP.Grid      = 'on';

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Newton-Euler Equations of Motion
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Generate symbolic variables
syms omega domega gamma m r dgamma ddgamma k g t time r b Mu Mv Mw Js Jt Jg Mc w phi
    theta psi
syms ws wt wg dwbn dws dwt dwg Igs Igt Igg Iws Iwt Iwg Ms Mt Mg s gamma0 wu wv ww
    dwu dwv dww wuS wvS wwS gammaS
    disp('EoM via Newton-Euler...  ')

%% Gimbal 1
% Unit vectors of the first gimbal fixed frame
gs = [1; 0; 0];
gt = [0; 1; 0];
gg = [ 0; 0;1];

% projection of the gimbal fixed frame on the body fixed frame
eu1 = [cos(-gamma); sin(-gamma); 0];
ev1 = [-sin(-gamma);cos(-gamma); 0];
ew1 = [0; 0; 1];

g1Rb= [eu1 ev1 ew1];      % Rotation matrix from body to gimbal fixed frame
bRg1= transpose(g1Rb); % Rotation matrix from gimbal to body fixed frame

% Angular velocities in the gimbal frame
wbg_g1 = [0 ; 0; -dgamma];
wwg_g1 = [omega; 0; 0];
wgb_g1 = [0;0;dgamma];
```

```
% Angular velocities in the body frame
wbn_b = [wu;wv;ww];
wbg1_b = bRg1*wbg_g1;
wg1n_b = wbn_b-wbg1_b;

wbn_g1 = g1Rb*wbn_b;

% Moment of inertia tensor in Gimbal frame
Iwheel_g1 = diag([Iws;Iwt;Iwt]);
Igimbal_g1 = diag([Igs;Igt;Igg]);

% Angular momentum in gimbal frame
Hwheel_g1  = Iwheel_g1*(wwg_g1 + wgb_g1 + wbn_g1);
Hgimbal_g1 = Igimbal_g1*(wgb_g1 + wbn_g1);

% Angular acceleration of the body frame wrt the natural frame expressed in
% the body frame
dwbn_b_b = [dwu; dwv; dww];

% Angular acceleration of the gimbal frame wrt the natural frame expressed in
% the body frame
dwbn_g1_b = dwbn_b_b + cross(wbg1_b,wbn_b);

% Angular accelerations in the gimbal frame
dwbn_g1_g1 = g1Rb*dwbn_g1_b;
dwwg_g1_g1 = [domega;0;0];
dwgb_g1_g1 = [0;0;ddgamma];

domega     = 0;

% Take the time derivative with respect to the G frame
dHwheel_g1_g1 = Iwheel_g1*(dwgb_g1_g1 + dwbn_g1_g1);
dHgimbal_g1_g1 = Igimbal_g1*(dwgb_g1_g1 + dwbn_g1_g1);


% Use transport theorem to calculate derivatives with respect to N frame
dHwheel_N_g1 = dHwheel_g1_g1 + cross(g1Rb*(wg1n_b),Hwheel_g1);
dHgimbal_N_g1 = dHgimbal_g1_g1 + cross(g1Rb*(wg1n_b),Hgimbal_g1);
dH_N_g1 = dHwheel_N_g1 + dHgimbal_N_g1;

dH1_N_b = simplify(bRg1*dH_N_g1);

M1      = dH1_N_b - [0;0;-k*(gamma-gamma0)-b*dgamma+Mc];
%% Gimbal 2
% projection of the gimbal fixed frame on the gimbal fixed frame
eu2 = [cos(gamma); sin(gamma); 0];
ev2 = [-sin(gamma);cos(gamma); 0];
ew2 = [0; 0; 1];
bRg2= transpose([eu2 ev2 ew2]); % Rotation matrix from gimbal to body fixed frame
g2Rb= [eu2 ev2 ew2];

% Angular velocities in the second gimbal frame
wbg2_g2 = [0 ; 0; dgamma];
wwg2_g2 = [-omega;0;0];
wg2b_g2 = [0;0;-dgamma];

```

```
% Angular velocities in the body frame
wbn_b = [wu;wv;ww];
wbg2_b = bRg2*wbg2_g2;
wg2n_b = wbn_b-wbg2_b;

% Moment of inertia tensor in Gimbal frame
Iwheel_g2   = diag([Iws;Iwt;Iwt]);
Igimbal_g2 = diag([Igs;Igt;Igg]);

% Angular momentum in gimbal frame
Hwheel_g2   = Iwheel_g2*(wwg2_g2 + wg2b_g2 + g2Rb*wbn_b);
Hgimbal_g2 = Igimbal_g2*(wg2b_g2 + g2Rb*wbn_b);

%Angular acceleration of the second gimbal fram wrt the natural frame
%expressed in the body frame
dwbn_g2_b = dwbn_b_b + cross(wbg2_b,wbn_b);

% Angular accelerations in the gimbal frame
dwbn_g2_g2 = g2Rb*dwbn_g2_b;
dwwg2_g2_g2 = [-domega;0;0];
dwg2b_g2_g2 = [0;0;-ddgamma];

dHwheel_g2_g2 = Iwheel_g2*(dwg2b_g2_g2 + dwbn_g2_g2);
dHgimbal_g2_g2 = Igimbal_g2*(dwg2b_g2_g2 + dwbn_g2_g2);

% Use transport theorem to calculate derivatives with respect to N frame
dHwheel_N_g2 = dHwheel_g2_g2 + cross(g2Rb*wg2n_b,Hwheel_g2);
dHgimbal_N_g2 = dHgimbal_g2_g2 + cross(g2Rb*wg2n_b,Hgimbal_g2);
dH_g2 = dHwheel_N_g2 + dHgimbal_N_g2;

dH2_N_b = simplify(bRg2*dH_g2);

M2           = dH2_N_b - [0;0;+k*(gamma-gamma0)+b*dgamma+Mc];
%% Total system
% Moment due to spring and dampers
Mc = solve(M2(3) == 0,Mc);
M1 = subs(M1);
M_b = M1 + [M2(1);M2(2);0];
MBODY = -M_b;

% equation of motion in body frame
[I_b, Mom_b]    = equationsToMatrix(MBODY(3) == 0,ddgamma);

ddgamma_eq     = simplify(inv(I_b)*Mom_b);
%dwb_bn_b_b = simplify(inv(I2_b)*Mom2_b);


%% Validation

Htot_b = bRg1*Hwheel_g1 + bRg2*Hwheel_g2;
rotphi = [1 0 0;0 cos(phi) -sin(phi);0 sin(phi) cos(phi)];
rottheta = [cos(theta) 0 sin(theta);0 1 0;-sin(theta) 0 cos(theta)];
rotpsi = [cos(psi) sin(psi) 0;-sin(psi) cos(psi) 0;0 0 1];
Htot_N = rotphi*rottheta*rotpsi*Htot_b;

dHtot_b = bRg1*dHwheel_N_g2 + bRg2*dHwheel_N_g2;
```

```
dHtot_N = rotphi*rottheta*rotpsi*dHtot_b;

% Validation(Htot_N,dHtot_N);


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Lagrange Equations of Motion
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    disp('EoM via Lagrange...  ')

q = gamma;
dq = dgamma;
ddq = ddgamma;

% Kinetecs and Potential Engeries
T1 = 0.5 * ((omega*gs + dgamma*gg + g1Rb*wbn_b).'* Iwheel_g1 * (omega*gs + dgamma*gg
    + g1Rb*wbn_b) + (dgamma*gg + g1Rb*wbn_b).'* Igimbal_g1 * (dgamma*gg + g1Rb*wbn_b
    ));
T2 = 0.5 * ((omega*-gs + -dgamma*gg + g2Rb*wbn_b).'* Iwheel_g2 * (omega*-gs + -
    dgamma*gg + g2Rb*wbn_b) + (-dgamma*gg + g2Rb*wbn_b).'* Igimbal_g2 * (-dgamma*gg +
    g2Rb*wbn_b));
T = T1+T2;
V1 = 0.5 * (k * (gamma-gamma0)^2);
V2 = 0.5 * (k * (gamma0-gamma)^2);
V = V1+V2;
L = T-V;

dLdq     = jacobian(L,q);
dLdqd    = jacobian(L,dq);
ddtdLdqd = jacobian(dLdqd,[q; dq; wbn_b])*[dq; ddq; dwbn_g1_b];

Qnc = -2*b*dgamma;

L_eq = simplify(ddtdLdqd - dLdq.' - Qnc);

[Inertia,Moment]   = equationsToMatrix(L_eq == 0, ddq);
ddq_eq   = simplify(Inertia\Moment);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Check if Newton-Euler and Lagrange are equivalent
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Error = simplify(ddgamma_eq - ddq_eq);

    if Error == 0
        disp('Newton Euler and Lagrange are equivalent')
    else
        error('Formulations are not equivalent. Please check definitions')
    end

% omega      =  2513; %1500
Igt         = Igs;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Compute General Transfer functions of the system
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

CompAllTFs = yes_or_no('Compute all the Transfer Functions?'); % function by Daniel
```

```
    Lemus

if(CompAllTFs)


%linearize for different gammas, stiffness or damping
gammatemp = [gammaS];
% optional to use different spring stiffness or damping
ktemp = [0.1;0.5;1;3;5;10];
btemp = [0.1;0.5;1;3;5;10];
domega     = 0;
% linearize for specific anglar velocity of the human
wbn_b0 = [wuS;wvS;wwS];


for i = 1:length(gammatemp)
    %b = btemp(i);
    A = linearization((-dH1_N_b-dH2_N_b),[ddgamma;dgamma;gamma;wbn_b;dwbn_b_b],gammatemp(i),wbn_b0); % Linearization of the Moments
    dH_lin = A*[ddgamma;dgamma;gamma-gammatemp(i);wbn_b-wbn_b0;dwbn_b_b];

    ddgamma     = simplify(inv(subs(I_b))*subs(Mom_b)); % recalculate ddgamma
    [Ag] = linearization([ddgamma],[dgamma;gamma;wbn_b;dwbn_b_b],gammatemp(i),wbn_b0); % Linearization of ddgamma
    ddgamma_lin = Ag*[dgamma;gamma;wbn_b;dwbn_b_b];

    dgamma = s*gamma;  % Take the Laplace transforms
    ddgamma = s^2*gamma;
    dwu = s*wu;
    dwv = s*wv;
    dww = s*ww;

    gamma = simplify(solve(subs(ddgamma_lin) - s^2*gamma == 0,gamma));  % solve for gamma

    sdH = simplify(subs(subs(dH_lin))); % Fill in the Laplace transforms in the Linearized moments
    AA = linearization(sdH,wbn_b,[],wbn_b0); % Linearize again
    dH_reduced = AA * wbn_b;

    eq1 = dH_reduced - [Mu;Mv;Mw]; % Make equation: terms - M = 0
    % Compute transfer functions
    Gsuu = comptf(eq1,wu,1,Mu,1); Gsuv = comptf(eq1,wv,2,Mu,1); Gsuw = comptf(eq1,ww,3,Mu,1);
    Gsvu = comptf(eq1,wu,1,Mv,2); Gsvv = comptf(eq1,wv,2,Mv,2); Gsvw = comptf(eq1,ww,3,Mv,2);
    Gswu = comptf(eq1,wu,1,Mw,3); Gswv = comptf(eq1,wv,2,Mw,3); Gsww = NaN;

%      clear gamma dgamma ddgamma dws dwt dwg
%      syms gamma dgamma ddgamma
%      % Comute transmissability
%      eq2 = gamma2 - gamma;
%      H1 = comptf(eq2,wu,1,gamma,1);
%      H2 = comptf(eq2,wv,2,gamma,1);
%
%      H3 = comptf(eq2,ww,3,gamma,1);
```

```
%       Hs = [H1 H2 H3];
%
%       Hs.InputName            = 'Moment';
%       Hs.OutputName           = 'omega';
%       H = bodeplot(Hs,PP);
%       setoptions(H,'FreqUnits','Hz','PhaseVisible','on');
%       hold on
%       grid on

end

end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Load Optimal Parameters
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
LoadPar = yes_or_no('Load the best paramters?');
if (LoadPar)
close all
% addpath('Par_Scissored')
it = 100;
n_par = 5;
x = zeros(n_par,it);
resnorm = ones(1,it)*1e10;
x0 = zeros(n_par,it);

% Load results of the optimizations
for j = 1:it

 parameter(j) = load(['opt_parameter_' num2str(j) '.mat'],'x','resnorm','Gs','x0');
 x(:,j) = parameter(j).x;
 resnorm(j) = parameter(j).resnorm;
 Gs(:,j) = parameter(j).Gs;
 x0(:,j) = parameter(j).x0;

end

% Find the best parameters, Gs and initial guess
[~, col] = find(min(resnorm) == resnorm);
col = max(col);
x_best = x(:,col);
Gs = Gs(:,col);
x0 = x0(:,col);
% k = x_best(1); b = x_best(2); Iws = x_best(3); Iwt = x_best(4); Igs = x_best(5);
    Igt = Igs; Igg = x_best(6); gammaS = x_best(7);
k = x_best(1); b = x_best(2); Iws = x_best(3); Igg = x_best(4); Iwt = 1/2*Iws; Igs =
     1/2*Igg; Igt = Igs;  gammaS = x_best(5);

gamma0 = gammaS;

sortRes = sort(resnorm,'descend');

% Create desired transfer function
kp = 100;
kd = 32;
Jdes = 0.5; bdes = 5; kdes= 30;
```

```
% TFdes = syms2tf(+(kp+kd*s)/s);
TFdes = syms2tf(-(bdes*s)/s);
% Find the poles, zeros, and the natural frequency
Gpole = pole(Gs);
[wn, zeta] = damp(Gs);
Gzero = zero(Gs);

% wuS = 0.1; wvS = 0.1; wwS = 0.1;
% omega = 2513;
% GsvvTemp = syms2tf(subs(Gsvv));

BodeGraph(Gs,TFdes)

% Plot the Resnorm in descending order
% figure()
% semilogy(sortRes,'mo',...
%       'LineWidth',1.5,...
%       'MarkerEdgeColor','k',...
%       'MarkerFaceColor',[.49 1 .63],...
%       'MarkerSize',10)
% title('Optimizations Sorted by Resnorm')
% ylabel('resnorm')
% xlabel('Number of Iterations')
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Optimization of the Transfer Functions
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
 if LoadPar == 0
OptTF = yes_or_no('Optimize the Transfer Function?'); % function by Daniel Lemus
 if (OptTF)

% Fill in unoptimizable parameters
 Igt = Igs;
 Igs = 1/2*Igg;
 Iwt = 1/2*Iws;
wuS = 0; wvS = 0; wwS = 0;
omega = 1500;

% Create desired transfer function
kp = 100;
kd = 32;
 Jdes = 0.5; bdes = 5; kdes= 30;

% Weights for the Cost function
w1 = 100; %best 100
w2 = 1;
% Parameters that will be optimized
 par = [k b Iws Igg gammaS];
% Create frequency vector in Hz
wHz = logspace(-2,1,2e2);
% Create frequency vector in rad/s
w = wHz*2*pi;
num_opt = 100;

TFdes = -(Jdes*s^2 + bdes*s + kdes)/s;
```

```
% TFdes= +(kp+kd*s)/s;

s = 1j*w;                                                        % substitude s for jw
Gsn = subs(subs(Gsvv));
TFdes1 = subs(subs(TFdes));
C1 = w1*(imag(TFdes1-Gsn));                                      % Phase part of costfunction
C2 = w2*(real(TFdes1-Gsn));                                      % Magnitude part of costfunction
C = C1+C2;
errorfun = matlabFunction(C,'Vars',{par});


for j = 1:num_opt
[x,resnorm,Gs,~,x0] = optimization(Gsvv,TFdes,errorfun);
save(['opt_parameter_' num2str(j) '.mat'],'x','resnorm','Gs','x0')
end


clear s
syms s



load gong.mat;
sound(y,Fs);


else
% k = 30.20; b = 20.13; omega = 2.513e+03; Iws = 0.1238; Iwt = 0.0116; Igg = 0.153; gammaS = -1.891; gamma0 = gammaS; Igs = 0.001; Igt = 0.001;
end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Fill in Parameters and compute Frequency response
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if (CompAllTFs)
SubsTF = yes_or_no('Fill in parameters in TFs and compute Freq Response?');

if (SubsTF)
wuS = 0.1; wvS = 0.1; wwS = 0.1;
Gsuu = zpk(syms2tf(subs(Gsuu)));    Gsvu = zpk(syms2tf(subs(Gsvu)));    Gswu = zpk(syms2tf(subs(Gswu)));
Gsuv = zpk(syms2tf(subs(Gsuv)));    Gsvv = zpk(syms2tf(subs(Gsvv)));    Gswv = zpk(syms2tf(subs(Gswv)));
Gsuw = zpk(syms2tf(subs(Gsuw)));    Gsvw = zpk(syms2tf(subs(Gsvw)));    Gsww = zpk(syms2tf(subs(0)));

Gstot = [Gsuu Gsvu Gswu;Gsuv Gsvv Gswv; Gsuw Gsvw Gsww];
    Gstot.InputName        = 'omega';
    Gstot.OutputName       = 'Moment';
figure()
bodeplot(Gstot,PP)
fh = gcf;
```

```
lh = findall(fh,'Type','Line');
arrayfun(@(x) set(x,'LineWidth',2),lh)

end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% Comp Time Response from Gait Data
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
CompTimeResp = yes_or_no('Compute time response from gait data?');
if (CompTimeResp)
close all
clear s dwu dwv dww gamma dgamma ddgamma wu wv ww
syms dwu dwv dww gamma dgamma ddgamma t time wu wv ww


FrameRate = 100; % per second
h = 1/FrameRate; % time step
h2 = 0.01*h;     % time step for interpolation

omega = 1500;

ddgamma_eq = subs(ddgamma_eq);
M_b_opt = subs(subs(MBODY));

Condition = 1;

% Load gait data
addpath('Matlab Motion Data')
AngVel = load(['AngVel' num2str(Condition) '.txt']);
AngAcc_temp = load(['AngAcc' num2str(Condition) '.txt']);
AngAcc = zeros(length(AngVel),3);
AngAcc(2:end-1,:) = AngAcc_temp;
TrunkRot = wrapTo360(load(['TrunkRot' num2str(Condition) '.txt']));
EventData = xlsread(['Events' num2str(Condition) '.xlsx']);
[LFO,LFS,RFO,RFS,TimePoint] = RecEvent(EventData);
t = (0:length(AngVel)-1)'*h;
% t2= (h:length(AngAcc))'*h;

% Create function of the gait data
% omega_func    = @(t_i) interp1(t,AngVel,t_i);
% wv_func       = @(t_i) interp1(t,AngVel(:,2),t_i);
% wu_func       = @(t_i) interp1(t,AngVel(:,1),t_i);
% dww_func      = @(t_i) interp1(t,AngAcc(:,3),t_i);

% Create function of the moments and ddgamma
Mcmg_sc        = matlabFunction(M_b_opt,'file','Mcmg_sc');
ddgamma_fun_sc = matlabFunction(ddgamma_eq,'file','ddgamma_fun_sc');

% % Create function handle and use ode15s for numerical integration
% ddgamma_func = @(t,y) ddgamma_fun_b(y(2),dww_func(t),y(1),wu_func(t),wv_func(t));
% [t,y] = ode15s(ddgamma_func,[0 3],[0 1]);

wu1 = AngVel(:,1);

dwu1 = AngAcc(:,1);
```

```
dwv1 = AngAcc(:,2);
dww1 = AngAcc(:,3);

Time = length(wu1)*h;
% Interpolate to improve integration
wu1 = interp1(0:h:(Time-h),wu1,0:h2:(Time-h2),'PCHIP');
wv1 = AngVel(:,2);
wv1 = interp1(0:h:(Time-h),wv1,0:h2:(Time-h2),'PCHIP');
ww1 = AngVel(:,3);
ww1 = interp1(0:h:(Time-h),ww1,0:h2:(Time-h2),'PCHIP');

dwu1 = interp1(0:h:(Time-h),dwu1,0:h2:(Time-h2),'PCHIP');
dwv1 = interp1(0:h:(Time-h),dwv1,0:h2:(Time-h2),'PCHIP');
dww1 = interp1(0:h:(Time-h),dww1,0:h2:(Time-h2),'PCHIP');

TrunkRot = interp1(0:h:(Time),TrunkRot,0:h2:(Time),'PCHIP');
% Create initial conditions
w = zeros(3,length(wu1));
dw = zeros(3,length(wu1));
wu = wu1(1,1);    wv = wv1(1,1);    ww = ww1(1,1);
dwu= dwu1(1,1);   dwv= dwv1(1,1);   dww= dww1(1,1);
ddgamma1 = zeros(1,length(wu1)); dgamma1  = zeros(1,length(wu1)); gamma1   = zeros
    (1,length(wu1));
gamma     = gammaS;
dgamma    = 0;
initial_conditions = [wu;wv;ww;gamma;dgamma];
M_b_opt1 = zeros(3,length(wu1));

ddgamma1(1,1) = ddgamma_fun_sc(dgamma,gamma,wu,wv);
ddgamma = ddgamma1(1,1);
M_b_opt1(1:3,1) = Mcmg_sc(ddgamma,dgamma,dwu,dwv,gamma,wu,wv,ww);
gamma1(1,1) = gamma;
dgamma1(1,1) = dgamma;

% Numerical integration
for i = 2:length(wv1)
    nC = rotx((TrunkRot(i,1)))*roty((TrunkRot(i,2)))*rotz((TrunkRot(i,3)+pi/2));
    w(1:3,i) = nC*[wu1(i);wv1(i);ww1(i)];
    dw(1:3,i)  = nC*[dwu1(i);dwv1(i);dww1(i)];
    wu = w(1,i);    wv = w(2,i);    ww = w(3,i);
    dwu= dw(1,i);   dwv= dw(2,i);   dww= dw(3,i);
    ddgamma1(i) = ddgamma_fun_sc(dgamma,gamma,wu,wv);
    ddgamma = ddgamma1(i);
    dgamma1(i) = dgamma1(i-1) + double(ddgamma1(i)*h2);
    dgamma = dgamma1(i);
    gamma1(i) = gamma1(i-1) + double(dgamma1(i)*h2 + 0.5*ddgamma*h2^2);
    gamma = gamma1(i);

    M_b_opt1(:,i) = Mcmg_sc(ddgamma,dgamma,dwu,dwv,gamma,wu,wv,ww);

    if isnan(ddgamma) == 1
        error('decrease time step')
    end

end
```

```matlab
% Plot Gait Data
GaitEvent = [LFO,LFS,RFO,RFS];
FirstEvent = find(GaitEvent(1,:) == 0);
if FirstEvent == 1
    Tag1 = 'LFO';
    Tag2 = 'LFS';
    Tag3 = 'RFO';
    Tag4 = 'RFS';
elseif FirstEvent == 2
    Tag1 = 'LFS';
    Tag2 = 'RFO';
    Tag3 = 'RFS';
    Tag4 = 'LFO';
elseif FirstEvent ==3
    Tag1 = 'RFO';
    Tag2 = 'RFS';
    Tag3 = 'LFO';
    Tag4 = 'LFS';
elseif FirstEvent == 4
    Tag1 = 'RFS';
    Tag2 = 'LFO';
    Tag3 = 'LFS';
    Tag4 = 'RFO';
end
Tag5 = Tag1;
Tag6 = Tag2;
Tag7 = Tag3;
if TimePoint(1) < 0.1
    Tag1 = '  ' ;
end


figure()
subplot(4,1,1)
plot(0:h2:(Time-h2),M_b_opt1(1,:),'Linewidth',2,'Linestyle','-')
hold on
plot(0:h2:(Time-h2),M_b_opt1(2,:),'Linewidth',2,'Linestyle','-.')
plot(0:h2:(Time-h2),M_b_opt1(3,:),'Linewidth',2,'Linestyle',':')
ylabel('Moment in Nm')
xlim([0.1 Time(end)])
ylim([min(M_b_opt1(:)) max(M_b_opt1(:))])
vline(TimePoint(1),'k');
text(TimePoint(1),max(M_b_opt1(:)),Tag1, 'HorizontalAlignment','center','VerticalAlignment','bottom','FontSize',11)
vline(TimePoint(2),'k');
text(TimePoint(2),max(M_b_opt1(:)),Tag2, 'HorizontalAlignment','center','VerticalAlignment','bottom','FontSize',11)
vline(TimePoint(3),'k');
text(TimePoint(3),max(M_b_opt1(:)),Tag3, 'HorizontalAlignment','center','VerticalAlignment','bottom','FontSize',11)
vline(TimePoint(4),'k');
text(TimePoint(4),max(M_b_opt1(:)),Tag4, 'HorizontalAlignment','center','VerticalAlignment','bottom','FontSize',11)
vline(TimePoint(5),'k');
text(TimePoint(5),max(M_b_opt1(:)),Tag5, 'HorizontalAlignment','center','VerticalAlignment','bottom','FontSize',11)
```

```
vline(TimePoint(6),'k');
text(TimePoint(6),max(M_b_opt1(:)),Tag6, 'HorizontalAlignment','center', '
    VerticalAlignment','bottom','FontSize',11)
vline(TimePoint(7),'k');
text(TimePoint(7),max(M_b_opt1(:)),Tag7, 'HorizontalAlignment','center', '
    VerticalAlignment','bottom','FontSize',11)
legend('$M_u$' , '$M_v$' , '$M_w$','Location','best');

subplot(4,1,2)
plot(0:h2:(Time-h2),ddgamma1,'Linewidth',1.5);
ylabel('$\ddot{\gamma}$ in rad/s$^{2}$')
xlim([0.1 Time(end)])
ylim([mean(ddgamma1)-2.5*std(ddgamma1) mean(ddgamma1)+2.5*std(ddgamma1)])
vline(TimePoint(1),'k');
vline(TimePoint(2),'k');
vline(TimePoint(3),'k');
vline(TimePoint(4),'k');
vline(TimePoint(5),'k');
vline(TimePoint(6),'k');
vline(TimePoint(7),'k');

subplot(4,1,3)
plot(0:h2:(Time-h2),dgamma1,'Linewidth',1.5);
ylabel('$\dot{\gamma}$ in rad/s')
% ylim([mean(dgamma1)-1.5*std(dgamma1) mean(dgamma1)+1.5*std(dgamma1)])
xlim([0.1 Time(end)])
ylim([min(dgamma1(:)) max(dgamma1(:))])
vline(TimePoint(1),'k');
vline(TimePoint(2),'k');
vline(TimePoint(3),'k');
vline(TimePoint(4),'k');
vline(TimePoint(5),'k');
vline(TimePoint(6),'k');
vline(TimePoint(7),'k');

subplot(4,1,4)
plot(0:h2:(Time-h2),gamma1,'Linewidth',1.5);
ylabel('${\gamma}$ in rad')
xlabel('Time in s')
xlim([0.1 Time(end)])
ylim([min(gamma1(:))-0.00000000001 max(gamma1(:))+0.00000000001])
vline(TimePoint(1),'k');
vline(TimePoint(2),'k');
vline(TimePoint(3),'k');
vline(TimePoint(4),'k');
vline(TimePoint(5),'k');
vline(TimePoint(6),'k');
vline(TimePoint(7),'k');

setInterpreter(gcf,'latex');


% figure()
% plot(TimePoint,[LFO,LFS,RFO,RFS],'x','MarkerSize',10,'LineWidth',2)
% hold on
% plot(0:h2:(Time-h2),w,'Linewidth',2)
```

```
% legend('LFO','LFS','RFO','RFS','wu','wv','ww')
end

function [x1,resnorm,Gs,TFdes,x0] = optimization(Gs,TFdes,errorfun)
ub = [4500, 3800, 0.04, 0.02,  0.1];                                % Upper bounds

x0 = [rand(1)*0.01 , rand(1)*0.001 , rand(1)*ub(3), rand(1)*ub(4), rand(1)*pi + rand
    (1)*-pi]; % Initial guess

% XS = [4500, 3800, 0.3, 0.3,  0];                                % Upper
    bounds
% x0 = [rand(1)*0.01 , rand(1)*0.001 , rand(1)*XS(3), rand(1)*XS(4), rand(1)*pi +
    rand(1)*-pi]; % Initial guess
%
% ub = [inf, inf, inf, inf, 0.3];

lb = [0 , 0, 0, 0, -0.3];                                % Lower
    bounds

options = optimoptions(@lsqnonlin,'Algorithm','trust-region-reflective');
options.MaxFunctionEvaluations = 180000;
options.MaxIterations = 12000;
[x1, resnorm] = lsqnonlin(errorfun,x0,lb,ub,options);                                %
    Optimization function
k = x1(1); b = x1(2); Iws = x1(3); Igg = x1(4); gammaS = x1(5); Iwt = 1/2*Iws; Igs =
    1/2*Igg; Igt = Igs;



clear s
syms s
Gs = syms2tf(subs(Gs));

end
```

E.3. Extra Functions

E.3.1. Linearization

```
function [A] = linearization(f,x,gamma,wbn)

gamma = gamma;
gamma0 = gamma;
dgamma = 0;
ddgamma = 0;
ws = wbn(1);
wt = wbn(2);
wg = wbn(2);
wu = wbn(1);
wv = wbn(2);
ww = wbn(3);
dws = 0;
dwt = 0;
dwg = 0;
dwu = 0;
dwv = 0;
dww = 0;
```

```
A = jacobian(f,x);
A = subs(subs(A));


end
```

E.3.2. Compute Transfer Function

```
function sys = comptf(fun,anguler_velocity,angular_axis,Moment,Moment_axis)
eq1 = fun(Moment_axis);
gamma = 0;
if angular_axis == 1
    wt = 0;
    wg = 0;

    wv = 0;
    ww = 0;
elseif angular_axis == 2
    ws = 0;
    wg = 0;

    wu = 0;
    ww = 0;
elseif angular_axis == 3
    ws = 0;
    wt = 0;

    wu = 0;
    wv = 0;
end

if Moment_axis == 1
    Mt = 0;
    Mg = 0;

    Mv = 0;
    Mw = 0;
elseif Moment_axis == 2
    Ms = 0;
    Mg = 0;

    Mu = 0;
    Mw = 0;
elseif Moment_axis == 3
    Ms = 0;
    Mt = 0;

    Mu = 0;
    Mv = 0;
end
eq1 = subs(subs(eq1));

w = solve(eq1 == 0,anguler_velocity);
M_w = Moment/w;
M_w = simplify(subs(M_w));
sys = M_w;
```

```
% if numel(symvar(M_w)) == 0
%       sys = tf(double(M_w),1);
% end
%
% if numel(symvar(M_w)) == 1
%      sys = syms2tf(M_w);
% end
% if numel(symvar(M_w)) > 1
%      sys = 0;
% end


end
```

E.3.3. Bode Plots

```
function BodeGraph(Gs,TFdes)
% Makes a bode plot of two transfer function. For the transfer function Gs
% the poles and zeros will be marked.


Gpole = pole(Gs);
[wn,~] = damp(Gs);
Gzero = zero(Gs);

w = logspace(-4,6,700000);

w = sort([w 0.5], 'ascend');
SkipPole = 0;
if isempty(Gpole) == 1
     SkipPole = 1;
[~,wixZ1] = min(abs(w-abs(Gzero(1))));

elseif isreal(Gpole) == 1

[~,wixP1] = min(abs(w-abs(Gpole(1))));
[~,wixP2] = min(abs(w-abs(Gpole(2))));
[~,wixZ1] = min(abs(w-abs(Gzero(1))));
[~,wixZ2] = min(abs(w-abs(Gzero(2))));
[~,wixZ3] = min(abs(w-abs(Gzero(3))));
else
[~,wixP1] = min(abs(w-abs(wn(1))));
[~,wixP2] = min(abs(w-abs(wn(2))));
[~,wixZ1] = min(abs(w-abs(Gzero(1))));
[~,wixZ2] = min(abs(w-abs(Gzero(2))));
[~,wixZ3] = min(abs(w-abs(Gzero(3))));
end



[magGs,phaseGs] = bode(Gs,w);
phaseGs = wrapTo180(phaseGs);
[magTFdes,phaseTFdes] = bode(TFdes,w);
phaseTFdes = wrapTo180(phaseTFdes);
```

```
figure(1)
subplot(2,1,1)

% Magnitude
loglog(w, squeeze(magGs), 'b','Linewidth',2,'Linestyle','-')
hold on
loglog(w, squeeze(magTFdes),'r','Linestyle','--','Linewidth',2)
ylim([10e-2 10e3]);

% Magnitude Poles
if SkipPole ==1
elseif wixP1 == wixP2
loglog(w(wixP1), magGs(1,1,wixP1),'x','MarkerSize',15,'LineWidth',2,'Color','blue')
loglog(w(wixP2), magGs(1,1,wixP2),'+','MarkerSize',15,'LineWidth',2,'Color','blue')
text(w(wixP2),(max(magGs)*1e2),['p_{1,2}=' num2str(real(Gpole(2)),3) '\pm' num2str(imag(Gpole(2)),3) 'i'], 'HorizontalAlignment','left', 'VerticalAlignment','bottom','FontSize',11)

else
<<<<<<< HEAD:Matlab/Necessary_functions/BodeGraph.m
text(w(wixP1),(max(magGs)*1e2),['p_1=' num2str(Gpole(1),3) ''], 'HorizontalAlignment','center', 'VerticalAlignment','bottom','FontSize',11)
text(w(wixP2),(max(magGs)*1e2),['p_2=' num2str(Gpole(2),3) ''], 'HorizontalAlignment','left', 'VerticalAlignment','bottom','FontSize',11)
=======
text(w(wixP1),(max(magGs)*1e3),['p_1=' num2str(Gpole(1),3) ''], 'HorizontalAlignment','left', 'VerticalAlignment','bottom','FontSize',11)
text(w(wixP2),(max(magGs)*1e3),['p_2=' num2str(Gpole(2),3) ''], 'HorizontalAlignment','right', 'VerticalAlignment','bottom','FontSize',11)
>>>>>>> a87887ad2f846ad954ea31c4f8e904e62f822533:Matlab/BodeGraph.m
loglog(w(wixP1), magGs(1,1,wixP1),'x','MarkerSize',15,'LineWidth',2,'Color','blue')
loglog(w(wixP2), magGs(1,1,wixP2),'x','MarkerSize',15,'LineWidth',2,'Color','blue')

end

% Magnitude Zeros
if Gzero(1) > 0
loglog(w(wixZ1), magGs(1,1,wixZ1),'o','MarkerSize',16,'LineWidth',2,'Color','blue')
end
if wixZ1 == wixZ2
loglog(w(wixZ2), magGs(1,1,wixZ2),'o','MarkerSize',10,'LineWidth',2,'Color','blue')
<<<<<<< HEAD:Matlab/Necessary_functions/BodeGraph.m
text(w(wixZ2),magGs(1,1,wixZ2)*1000000,['z_{1,2}=' num2str(real(Gzero(2)),3) '\pm' num2str(imag(Gzero(2)),3) 'i'], 'HorizontalAlignment','center', 'VerticalAlignment','bottom','FontSize',11)
loglog(w(wixZ3), magGs(1,1,wixZ3),'o','MarkerSize',16,'LineWidth',2,'Color','blue')
text(w(wixZ3),magGs(1,1,wixZ2)*1000000,['z_3=' num2str(Gzero(3),3) ''], 'HorizontalAlignment','left', 'VerticalAlignment','bottom','FontSize',11)

elseif wixZ2 == wixZ3
% text(w(wixZ1),magGs(1,1,wixZ2)*1000000,['z_1=' num2str(Gzero(1),3) ''], 'HorizontalAlignment','center', 'VerticalAlignment','middle','FontSize',11)
loglog(w(wixZ2), magGs(1,1,wixZ2),'o','MarkerSize',10,'LineWidth',2,'Color','blue')
text(w(wixZ2),magGs(1,1,wixZ2)*1000000,['z_{2,3}=' num2str(real(Gzero(2)),3) '\pm' num2str(imag(Gzero(2)),3) 'i'], 'HorizontalAlignment','center', 'VerticalAlignment','bottom','FontSize',11)
```

```
loglog(w(wixZ3), magGs(1,1,wixZ3),'o','MarkerSize',16,'LineWidth',2,'Color','blue')

elseif wixZ1 == wixZ2 && wixZ2 == wixZ3
% text(w(wixZ1),magGs(1,1,wixZ2)*1000000,['z_1=' num2str(Gzero(1),3) ''], 'HorizontalAlignment','center', 'VerticalAlignment','top','FontSize',11)
loglog(w(wixZ2), magGs(1,1,wixZ2),'o','MarkerSize',10,'LineWidth',2,'Color','blue')
text(w(wixZ2),magGs(1,1,wixZ2)*1000000,['z_{2,3}=' num2str(real(Gzero(2)),3) '\pm' num2str(imag(Gzero(2)),3) 'i'], 'HorizontalAlignment','center', 'VerticalAlignment','bottom','FontSize',11)
=======
text(w(wixZ2),(min(magGs)*0.000001),['z_{1,2}=' num2str(real(Gzero(2)),3) '\pm' num2str(imag(Gzero(2)),3) 'i'], 'HorizontalAlignment','center', 'VerticalAlignment','bottom','FontSize',11)
loglog(w(wixZ3), magGs(1,1,wixZ3),'o','MarkerSize',16,'LineWidth',2,'Color','blue')
text(w(wixZ3),(min(magGs)*0.000001),['z_3=' num2str(Gzero(3),3) ''], 'HorizontalAlignment','left', 'VerticalAlignment','bottom','FontSize',11)

elseif wixZ2 == wixZ3
text(w(wixZ1),(min(magGs)*0.000001),['z_1=' num2str(Gzero(1),3) ''], 'HorizontalAlignment','center', 'VerticalAlignment','middle','FontSize',11)
loglog(w(wixZ2), magGs(1,1,wixZ2),'o','MarkerSize',10,'LineWidth',2,'Color','blue')
text(w(wixZ2),(min(magGs)*0.000001),['z_{2,3}=' num2str(real(Gzero(2)),3) '\pm' num2str(imag(Gzero(2)),3) 'i'], 'HorizontalAlignment','center', 'VerticalAlignment','bottom','FontSize',11)
loglog(w(wixZ3), magGs(1,1,wixZ3),'o','MarkerSize',16,'LineWidth',2,'Color','blue')

elseif wixZ1 == wixZ2 && wixZ2 == wixZ3
text(w(wixZ1),(min(magGs)*0.000001),['z_1=' num2str(Gzero(1),3) ''], 'HorizontalAlignment','center', 'VerticalAlignment','top','FontSize',11)
loglog(w(wixZ2), magGs(1,1,wixZ2),'o','MarkerSize',10,'LineWidth',2,'Color','blue')
text(w(wixZ2),(min(magGs)*0.000001),['z_{2,3}=' num2str(real(Gzero(2)),3) '\pm' num2str(imag(Gzero(2)),3) 'i'], 'HorizontalAlignment','center', 'VerticalAlignment','bottom','FontSize',11)
>>>>>>> a87887ad2f846ad954ea31c4f8e904e62f822533:Matlab/BodeGraph.m
loglog(w(wixZ3), magGs(1,1,wixZ3),'o','MarkerSize',20,'LineWidth',2,'Color','blue')
elseif wixZ1 == 1

loglog(w(wixZ2), magGs(1,1,wixZ2),'o','MarkerSize',16,'LineWidth',2,'Color','blue')
text(w(wixZ2),magGs(1,1,wixZ2)*1000000,['z_2=' num2str(Gzero(2),3) ''], 'HorizontalAlignment','center', 'VerticalAlignment','bottom','FontSize',11)

loglog(w(wixZ3), magGs(1,1,wixZ3),'o','MarkerSize',16,'LineWidth',2,'Color','blue')
text(w(wixZ3),magGs(1,1,wixZ2)*1000000,['z_3=' num2str(Gzero(3),3) ''], 'HorizontalAlignment','center', 'VerticalAlignment','bottom','FontSize',11)

else

loglog(w(wixZ2), magGs(1,1,wixZ2),'o','MarkerSize',16,'LineWidth',2,'Color','blue')
<<<<<<< HEAD:Matlab/Necessary_functions/BodeGraph.m
text(w(wixZ1),magGs(1,1,wixZ2)*1000000,['z_1=' num2str(Gzero(1),3) ''], 'HorizontalAlignment','center', 'VerticalAlignment','bottom','FontSize',11)
text(w(wixZ2),magGs(1,1,wixZ2)*1000000,['z_2=' num2str(Gzero(2),3) ''], 'HorizontalAlignment','center', 'VerticalAlignment','bottom','FontSize',11)

loglog(w(wixZ3), magGs(1,1,wixZ3),'o','MarkerSize',16,'LineWidth',2,'Color','blue')
text(w(wixZ3),magGs(1,1,wixZ2)*1000000,['z_3=' num2str(Gzero(3),3) ''], '
```

```
HorizontalAlignment','center', 'VerticalAlignment','bottom','FontSize',11)
=======
text(w(wixZ1),(min(magGs)*0.001),['z_1=' num2str(Gzero(1),3)''], 'HorizontalAlignment','center', 'VerticalAlignment','bottom','FontSize',11)
text(w(wixZ2),(min(magGs)*0.000001),['z_2=' num2str(Gzero(2),3)''], 'HorizontalAlignment','center', 'VerticalAlignment','bottom','FontSize',11)

loglog(w(wixZ3), magGs(1,1,wixZ3),'o','MarkerSize',16,'LineWidth',2,'Color','blue')
text(w(wixZ3),(min(magGs)*0.000001),['z_3=' num2str(Gzero(3),3)''], 'HorizontalAlignment','center', 'VerticalAlignment','bottom','FontSize',11)
>>>>>>> a87887ad2f846ad954ea31c4f8e904e62f822533:Matlab/BodeGraph.m


end

vline((0.01)*2*pi,'k');
vline(10*2*pi,'k') ;


grid
xlabel('Frequency in rad/s')
ylabel('Magnitude in dB')
% Phase
subplot(2,1,2)
semilogx(w, squeeze(phaseGs),'b','Linewidth',2)
hold on
semilogx(w, squeeze(phaseTFdes),'r','Linewidth',2,'Linestyle','--')
% Phase Markers
semilogx(w(wixP1), phaseGs(1,1,wixP1),'x','MarkerSize',15,'LineWidth',2,'MarkerEdgeColor','b')
if wixP1 == wixP2
semilogx(w(wixP2), phaseGs(1,1,wixP2),'+','MarkerSize',15,'LineWidth',2,'MarkerEdgeColor','b')
else
semilogx(w(wixP2), phaseGs(1,1,wixP2),'x','MarkerSize',15,'LineWidth',2,'MarkerEdgeColor','b')
end

% semilogx(w(wixZ1), phaseGs(1,1,wixZ1),'o','MarkerSize',16,'LineWidth',2,'Color','blue')
if wixZ1 == wixZ2 || wixZ2 == wixZ3
semilogx(w(wixZ2), phaseGs(1,1,wixZ2),'o','MarkerSize',10,'LineWidth',2,'Color','blue')
else
semilogx(w(wixZ2), phaseGs(1,1,wixZ2),'o','MarkerSize',16,'LineWidth',2,'Color','blue')
end
semilogx(w(wixZ3), phaseGs(1,1,wixZ3),'o','MarkerSize',16,'LineWidth',2,'Color','blue')
vline((0.01)*2*pi,'k');
vline(10*2*pi,'k') ;

grid
xlabel('Frequency in rad/s')
ylabel('Phase in deg')
legend('M_v/\omega_v','TFdes');
```

```
h = gca;
h.YTick = -180:90:180;

end
```