NATIONAL TECHNICAL UNIVERSITY OF ATHENS
School of Mechanical Engineering
Department of Mechanical Design and Control Systems
Control Systems Lab

Diploma Thesis

A Method for Designing High Performance Two-Segment Legs for Running Quadruped Robots

Spyridon Dallas

Supervising Professor: E.G. Papadopoulos

ATHENS 2017

Abstract

Quadruped animals have been on the side of man for thousands of years, providing various valuable services, such as fast and reliable transportation through difficult terrain, tracking and rescuing victims of natural disasters or protection of man and his property. Quadrupeds move with great agility in domestic or rugged terrain, exploiting isolated footholds to ensure stability of movement. In the last decade, the impressive evolution in the field of articulated robots has lead researchers to recognize the potential benefits of constructing quadruped robots; robots that could offer many of the services quadruped animals do and even more. Although wheeled or tracked systems are alternatives widely used for terrestrial locomotion, their inability to ensure traction in difficult terrain (steep inclinations or loose grounds), has intensified the interest of researchers in their legged counterparts.

To this day, many impressive quadruped robots have been constructed; robots that can run fast, jump over obstacles, carry heavy loads, climb stairs or jump over fences, walk on ice. These achievements depend strongly on empirical data gained from systematic hardware experimentation and meticulous control strategies. Nevertheless, as the nature of the quadruped locomotion is highly challenging, it still remains unclear how the morphology of the robot - and expecially that of the legs - can affect its performance. For this reason, many of the existing leg design approaches are not systematic. Many research teams mimic the leg morphologies found in nature. However, the actuation systems of animals and the tasks they undertake are much different than these of the robots. Other teams base their design efforts on kinematic performance criteria. Quadrupedal locomotion however is a highly dynamic phenomenon, with alternating load phases and significant accelerations required from the actuating systems. A few teams use sophisticated robot descriptions and control strategies, but their approaches are so complicated that only a few leg design alternatives can be evaluated.

In this thesis, a systematic methodology is devised aiming to determine the proper leg attributes that maximize the running performance of the *NTUA Laelaps* quadruped robot. The proposed method takes into account the robot's mass and inertia, the desired leg architecture and leg material properties, the available actuation system, certain terrain properties and the gait in which the robot shall move. Through a three stage parametric search, the optimal leg morphology is found (length of leg segments and leg compliance) for achieving maximum velocity of locomotion, subjected to actuation and material strength constraints. For the optimal leg morphology, control indications for achieving maximum running velocity are also found, related with the positions of the footfalls, the apex of the robot's trajectory and the stride period.

The method is applied with alternative gaits and leg joint configurations as inputs and the optimal solutions found for each case are compared. The results are validated using models and controller independent to the method . For the gait and joint configuration that enhances the running performance of the robot further tests are conducted, to determine which of the available motors should be acting on which actuated joint. Moreover, the effect of supplementary reduction ratios in the running performance of the robot is studied. Finally, a tipover stability criterion is introduced and conditions are found for the robot to run in maximum velocity, withstanding terrain perturbations in the frontal plane of movement.

Περίληψη

Τα τετράποδα ζώα βρίσκονται στο πλευρό του ανθρώπου εδώ και χιλιάδες χρόνια, προσφέροντας του πολύτιμες υπηρεσίες όπως τη γρήγορη και αξιόπιστη μεταφορά σε δύσκολα εδάφη, τον εντοπισμό και διάσωση θυμάτων σε φυσικές καταστροφές ή την προστασία αυτού και της περιουσίας του. Τα τετράποδα κινούνται με μεγάλη ευκολία στο φυσικό ή σε ανθρώπινα διαμορφωμένο περιβάλλον, εξασφαλίζοντας ευστάθεια στην κίνηση τους μέσω μεμονωμένων πατημάτων. Την τελευταία δεκαετία, η εντυπωσιακή πρόοδος στον κλάδο των αρθρωτών ρομπότ, οδήγησε την επιστημονική κοινότητα να αναγνωρίσει τα οφέλη που τα τετράποδα ρομπότ δυνητικά μπορούν να προσφέρουν στον άνθρωπο· οφέλη πιθανώς πολύ μεγαλύτερα από αυτά που τα ζώα μπορούν να του παρέχουν. Αν και τα τροχοφόρα ή ερπυστριοφόρα οχήματα είναι ευρέως χρησιμοποιούμενες εναλλακτικές λύσεις στη χερσαία μετακίνηση, η αδυναμία τους να διασφαλίσουν πρόσφυση σε δύσκολο έδαφος (απότομες κλίσεις ή σαθρή γεωλογία) εντατικοποίησε το ενδιαφέρον των ερευνητών στα τετράποδα ανάλογα τους.

Μέχρι σήμερα, πολλά εντυπωσιακά τετράποδα ρομπότ έχουν κατασκευαστεί· ρομπότ τα οποία τρέχουν σε υψηλές ταχύτητες, υπερπηδούν εμπόδια, μεταφέρουν βαριά φορτία, σκαρφαλώνουν σε σκάλες ή πηδούν πάνω από φράχτες και διατηρούν την ευστάθεια τους στον πάγο. Πολλά από αυτά τα κατορθώματα οφείλονται σε εμπειρικά δεδομένα λόγω συστηματικού πειραματισμού και σε προσεχτικά σχεδιασμένες στρατηγικές ελέγχου. Παρόλα αυτά, καθώς η τετράποδη κίνηση ενέχει εκ φύσεως πολλές προκλήσεις, παραμένει αβέβαιο πώς η μορφολογία του ρομπότ - και ιδιαίτερα των ποδιών του - μπορεί να επηρεάσει τις επιδόσεις του. Έτσι, πολλές από τις υπάρχουσες προσεγγίσεις σχεδιασμού ποδιών παραμένουν μη συστηματικές. Πολλές ερευνητικές ομάδες μιμούνται τις μορφολογίες ποδιών που απαντώνται στη φύση. Ωστόσο, τα ζώα ούτε διαθέτουν τους ίδιους επενεργητές με τα ρομπότ, αλλά ούτε και απαραιτήτως καταπιάνονται με τα ίδια καθήκοντα. Άλλες ομάδες βασίζουν τις σχεδιαστικές τους προσπάθειες σε κριτήρια κινηματικής, παρά το γεγονός ότι η τετράποδη κίνηση είναι εντόνως δυναμική, με απότομες αλλαγές στο φορτίο και υψηλές απαιτήσεις επιτάχυνσης στους επενεργητές. Αν και ορισμένες άλλες προσεγγίσεις περιλαμβάνουν περίπλοκα μοντέλα ρομπότ και ελεγκτές στη διαδικασία σχεδιασμού, οι προσεγγίσεις τους είναι τόσο σύνθετες που επιτρέπουν τη σύγκριση και αξιολόγιση περιορισμένων εναλλακτικών μορφολογιών ποδιών.

Στην παρούσα διπλωματική εργασία, προτείνεται μια συστηματική μεθοδολογία σχεδιασμού ποδιών τετράποδων ρομπότ, με στόχο την αναγνώριση εκείνων των στοιχείων σχεδιασμού που θα βελτιστοποιήσουν την επίδοση τρεξίματος του τετράποδου *Λαίλαψ* του ΕΜΠ. Η προτεινόμενη μέθοδος λαμβάνει υπόψιν τη μάζα και την αδράνεια του ρομπότ, την επιθυμητή δομή του ποδιού και τις ιδιότητες των υλικών του, το διαθέσιμο σύστημα επενέργησης, εκτιμώμενες ιδιότητες εδάφους και τον τύπο βαδίσματος με το οποίο το ρομπότ μετακινείται. Μέσω μιας παραμετρικής αναζήτησης τριών σταδίων, καθορίζεται η βέλτιστη μορφολογία ποδιών (μήκη τμημάτων ποδιού και ελαστικότητα ποδιού) για επίτευξη μέγιστης ταχύτητας κίνησης υπο τους υπάρχοντες περιορισμούς επενέργησης και αντοχής υλικών. Για τη βέλτιστη μορφολογία ποδιών, βρίσκονται επιπλέον ενδείξεις ελέγχου, συσχετιζόμενες με την τοποθέτηση των ποδιών στο έδαφος, το άνω σημείο της θέσης του σώματος του ρομπότ και τη περίοδο δρασκελισμού, ώστε το ρομπότ να τρέχει με τη μέγιστη δυνατή ταχύτητα.

Η μέθοδος εφαρμόζεται για διαφορετικά βαδίσματα και διαμορφώσεις αρθρώσεων γονάτου και οι βέλτιστες λύσεις για κάθε περίπτωση υποβάλλονται σε σύγκριση. Τα αποτελέσματα επαληθεύονται χρησιμοποιώντας μοντέλα και ελεκτή ανεξάρτητα της μεθόδου. Για το βάδισμα και τη διαμόρφωση αρθρώσεων που μπορεί να επιτύχει τη μέγιστη ταχύτητα, εξετάζεται επιπλέον σε ποια άρθρωση πρέπει να επιδρά ο καθένας από τους διαθέσιμους επενεργητές ώστε να βελτιωθεί περαιτέρω η επίδοση τρεξίματος του ρομπότ. Εξετάζεται δε και ο ρόλος της επιπλέον μείωσης στους κινητήρες. Τέλος, εισάγοντας ένα κριτήριο ευστάθειας έναντι ανατροπής, αναγνωρίζονται οι συνθήκες ώστε το ρομπότ να τρέχει στη μέγιστη ταχύτητα έχοντας τη δυνατότητα να αντισταθεί σε διαταραζές εδάφους στο μετωπιαίο επίπεδο κίνησης.

Acknowledgements

I would like to thank my supervisor Professor E. Papadopoulos for his guidance throughout the elaboration of this thesis. His understanding of robotics, his advices and constructive criticism and high work standards contributed significantly to the quality of this thesis and the carving of my character.

Furthermore, I would like to thank the PhD candidates and good friends of mine K. Koutsoukis and K. Machairas. Their highly motivated hard working profile and their strong interest in legged robotics set an inspiring example for me. Their encouragement and support in the elaboration of this thesis is highly appreciated.

Finally, I'd like to thank the rest of the Control System Lab members for providing a pleasant work environment for me. Their friendly predespodition and eagerness for cooperation helped me a lot throughout my research efforts.

Dedicated to those who supported my efforts,
Family, Friends and Teachers

Table of Contents

List of Figures

List of Tables

Nomenclature

γ_1 angle between the total force exerted on the CoM and the CoM to toe vector

γ_2 angle between the total force exerted on the CoM and the robot weight vector

γ_ε ground slope in the frontal plane

δ_y ground identation of Hunt - Crossley contact model

δt_s stance phase duration

$\delta l_{2,i}$ deformation of the spring of the i-th leg

ε a parameter determining the search interval of the leg space exhaustive search

θ robot body pitch angle

$\theta_{1,i}$ hip joint angle of the i-th leg

$\theta_{2,i}$ knee joint angle of the i-th leg

$\theta_{\max}$ maximum pitch observed during the stride period

$\dot{\theta}_{\max,j}$ maximum permissible angular velocity of the DC motor acting on the j-th leg joint

λ stride length

μ Coulomb static friction coefficient

μ_c kinematic friction coefficient

ν Hertzian contact coefficient

ρ density of the tubular leg segments

$\sigma_{b,j,i}$ bending stress applied on the j-th tubular leg segment of the i-th leg

$\sigma_{c,j,i}$ compressive stress applied on the j-th tubular leg segment of the i-th leg

$\boldsymbol{\tau}$ vector containing the actuation torques exerted on every joint of every leg

$\tau_{1,i}$ actuation torque exerted on the hip joint of the i-th leg

$\tau_{2,i}$ actuation torque exerted on the knee joint of the i-th leg

τ_θ the sum of the torques acting on the CoM at every time instant

$\tau_{ct,j}$ maximum permissible continuous torque of the DC motor acting on the j-th leg joint

$\tau_{s,j,i}$ shear stress applied on the j-th tubular leg segment of the i-th leg

$\tau_{s,u}$ ultimate shear strength of the leg segments' material

$\tau_{st,j}$ maximum permissible short term torque of the DC motor acting on the j-th leg joint

$\varphi_{e,i}$ phase difference that determines the initial position of the toe on an elliptical toe trajectory

$\varphi_{x,i}$ phase difference of the horizontal sinusoidal ground force $F_{x,i}$

$\varphi_{y,i}$ phase difference of the vertical sinusoidal ground force $F_{y,i}$

ω_e circular frequency in which the elliptical toe trajectory is traversed

$\omega_{x,i}$ circular frequency of the horizontal sinusoidal ground force $F_{x,i}$

$\omega_{y,i}$ circular frequency of the vertical sinusoidal ground force $F_{y,i}$

a height of the rectangular robot body

a_e horizontal semi-axis of the elliptical toe trajectory

a_{fx} a parameter determining the size of $F_{x,\max}$, $a_{fx} \in [0,1]$

A	cross sectional area of the tubular leg segments
b	length of rectangular robot body
b_g	damping of Hunt - Crossley ground model
b_e	vertical semi-axis of the elliptical toe trajectory, equal to the leg clearance
b_l	damping of the leg spring
c_i	a variable that denotes contact of the i-th leg with the ground, $c_i \in \{0,1\}$
$\mathbf{C}$	centrifugal - Coriolis terms matrix of the quadruped robot's dynamics
CoM	center of mass
d	half hip separation distance in the sagittal plane
d_1	moment arm of the total force exerted on the robot's CoM, w.r.t the toe - ground contact point
d_{in}	inner diameter of the tubular leg segments
d_{out}	outer diameter of the tubular leg segments
DC	direct current
DF	duty factor
EoM	equations of motion
$\mathbf{F}$	vector of the external forces applied to the quadruped robot's system
$\mathbf{F}_i$	vector containing the horizontal and vertical force exerted on the i-th leg, $\mathbf{F}_i = [F_{x,i}\ F_{y,i}]^T$
F_n	normal Hunt - Crossley ground force
$F_{l,j,i}$	ground force component in the longitudinal to the j-th leg segment of the i-th leg direction
F_p	horizontal perturbation force in the frontal plane
$F_{s,i}$	force introduced by the deflection of the ideal spring of the i-th leg
F_t	friction force
$F_{v,j,i}$	ground force component in the vertical to the j-th leg segment of the i-th leg direction
F_w	robot weight
F_x	sum of the horizontal forces acting on the CoM at every time instant
$F_{x,i}$	horizontal component of the ground force exerted on the i-th leg
$F_{x,\max}$	amplitude of the horizontal ground force
$F_{x,ub}$	upper bound of $F_{x,\max}$, for which no slippage occurs
F_y	sum of the vertical forces acting on the CoM at every time instant
$F_{y,i}$	vertical component of the ground force exerted on the i-th leg
$F_{y,\max}$	amplitude of the vertical ground force
FL	front left
FR	front right
HL	hind left
HR	hind right
i	subscript denoting the i-th leg of the robot, where $i \in \{1,2,3,4\}$
i_a	intensity of the current on the field windings of a DC motor
I	total quadruped robot inertia

I_1 inertia of the proximal leg segment

I_2 inertia of the distal leg segment

I_a second moment of area of the tubular leg segments

I_b quadruped robot body inertia

I_{rj} rotor inertia before gearbox reduction of the motor actuating on the j-th leg joint

j subscript denoting the j-th joint, where $j \in \{1,2\}$ for the hip and knee joints respectively

$\mathbf{J}$ Jacobian matrix of the quadruped robot's system

k stiffness of the compliant distal leg segment

k_g ground stiffness of Hunt - Crossley contact model

k_p P term gain of P-V controller for trotting

k_v V term gain of P-V controller for trotting

$\mathbf{K}$ stiffness matrix of the quadruped robot's system

K_T torque constant of a DC motor

KB knee pointing backward leg configuration

KF knee pointing forward leg configuration

l_1 length of the proximal leg segment of any leg

l_{20} free length of the compliant distal leg segment

$l_{2,i}$ length of the distal compliant leg segment of the i-th leg

l_{cm1} distance between the hip joint and the CoM of the proximal leg segment

$l_{cm2,i}$ distance between the knee joint and the CoM of the distal leg segment of the i-th leg

$l_{h2t,i}$ hip to toe distance of the i-th leg

$l_{\max}$ maximum hip to toe distance of any leg observed in one stride period T

lb (as a subscript) lower bound

L Lagrangian

LISL long inequally segmented leg

m total quadruped robot mass

m_1 mass of the proximal leg segment

m_2 mass of the distal leg segment

m_b quadruped robot body mass

m_{j1} mass of the hip joint

m_{j2} mass of the knee joint

m_{l1} mass of the proximal tubular leg segment

m_{l2} mass of the distal tubular leg segment

m_s spring mass

m_t toe mass

$\mathbf{M}$ mass matrix of the quadruped robot's system

n n-th discrete time step, $t_n \in [0,T]$, $n \in [1,N]$

n_t transmission ratio of the gearbox of a DC motor

n_j	transmission ratio of the gearbox of a DC motor actuating on the j-th leg joint
$n_{j,e}$	supplementary transmission ratio at the exit of the motor's gearbox
N	number of discrete time instants in the duration of a stride period T
O	the origin $(0,0)$
P	power from actuation torques and external forces pumped into the quadruped robot's system
P_{th}	thermal losses power on the field windings of a DC motor
$\mathbf{q}$	quadruped robot state vector
$\mathbf{q}_C$	vector of CoM position and pitch, $\mathbf{q}_C = [x\ y\ \theta]$
$\mathbf{q}_i$	vector of joint variables of the i-th leg
$\boldsymbol{r}_i$	vector connecting the CoM position (x, y) with the position of the toe during stance phase
s_f	safety factor
$S_{c,u}$	ultimate compressive strength of the leg segments' material
SESL	short equally segmented leg
SLIP	spring loaded inverted pendulum
t	time
$t_{mf,i}$	time instant of midflight of the i-th leg
$t_{td,i}$	time instant of touch down of the i-th leg
$t_{td,i}\%$	time instant of touch down of the i-th leg, expressed as a percentage of the stride period T
$t_{to,i}$	time instant of take off of the i-th leg
$t_{to,i}\%$	time instant of take off of the i-th leg, expressed as a percentage of the stride period T
T	stride period
T_e	period in which the elliptical toe trajectory is traversed
T_n	free oscillation period of a system consisting of two springs in parallel and a mass
T_L	kinetic energy of the quadruped robot
u_e	threshold velocity for which toe to ground slippage occurs
u_s	Stribeck velocity
ub	(as a subscript) upper bound
U_L	potential energy of the quadruped robot
$\overline{v}$	mean quadruped robot velocity during the stride
w	half hip separation distance in the frontal plane
x	abscissa of the robot body CoM
$x_{1,i}$	abscissa of the CoM of the proximal leg segment
$x_{2,i}$	abscissa of the CoM of the distal leg segment
x_i	abscissa of the footfall of the i-th leg
$x_{i,es}$	optimal footfall abscissas at the end of the exhaustive search of Stage 1
$x_{i,es}$	optimal footfall abscissas at the end of the interior point optimization scheme of Stage 1
$x_{h,i}$	abscissa of the hip joint of the i-th leg
$x_{hr,i}$	relative hip to toe abscissa of the i-th leg

$x_{t,i}$ abscissa of the toe of the i-th leg

y ordinate of the robot body CoM

$y_{1,i}$ ordinate of the CoM of the proximal leg segment

$y_{2,i}$ ordinate of the CoM of the distal leg segment

$y_{h,i}$ ordinate of the hip joint of the i-th leg

$y_{hr,i}$ relative hip to toe ordinate of the i-th leg

y_i ordinate of the footfall of the i-th leg, ordinate of the ground level

$y_{t,i}$ ordinate of the toe of the i-th leg

1 Introduction

1.1 Motivation

During the last years, research in the field of terrestrial locomotion has focused in the opportunities offered by legged robots, mainly for two reasons. On one hand, legged robots pose great advantages in comparison to wheeled or tracked systems [1]. While the latter only have access to less than half of the Earth's land, legged systems can also adapt to uneven or rugged terrain, by utilizing active suspension and isolated footholds, with minimum impact to soil or vegetation, much as most animals do. Having that in mind, legged robots can come in use in many of man's endeavors, such as in rescue or exploration missions in hazardous or until now inaccessible areas. On the other hand, since Raibert built the *Quadruped* in 1990 - one of the first robots that could efficiently move and perform gaits similar to quadruped mammals - it became clear that complex legged locomotion was far from unachievable [2]. Raibert's *Quadruped* opened the way for the creation of many dynamically stable legged robots that appeared next, by providing important insights for design, modeling and control.

Since then, a combination of animal studies with breakthroughs in design and control has led to an overall progress in legged locomotion. Recent impressive results, both in academia and industry, showed that legged robots can run really fast (*Boston Dynamics Cheetah* reached an in lab record velocity of approximately 13 m/s, [3]), perform agile jumps while running (The *MIT Cheetah II* can jump over obstacles up to 80% of its leg length while running at 2.5 m/s, [4]), turn and perform complex gaits on inclined terrain, or maintain stability while on ice (*Boston Dynamics* Spot and *WildCat*, [5] - [6]), cross rugged terrain and perform stunts, (*RHEX* can perform flips, operate on both of its sides, climb stairs, jump across gaps and is waterproof [7]), climb walls and trees (*RiSE Climbing Robot* [8]).

Despite the great progress in legged locomotion, many design issues still remain to be addressed, that concern stability, energy efficiency and high performance. While several legged robots, have achieved energy efficiency comparable to this of animals (*iSprawl* [9], *MIT Cheetah* robot [10]), their performance is still inferior when compared with the animals' incredible capabilities. For instance, no existing legged robot can compete the cheetah's (*Acinonyx jubatus*) acceleration or maximum speed (29 m/s), or the mountain goat's (*Oreamnos americanus*) and Nubian ibex's (*Capra nubiana*) agility in climbing steep rocky mountain sides.

The ever existing limitations of hardware in comparison to biology (control, actuation, materials), motivates the robotics community to continue searching towards smart design solutions that can enhance further quadrupedal locomotion. One of the most challenging structural parts for quadruped robots has proved to be the mean of interaction with the tranversed terrain, i. e. the leg. During quadrupedal locomotion, legs have to consistently endure shock loads from the ground and are the means through which the actuators provide energy to the system. Intrigued by the challenging nature of legs, in the current thesis an appropriate leg design is sought, that can enhance the performance of the quadruped platform *NTUA Laelaps* in one of its most basic functions, running. In more detail, an optimization - based method is proposed that given the running task and the actuation constraints,

provides as outputs optimal dimensions and compliance for the leg, so that the robot can reach maximum horizontal velocity.

1.2 Literature review

Many efforts have been made in the direction of leg design for quadruped robots. The main points of interest in designing robot legs seem to be leg segmentation, leg segment proportions, leg configuration, leg compliance scheme, actuation and construction materials. In the following review, representative examples of surveys, methods, design choices and implementations on the above issues are presented, in order to understand former work done, draw design principles and suggest improvements.

The term *leg segmentation* refers to the number of joints and links from which the leg is composed, see Figure 1-1. The simplest case is that of the prismatic leg. This leg consists of an actuated rotary hip joint and a flexible prismatic joint. Such legs where utilized in early quadruped robots, as the *Quadruped* and the *Scout II* [11] robots, see Figure 1-2. While these legs pose great advantages when it comes to energy efficiency as they exploit passive dynamics, [12], [13], their main disadvantage is their difficulty in creating adequate toe to ground clearance, especially at high speeds. An alternative to prismatic legs is the segmented or the articulated leg. Two-segment legs are employed by most quadruped robots of *Boston Dynamics*, and also by *StarlETH*, [14] and *HyQ*, [15] quadruped robots, see Figure 1-3. As described in [16], two-segment legs in comparison with prismatic compliant legs provide self-stable running for a greater range of forward running velocities and promote stability against perturbations in the angle of attack of the leg and the apex height of the robot. Witte et. al. [17]- [18] suggested that for a three-segment leg model applied in small mammals, the most proximal and distal segments are held nearly in parallel most of the time during the contact of the leg with the ground, a behavior described as “pantograph behavior”. Inspired maybe by similar works in biology, *MIT Cheetah II* and the *Cheetah Cub* [19] robots have three-segment “pantograph” legs. Lately, five bar legs have appeared as in the case of the *Super Mini Cheetah* [20], *ATRIAS* [21] and *MiniTaur* [22] robots. As described in [23], five bar legs, driven by two motors in parallel, pose an advantage when it comes to force application, because both motors contribute in retraction or protraction of the leg, exerting opposing torques. On the other hand, to swing the leg when the robot walks or runs, both motors move in the same direction, causing one to produce positive and the other negative work, with energy cycling between two motors. As a result, energy is repeatedly lost in the transmissions. In a comparison conducted in [22], it was shown that symmetric five bar legs have improved performance when it comes to manipulability measures, [24].

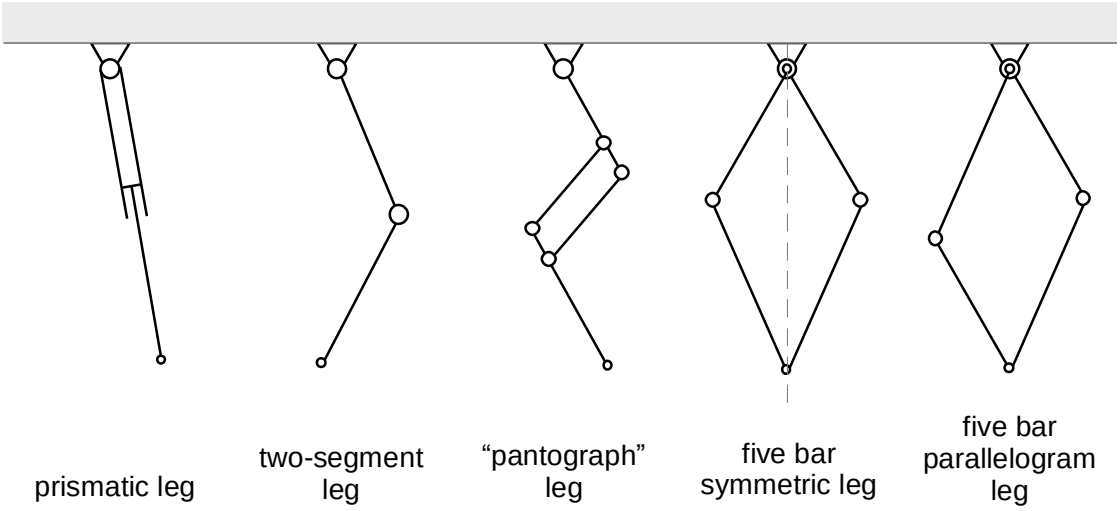


Figure 1-1. Alternative quadruped robot legs.

Figure 1-2. Left to right, the *Quadruped*, [25] and *Scout II*, [26] quadruped robots.

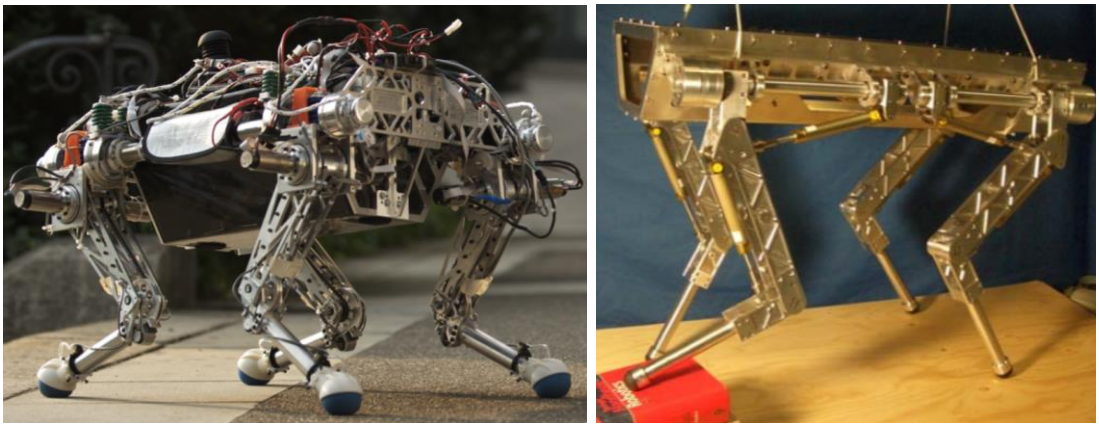


Figure 1-3. Left to right, the *StarlETH*, [14] and *HyQ*, [15] quadruped robots.

Leg proportioning refers to the relative length of the leg segments with respect to the total leg length. Many teams have chosen leg lengths and their proportions based on biological data. The proportions of the *Cheetah Cub* robot leg were taken similar to those of small cat like animals (felidae), from data presented in [27]. The *HyQ* team decided on their robot's leg length, based on observations from dog and horse breeders, [28]. To their words in [15], [comparing body length with leg length], "*A square shape seems to be a good compromise between agility, endurance and strength to carry loads.*" The proportions of the leg were taken equal for the sake of simplicity. The leg segment lengths of ScarlETH, the monopod predecessor of *StarlETH* quadruped, were chosen to be equal. The only explanation found in published work for this choice is that, *"With segment lengths of 0.2 m the final prototype is compact, lightweight and can thus be handled safely by a single person, yet is strong enough to carry various sensor units."*, [29] and also referring to *StarlETH* this time, *"With a body length of about 0.5m, segment lengths of 0.2m, and a total weight of 23kg, this robot has roughly the overall dimension of a medium-sized dog"*, [14]. Other teams have devised different methods to determine the leg length of their quadruped robot. Chatzakos investigated the role of leg length (prismatic legs) in the specific resistance of locomotion (a metric of consumed power for running in a particular speed) for the bounding gait, [30]. The *MIT Cheetah* team, decided upon fixed leg proportions, but adjusted the shoulder height based on a force map method, in which the leg was decoupled from the body, [31]. The proportions of the five bar leg of the *Super Mini Cheetah*, where selected taking into account the desired leg workspace and a Jacobean based estimation of the necessary ground forces for running, [20]. The proportions and leg length of the *MiniTaur*, were chosen to optimize proprioceptive sensitivity, force production and thermal cost of force (Jacobian based manipulability metrics), in a wide portion of a selected workspace. Also in this case, the leg was optimized decoupled from the body.

With the term *leg configuration*, we refer mainly to the orientation of the leg segments in the leg chain, see Figure 1-4. Some works have been published on optimum leg configuration, mainly for two-segment and three-segment legs. For trotting robots with two-segment legs, it was suggested that the orientation of the knee joints of the front and the back legs inwards improves self-stability by reducing pitching moments [32], [33]. With the consensus used, the knee joint is the rotary joint connecting the proximal to the body leg segment, with the distal to the body leg segment, for the two-segment leg. Furthermore, in [34] again for a trotting gait, it was observed that centrosymmetric joint configurations (both knee joints pointing inward or outward) are beneficial for slippery terrain and can improve stability. Apart from reduced pitching angles, elbow and knee joint inwards orientation, is considered to result in smaller roll angles during trotting, [35]. In [36] knee forward and knee backward configurations (pointing backward or forward with respect to the direction of the motion) of a free falling two-segment planar leg model were compared, with criterion the impact losses during collision with the ground. The paper concluded in favor of the knee backward configuration. Based on findings from [33], [34] the *HyQ* team decided on a knee inward or X configuration (knee of hind leg pointing forward, knee of front leg pointing backward) as in Figure 1-3. The *StarlETH* team on the other hand, due to lack of experimental evidence was not convinced on the validity and generality of the above results and conclusions, as to their words

"To our best knowledge, experimental studies with a sophisticated platform are still missing", [14]. That led the team to decide upon a modular leg design, so that every leg can be placed in any of the two possible configurations. For the three-segment leg model, the configuration is based on biological data collected utilizing cineradiography on therian mammals. A detailed review is presented in [37]. While the legs of therian mammals consist of many segments and joints, it is suggested that only three segments per leg contribute mainly to the kinematics of the forward progression, the thigh, shank and a third element that consists of the foot and toes for the hind leg and the scapula, upper arm and a third segment that consists of the lower arm and the hand for the front leg, see Figure 1-5. The *Cheetah Cub* and *MIT Cheetah II* have legs with the shoulder/knee joint pointing forward and the ankle/elbow joint pointing backward, see Figure 1-6. Finally, to the best of our best knowledge, five bar leg configurations are mainly found in a joint outwards configuration to avoid collision while the legs are compressed, see Figure 1-7.

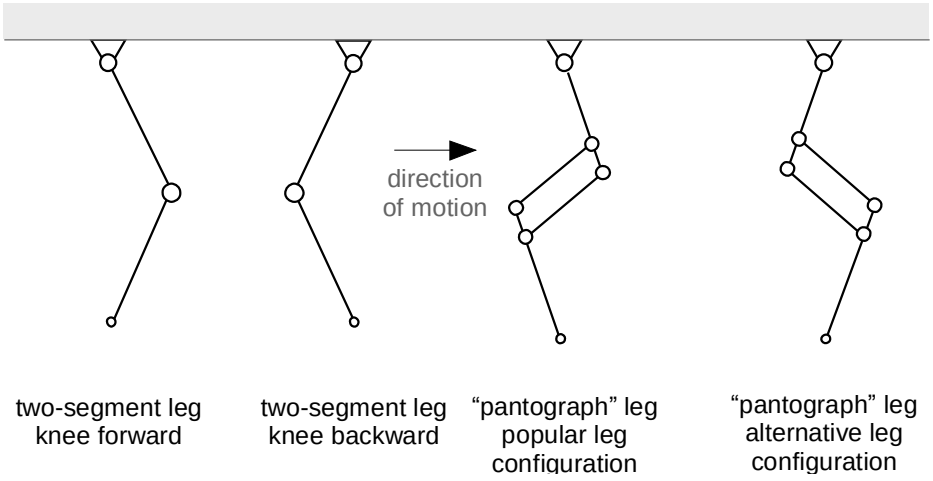


Figure 1-4. Leg configurations of the two-segment and the "pantograph" leg.

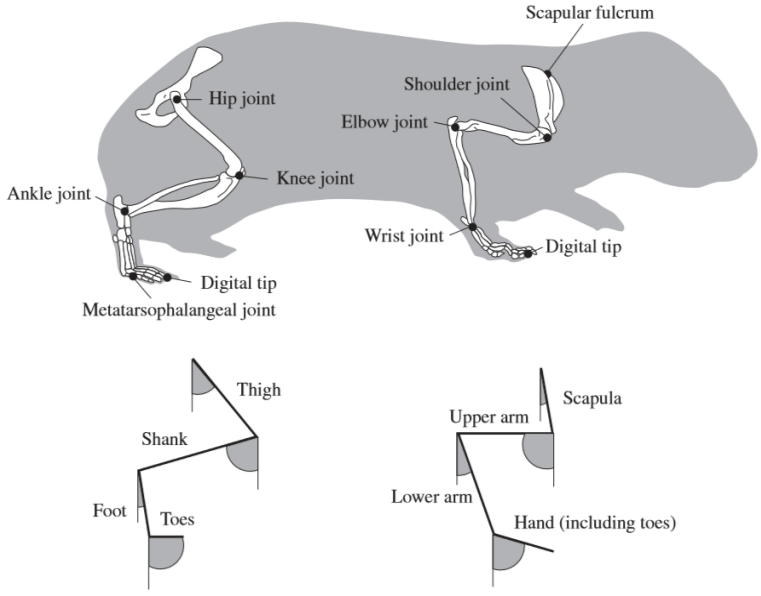


Figure 1-5. Representative leg configuration for therian mammals. Image originally shown in [37].

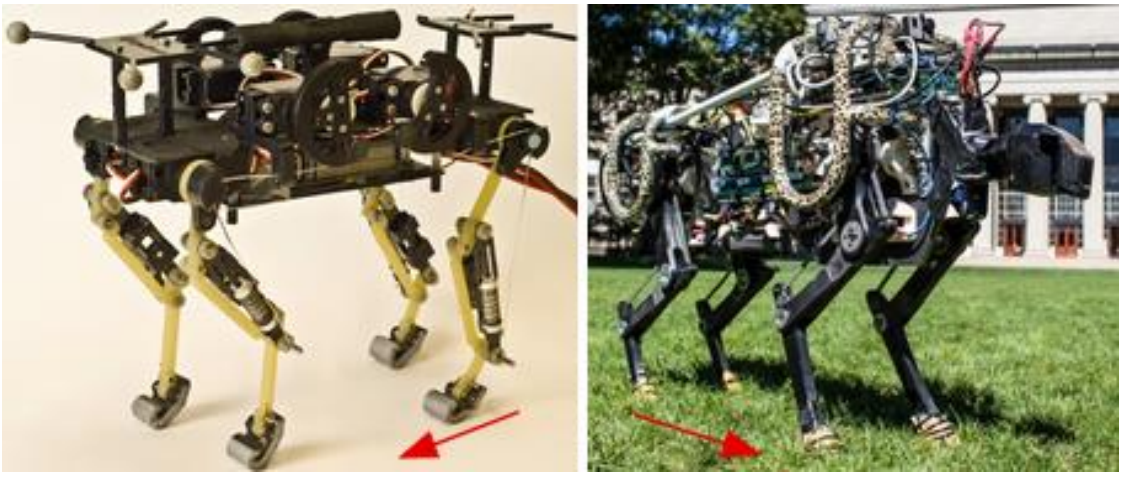


Figure 1-6. From left to right, the *Cheetah Cub* [38] and *MIT Cheetah II* [39] quadruped robots.

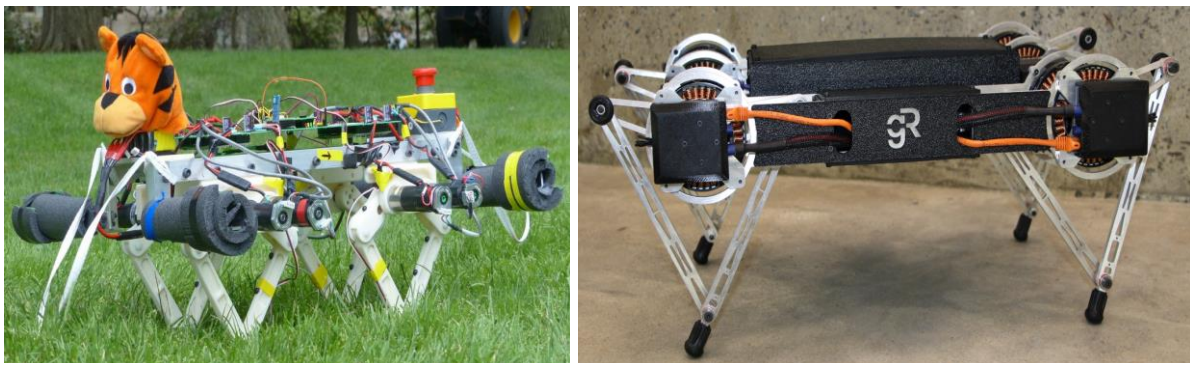


Figure 1-7. From left to right, *Super Mini Cheetah*, [40] and *MiniTaur*, [41] robots.

Early studies in legged locomotion [42] - [43], revealed the elastic behavior of legs. As every leg contacts the ground, it gets compressed and decompressed similar to simple spring mass oscillators. Today, the issue of elasticity/compliance in legs is dealt mainly with two different approaches; either by utilizing passive elements (springs) in the leg structure or by active compliance control of the actuators, [44]. The first approach poses the advantage that part of the necessary energy for locomotion, is stored as elastic potential energy in the system and is recirculated as passive elements compress and decompress, thus reducing the energy requirements of actuation. Furthermore, passive elements also act as shock absorbers during leg collision with the ground, protecting the actuation and transmission mechanisms. On the downside, passive elements introduce passive dynamics and low-frequency resonant modes into the system and therefore have to be tuned for a certain task. The second approach overcomes this obstacle, as the damping and the stiffness of the legs joints are determined by software. As a result, the compliance of the leg can be tuned for different tasks, restricted only by the actuators specifications. Nevertheless, the energy consumption in this approach is increased, as the necessary energy for the locomotive task is provided exclusively by the actuators. Moreover, it is reported that active compliance control in high speeds can only be achieved with actuators with low transmission ratios, as high transmission ratios increase the actuators passive impedance and backdriveability is reduced [45]. The *HyQ*, *MIT Cheetah II*, *Super Mini Cheetah* and *MiniTaur* quadruped robots have

actively compliant legs. Quadruped robots such as *Quadruped*, *Scout II*, *StarlETH* and *Cheetah Cub* on the other hand utilize passive elements in their leg structure.

The most common approaches in the application of passive compliance, are series elastic actuation (SEA) and parallel elastic actuation (PEA), see Figure 1-8. Actuation in series with a spring offers the following advantages: it increases actuation shock tolerance, provides torque-force tracking capabilities and reduces mechanical impedance, making possible to achieve force control on actuators with sifnificant transmission ratios, and absorbs power during stance phase, releasing it later in the stride cycle, [46]. Actuator inertia is suggested to not add to the joint inertia during collision (elastic energy is stored at the elastic element in series instead), and thus does not contribute to the collision losses. Nevertheless, SEA affects the response of the leg. High leg stiffness on low robot speeds, produces high roll and pitch movements that destabilize the quadruped, while low leg stiffness combined with high speeds causes legs to collapse, [19]. This creates the need for variable stiffness SEAs, which are rather complex and massive [47] - [48] for use on electrically actuated legs, where inertia should be reduced as possible. Solutions have been sought in nonlinear rubber springs, but while they reduce bulkiness, they introduce significant hysteresis, [49]. Promising also seems the use of revolute spiral leaf springs as series elastic feet [50], although they need position regulation and are rather heavy. Springs in parallel with actuation (PEA) are mainly used to cover actuation insufficiencies, as spring torques add to the actuator torques. However, the springs counteract actuators during spring excitation. Solutions have been sought by introducing mechanisms that unlatch the springs in parallel, in cases where they counteract actuation work, [51]. Folkertsma et al, examining the impact that parallel elastic elements would have on the *MIT Cheetah* leg, showed through simulation that they could lead to overall power reduction up to 50%, [52]. A comparison conducted between SEA and PEA applied to an electrically actuated monopod hopper [53], showed that the monopod hopping with SEA, needed less positive actuator work and had smaller electrical losses than one hopping with PEA.

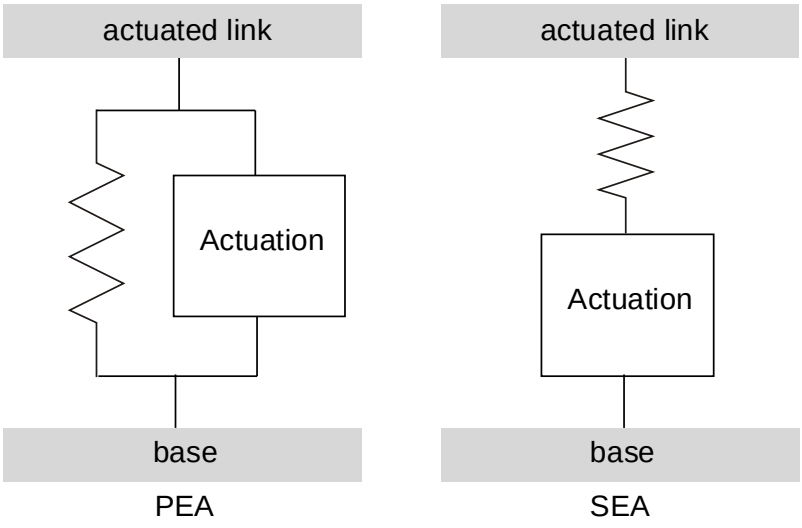


Figure 1-8. Abstract depiction of Parallel Elastic Actuation (PEA) and Series Elastic Actuation (SEA).

Apart from compliance, actuation is also a matter of great interest. The main approaches for actuating quadruped robots are *hydraulic* and *electric* actuation. Among the advantages of the electric motors in comparison with hydraulic actuators are their ease and accuracy of control, their low cost and the variety of available sizes and specifications, [15]. Their main drawback is presented to be the small torque they can produce in comparison with their size and weight, a trait that makes necessary the introduction of high reduction ratios through gearboxes. Reduction gears introduce undesired friction and increase the actuators' passive impedance that makes the motors non-backdrivable. State of the art solutions have been proposed by the *MIT Cheetah* and the ANYmal teams. The *MIT Cheetah* team studied and optimized the role of the gap radius in electromagnetic motors resulting in the construction of a low transmission high torque motor, [45]. The ANYmal team based on the actuation architecture of *StarlETH*, created the ANYdrive compact series elastic motors, [54]. On the other hand, as described in [54], hydraulic actuators have by nature high power and torque density and show great robustness against impulsive loads. In the case of autonomous robots, it is important that power supply and actuators can be compactly mounted on the robot. Electric motors can be supplied with power by batteries, while the hydraulic actuators need a pump to provide pressure to the hydraulic oil and an electric motor/battery setup or an internal combustion engine/fuel tank setup to supply the necessary power to the pump. As a result, for hydraulically actuated robots scalability is a great challenge. To this day, compact hydraulically actuated quadruped robots, as Spot or WildCat, are rather large, heavy and noisy [55] - [56]. The rest hydraulically actuated robots, as *HyQ* and its successor *HyQ*2Max [57] rely on off board supply.

Finally, *the materials* used in the construction of robotic legs are presented. Robotic legs should be lightweight so that fast swinging can be achieved. The inertial properties of the leg are also associated with impact losses during collision with the ground, as suggested in [10], [58]. On the other hand, the materials used in the construction of the leg should be strong enough to withstand the impulsive forces of ground contact, [59]. To deal with the trade-off of high strength and low mass/inertia various materials are being used. The legs of the *StarlETH* quadruped are made from high tensile aluminum, [14]. The *ANYmal* robot has legs made from carbon fiber and aluminum, [54]. Carbon fiber tubes were used in the legs of the *ATRIAS* robot, [23]. The legs of *MIT Cheetah* were constructed by a custom made composite material, consisting of a polyurethate foam core and a polyurethane casting resin shell, [60]. *HyQ* and *HyQ2Max* legs were constructed by aerospace type aluminium and steel alloys, [15], [57]. In smaller robots, such as *Super Mini Cheetah*, where strength requirements are not so strict, 3D printed ABS is used, [20].

1.3 Thesis Outline

In the following chapters, the approach adopted in this thesis regarding the appropriate leg design for high speed quadruped robots is presented. Along with the first introductory chapter, where the motivation of this work and previous work is discussed, the thesis is structured in six chapters.

In the second chapter, the descriptions used for the modelling of the physical system are presented. These refer to the gaits adopted for quadrupedal locomotion, the interaction of the quadruped robot with the traversed terrain, the profile of the forces exerted from the ground to the legs, the leg architecture and the quadruped robot. Moreover, the constraints incurred from the actuation system and the selected leg materials are described.

In the third chapter, the devised leg design methodology is described. The inputs, the outputs and the three parametric search stages of the method are explained. In the first stage the optimal footfalls and robot CoM trajectories for the motion of the quadruped are found. The description of the second stage follows, where permissible leg morphologies are found that satisfy actuation and leg material constraints. The third (outer) stage is explained, a stage where the parametric space of running task related parameters is spaned.

In the fourth chapter, the method is implemented for different gaits and leg configurations. The effect of these aspects in the maximum achievable velocity of locomotion is investigated. For the set of optimal parameters corresponding to the gait and leg configuration maximizing velocity of motion the results are validated, using independent to the method models and controller.

In the fifth chapter, the method is applied to yield the optimal gait and leg configuration for the existing actuation system of the *NTUA Laelaps* quadruped robot. Cases are studied regarding to which motor should act on which joint. A tipover stability criterion is presented and optimal solutions for three dimensional stable movement are presented. Furthermore, the role of supplementary motor reduction is investigated in the ability of the robot to run faster.

In the last chapter, the conclusions of the thesis are summarized and future work is suggested.

2 Physical system

2.1 Introduction

In this chapter, the necessary theory to understand the developed method is presented. Firstly, the reader becomes familiar with the running task and its properties. Next, since legged locomotion requires interaction with the ground, a proper leg - ground force interaction model is proposed. Existing leg architectures are compared and a proper conceptual design for the running task is selected. With the conceptual leg architecture available, the extensive quadruped robot model is introduced and its equations of motion (EoM) are derived. A simplified centroidal dynamics analog of the extensive model is also presented. The simplicity introduced from the centroidal dynamics description is necessary for optimizing the running task. For this description, necessary conditions for steady state running are explained and applied. Closing the chapter, the inverse kinematics relating the quadruped robot center of mass (CoM) and the toes of each leg are presented, and the running task is seen from the scope of leg joint space. Finally, the actuation constraints are presented and related with the joint space task of running.

2.2 The running task

The task of quadrupedal locomotion can be approached from two different scopes. The scope of each leg and that of the quadruped robot as a system. From the scope of the leg, the locomotion task consists of successive *stance* and *flight phases.* During the *stance phase*, the end effector of the leg remains in contact with the ground, as the leg pushes the body forward. The *flight phase* begins when the leg takes off from the ground. After take off, the leg shortens creating a *clearance* necessary to avoid collision with the ground and the leg is protracted forward. At the end of the *flight phase*, the leg once again comes into position to attack the ground, see Figure 2-1. Positions p_0 to p_4 correspond to *leg touch down, midstance, take off, midflight and second touch down*. The second touchdown denotes the completion of *the stride circle*. In steady state locomotion, the stride is a periodic phenomenon with period, T . The ratio of the time δt_s a leg is in contact with the ground in a stride (stance phase duration) over the stride period is called duty factor (DF) of the leg,

$$DF = \frac{\delta t_s}{T} \tag{2-1}$$

For a multi legged system, the temporal/local sequence in which the legs touch the ground determines the chosen gait for locomotion. To visualize the sequence of leg contact, gait graphs are drawn that depict the time a leg spends in contact with the ground, over the stride period, see Figure 2-2. Legs are named with a two letter abbreviation after their Hind-Front and Right-Left position. For instance, we refer to the hind left leg with the abbreviation HL. Mammals utilize different gaits for forward movement, depending on the species and the velocity of movement, [61]. In small to medium velocities, *bounding, pacing and trotting gaits* are common. For greater speeds, *rotary and transverse gallop* are utilized, [62] - [63]. A more detailed description of various quadrupedal gaits with their spatial/temporal properties is provided in [64].

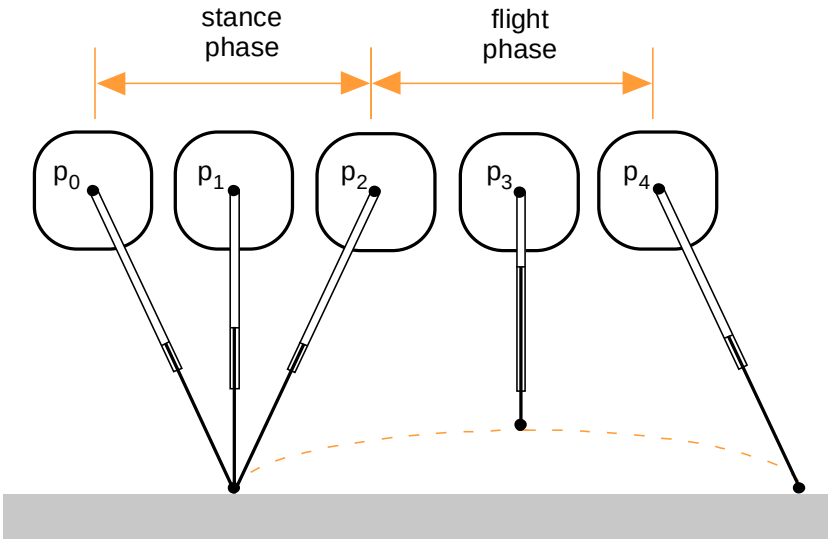


Figure 2-1. Stance and flight phase of a simple prismatic leg.

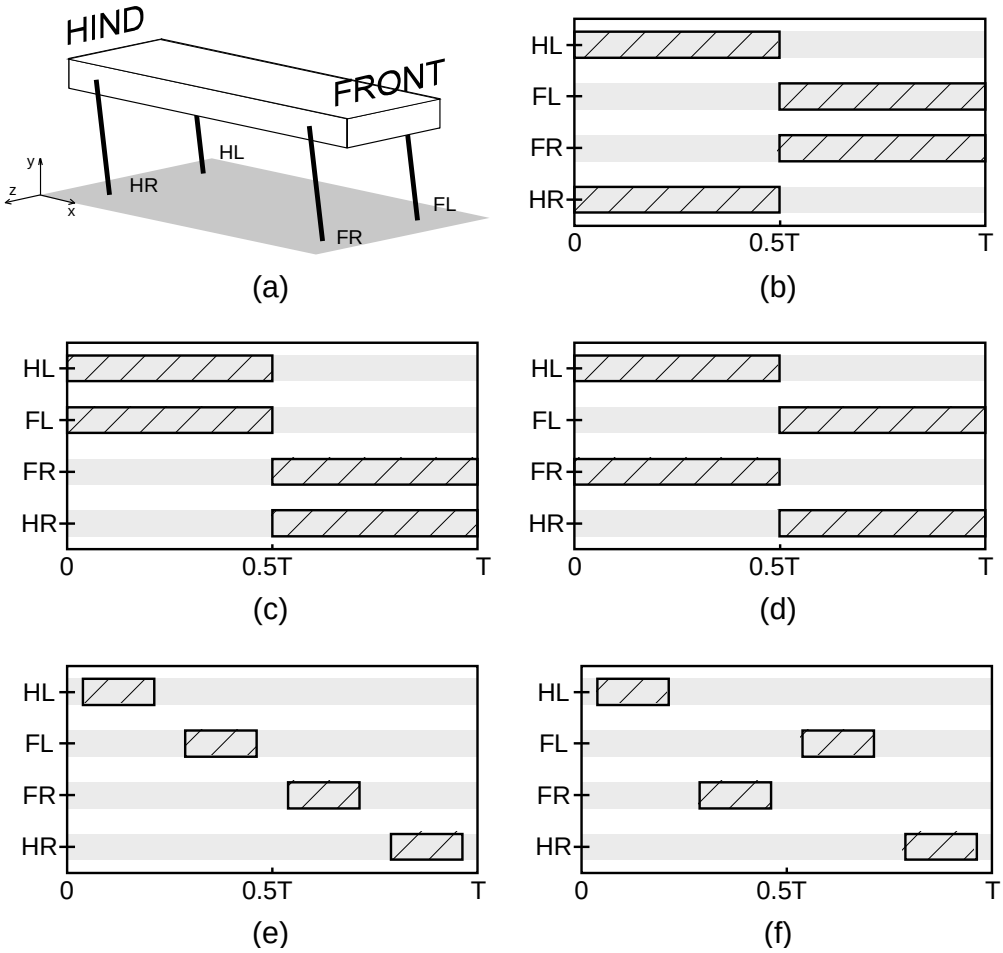


Figure 2-2. (a) Front, hind, left and right leg consensus. (b) to (f) Gait graphs of bounding, pacing, trotting, rotary and transverse galloping gaits.

Gaits can be further distinguished into *statically* and *dynamically* stable, [42]. If during a stable gait, there are at least three contact points between the legged system and the ground, then a support polygon is created from the points of contact. If the projection of the CoM of the legged system is situated into the support polygon then we refer to the gait as statically stable. If the projection of the CoM is not situated into the support polygon or no support polygon can be drawn (less than three contact points) then we refer to the gait as dynamically stable.

Legged locomotion is a three dimensional phenomenon, that takes place in the sagittal, transverse and frontal plane, see Figure 2-3. In this work though, we will be studying quadrupedal movement without turning (no yaw). Furthermore, proper gaits out of the aforementioned can be selected, for which the movement in the frontal plate (roll) is insignificant in comparison with the movement in the sagittal plane (pitch). For these gaits, the dynamics of the robot's forward locomotion are much simpler, as they are expressed exclusively in the sagittal plane. Such a requirement can be satisfied if during stance phase one left and one right leg are in contact with the ground simultaneously, as in the case of the bounding and the trotting gaits, see Figure 2-2 (b), (d). In the bounding gait, both front or both hind legs are in contact with the ground simultaneously. In the trotting gait, a pair of diagonal legs (HL-FR pair or HR-FL pair) are in contact with the ground at the same time.

As locomotive speed increases, a legged system transits from walking to running. Transition from walking to running seems to be coupled with a decrease in DF. It is suggested that walks have a duty factor greater than 0.5 per leg, while runs have a duty factor smaller than 0.5 per leg, [64]. As a result, there exist phases in running where no leg is in contact with the ground, the so called *full flight* or *ballistic phases*, see Figure 2-2 (e) and (f). Other researchers have supported that using DF as a criterion for transition from walking to running is unsafe, [65]. To their opinion, a safer criterion is the position of the CoM of the system at the moment of midstance. If at midstance the CoM is situated at its highest, then walking is performed. If, on the other hand, at midstance the CoM is situated at its lowest, then running is observed. In this work, running is defined using the midstance CoM position criterion.

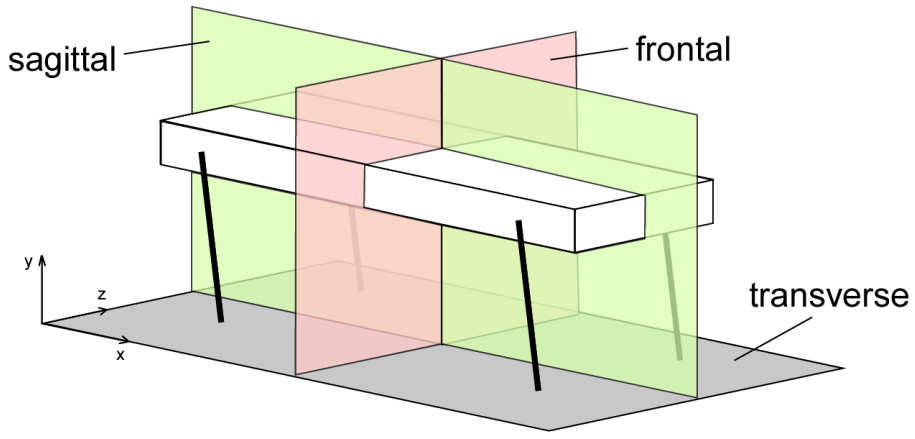


Figure 2-3. Sagittal, frontal and transverse planes.

2.3 Interaction with the ground

As explained in the previous section, the locomotive task is a sequence of leg stance and flight phases. In this section, we focus on the leg - ground interaction during the stance phase. Every time a leg touches the ground, forces are applied from the ground to the leg. The ground forces are manipulated by the legged system, in order to support its weight, stabilize its pitching movement and push itself forward.

The ground forces that are of importance for sagittal plane movement are the horizontal force $F_{x,i}$ and the vertical force $F_{y,i}$ exerted to the i-th leg in contact with the ground, for $i \in \{1,2,3,4\}$. Subscript $i=1$ corresponds to HL leg, $i=2$ to FL leg, $i=3$ to FR leg and, $i=4$ to HR leg. As contact events have very short duration, the nature of the ground forces is impulsive. That is supported from experimental data taken from running quadruped mammals, [59], [63] and running quadruped robots, [45], [60], [19]. The great applicability of the Spring Loaded Inverted Pendulum (SLIP) model, [42], is partly owed to its ability to accurately predict the form and magnitude of the ground forces applied to compliant legs. In all these works, the vertical force has a profile similar to a sinusoidal impulse, while the horizontal force has a decelerating - accelerating role, see Figure 2-4.

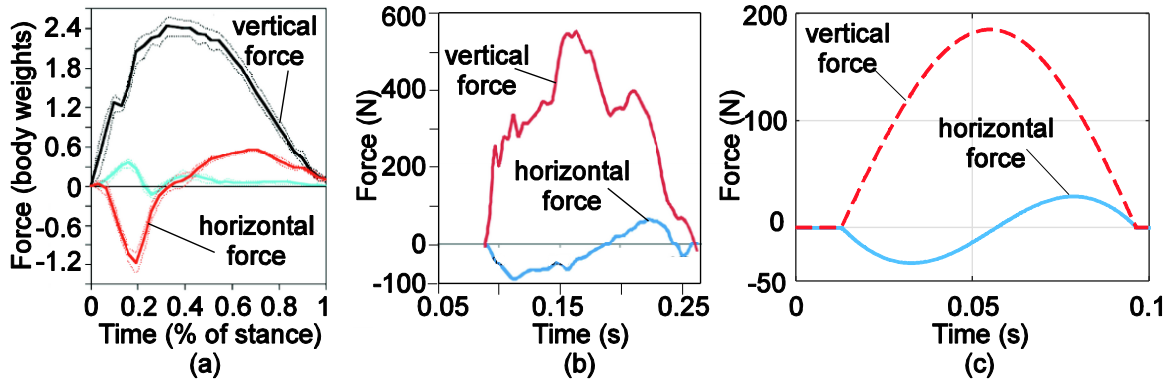


Figure 2-4. Evolution of ground forces with time for a single leg. (a) Measured forces from running dogs, [59] (b) measured forces applied on a leg of the *MIT Cheetah*, [60], (c) forces from a SLIP leg based quadruped model, [66].

For this reason, the ground forces were modeled in the form of sinusoidal functions. With the assumption that ground forces are equally distributed to all legs, the vertical force exerted to the i-th leg is described as a sinusoidal function with argument lying in [0, π] (half sine impulse),

$$F_{y,i}(t) = F_{y,\max} \sin(\omega_{y,i} t + \varphi_{y,i}) \tag{2-2}$$

and the horizontal force as a sinusoidal function with its argument lying in [π, 3π] and two times greater circular frequency compared to that of the vertical force,

$$F_{x,i}(t) = F_{x,\max} \sin(\omega_{x,i} t + \varphi_{x,i}), \quad \omega_{x,i} = 2\omega_{y,i} \tag{2-3}$$

If the i-th leg touches down at the time instant $t_{td,i}$ and takes off at $t_{to,i}$, with the duration of the stance being for all legs equal to δt_s, then the phase differences and circular frequencies of the sinusoidal ground forces can be found with respect to $t_{td,i}$, $t_{to,i}$, δt_s. Applying boundary conditions for the horizontal force $F_{x,i}$,

$$\begin{aligned} \omega_{x,i} t_{td,i} + \varphi_{x,i} &= \pi \\ \omega_{x,i} t_{to,i} + \varphi_{x,i} &= 3\pi \end{aligned} \tag{2-4}$$

Solving the system of eqs. (2-4),

$$\omega_{x,i} = \frac{2\pi}{t_{to,i} - t_{td,i}} = \frac{2\pi}{\delta t_s} \tag{2-5}$$

$$\varphi_{x,i} = \frac{\pi(t_{to,i} - 3t_{td,i})}{t_{to,i} - t_{td,i}} = \frac{\pi(t_{to,i} - 3t_{td,i})}{\delta t_s} \tag{2-6}$$

Boundary conditions are also applied for the vertical force $F_{y,i}$,

$$\begin{aligned} \omega_{y,i} t_{td,i} + \varphi_{y,i} &= 0 \\ \omega_{y,i} t_{to,i} + \varphi_{y,i} &= \pi \end{aligned} \tag{2-7}$$

Solving the system of eqs. (2-7),

$$\omega_{y,i} = \frac{\pi}{t_{to,i} - t_{td,i}} = \frac{\pi}{\delta t_s} \tag{2-8}$$

$$\varphi_{y,i} = \frac{-\pi t_{td,i}}{t_{to,i} - t_{td,i}} = \frac{-\pi t_{td,i}}{\delta t_s} \tag{2-9}$$

Employing the standard Coulomb friction model, for a leg to remain in contact with the ground without slipping, (2-10) must hold, where μ is the Coulomb friction coefficient. For (2-10) to hold throughout the stride period, an upper bound (subscript *ub*) (2-11) is set for the horizontal force amplitude,

$$F_{x,i}(t) \le \mu F_{y,i}(t) \tag{2-10}$$

$$F_{x,\max} \le F_{x,ub} \tag{2-11}$$

Introducing the parameter a_{fx}, inequality (2-11) is expressed in the form of an equation,

$$F_{x,\max} = a_{fx} F_{x,ub}, \; a_{fx} \in [0,1] \tag{2-12}$$

2.4 Leg architecture

As it became obvious in Section 1.2, various leg architectures are being used in literature for quadruped robots, each posing its own advantages and disadvantages. Before the leg architecture is decided, proper requirements should be set for the different parts of the legs: the joints and the links. Since the purpose of this work is to enhance the performance of the quadruped robot Laelaps in running, the parts of the leg should be strong enough to withstand the impulsive ground forces during stance phase, yet lightweight so that the leg inertia remains low and high accelerations can be achieved during leg swing in stance phase without much effort from actuation.

The leg joints are the elements through which the different links of the leg interact. From an abstract point of view, joints that are used for the cooperation of two links consist of three parts. One part where the first link is secured, a second one where the second link is secured and an intermediate part responsible for the non frictional cooperation of the other two parts, usually in the form of a bearing. For the three parts of the joint to cooperate without undesired oscillations and friction, they should have proper morphological and dimensional precision. If the desired cooperation is established, the materials of the cooperating parts should be robust (non-compliant) enough for the quality of the cooperation to remain intact after many cycles of use. A class of materials that satisfy these requirements of strength

and robustness, can be machined to the necessary precision, but are also lightweight enough for application in robotic legs is the aerospace aluminium alloys class. A representative example for this class of alloys is the 7075-T6 aluminium alloy, the properties of which are to be used for estimating the mass of the joints in this work, see Table 2-1.

The links of the leg determine the proportions and the total leg length. The links of the legs, apart from strong and lightweight, should also be easily interchangeable, so that the results proposed in theory and simulation can be tested in hardware without much reconstructing effort. Standardized products that meet the above requirements are carbon fiber tubes. Carbon fiber tubes are of low cost and available in many different standard sizes and with alternative specifications. Successive tube sizes can be used one inside the other to achieve the desired strength. The remaining hollow space on the inside can accommodate returning cables from sensors and encoders, leading to compact solutions. For all the above reasons, the geometry and properties of carbon tubes are to be used for estimating the mass and introduce strength constraints in the models to follow. The strength and density values of the carbon fiber tubes of Table 2-1 were experimentally specified, see Appendix A.

As the joints are the most heavy parts of the leg mechanism, the least number of joints should be selected that accommodate the needs of the leg. Among the alternative leg architectures (Figure 1-1), articulated legs pose an advantage in comparison with prismatic legs, when it comes to ground clearance. From the articulated legs, the most lightweight solution is the two-segment leg, as it consists of the least number of joints. Electric direct current (DC) motors are used for actuation, as it's been observed from literature that electrically actuated robots are lighter, more easily scalable and less noisy than their hydraulic analogues. Both electric motors necessary for the actuation of the two-segment leg are mounted on the robot's body (base), to make the legs lighter. A prismatic compliant joint is added at the most distal leg segment, to protect the mechanical parts (motors, transmissions, etc.) from the impulsive forces of ground interaction. In this way, the safety of the mechanical parts of both actuated joints is ensured, to the expense of the mass of the compliant element, see Figure 2-5.

Table 2-1. Materials used for modeling the structural parts of the legs and their properties.

Leg structural part	Material	density [kg/m^3]	yield strength [MPa]
joints	7075-T6	2810	430
links	carbon fiber tubes	1466	200

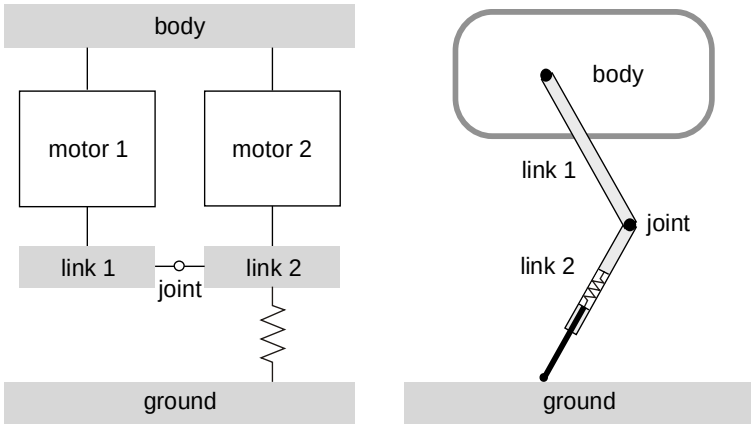


Figure 2-5. Abstract depiction of the two-segment leg selected architecture interacting with the ground.

2.5 Quadruped robot dynamics

Based on the leg architecture proposed in the previous section, a quadruped robot model for movement in the sagittal plane is introduced, see Figure 2-6. The position of the body CoM is situated at (x, y) with respect to the origin O. The pitch angle of the body is described by the variable θ. Each leg has three joints; a rotary hip joint which position is described by variable $\theta_{1,i}$, a rotary knee joint with respective variable $\theta_{2,i}$ and a passive compliant prismatic joint related with joint variable $l_{2,i}$. Note that lines (δ), (ε) are parallel and at an angle θ, so that joint angles $\theta_{1,i}$, $\theta_{2,i}$ are angles relative to the body. The joints described by $\theta_{1,i}$ and $\theta_{2,i}$ are actuated with the torques $\tau_{1,i}$, $\tau_{2,i}$. The interaction of the i-th leg with the ground is modeled with the use of the ground forces. When the i-th leg is in contact with the ground, the sinusoidal ground forces (2-2), (2-3) are exerted to the end effector of the leg (toe). In such a manner, the state vector necessary to describe the system is written as,

$$\mathbf{q} = \left[x \; y \; \theta \; \theta_{1,1} \; \theta_{2,1} \; l_{2,1} \; \theta_{1,4} \; \theta_{2,4} \; l_{2,4} \; \theta_{1,2} \; \theta_{2,2} \; l_{2,2} \; \theta_{1,3} \; \theta_{2,3} \; l_{2,3} \right]^T \tag{2-13}$$

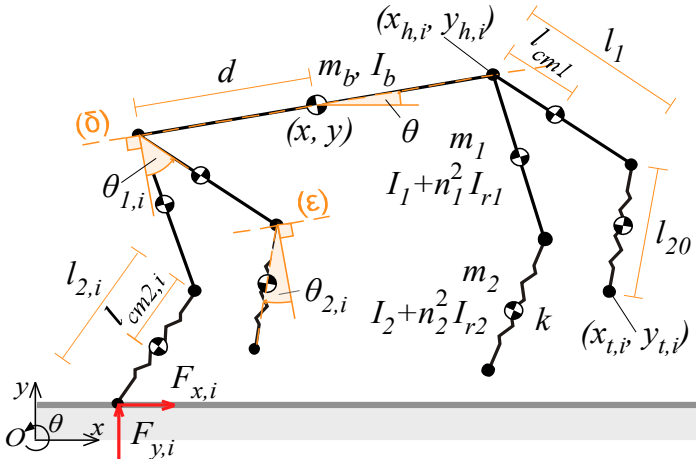


Figure 2-6. Quadrupedal robot model.

The robot body has mass m_b and inertia I_b and the distance of any hip joint from the CoM of the robot body is d. The body is considered rectangular, with length a and height b. In this way the moment of inertia of the body is calculated as,

$$I_b = \frac{1}{12} m_b (a^2 + b^2) \tag{2-14}$$

The model of the i-th leg consists of two segments. The proximal to the body segment has length l_1, inertia I_1 and mass m_1 at distance l_{cm1} from the hip joint. The distal to the body segment is compliant with stiffness k, free length l_{20}, inertia $I_{2,i}$ and mass m_2 at distance $l_{cm2,i}$ from the knee joint. Note that the leg specific index i is used for properties that vary among different legs for the same time instant. All leg segments are considered tubular, with outer diameter d_{out}, inner diameter d_{in} and density ρ, see Figure 2-7. The mass of the proximal link m_1 consists of half the hip joint mass $m_{j1}/2$ and half the knee joint mass $m_{j2}/2$ and the mass of the tubular link m_{l1},

$$\begin{aligned} m_1 &= \frac{m_{j1}}{2} + \frac{m_{j2}}{2} + m_{l1} \\ m_{l1} &= \rho\pi \frac{d_{out}^2 - d_{in}^2}{4} l_1 \end{aligned} \tag{2-15}$$

The mass of the distal link m_2 consists of half the knee joint mass $m_{j2}/2$, the spring mass m_s, the toe mass m_t and the mass of the tubular link m_{l2},

$$\begin{aligned} m_2 &= \frac{m_{j2}}{2} + m_s + m_{l2} + m_t \\ m_{l2} &= \rho\pi \frac{d_{out}^2 - d_{in}^2}{4} l_{20} \end{aligned} \tag{2-16}$$

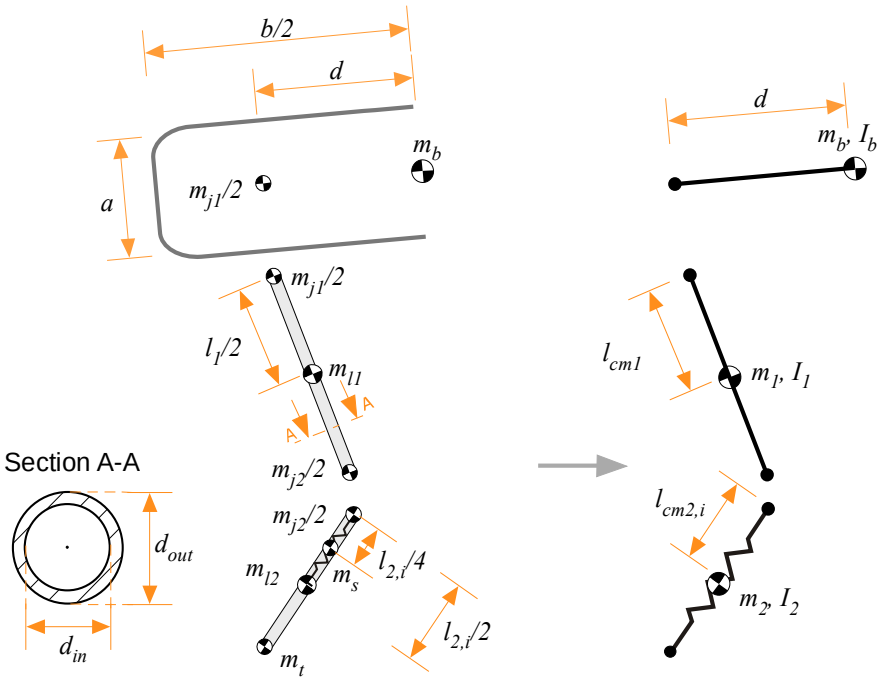


Figure 2-7. Inertial properties of lumped model.

From the distribution of the masses along every link, the distance of the CoM of the first link from the hip joint l_{cm1} and that of the CoM of the second link from the knee joint $l_{cm2,i}$ are calculated as,

$$l_{cm1} = \frac{m_{l1} \frac{l_1}{2} + \frac{m_{j2}}{2} l_1}{m_1} \tag{2-17}$$

$$l_{cm2,i} = \frac{m_s \frac{l_{2,i}}{4} + m_{l2} \frac{l_{2,i}}{2} + m_t l_{2,i}}{m_2} \tag{2-18}$$

The moments of inertia with respect to the CoM of the tubular leg segments are calculated as,

$$I_1 = \frac{1}{12} m_1 l_1^2 \tag{2-19}$$

$$I_{2,i} = \frac{1}{12} m_2 l_{2,i}^2 \tag{2-20}$$

The actuators are modeled as DC motors with transmission ratio n_j and rotor inertia I_{rj} (j=1, 2) and they are considered to be mounted on the body to make the legs lighter.

Having explicitly determined the state variables $\mathbf{q}$ and the dynamical properties of the system, we proceed to formulate the system's EoM, following the Euler - Lagrange formulation. From forward kinematics, the positions of the lumped masses are related to the state variables. The absolute coordinates of the hind hip joints $(i \in \{1,4\})$ are,

$$x_{h,i} = x - d\cos(\theta) \tag{2-21}$$

$$y_{h,i} = y - d\sin(\theta) \tag{2-22}$$

The absolute coordinates of the front hip joints $(i \in \{2,3\})$ are,

$$x_{h,i} = x + d\cos(\theta) \tag{2-23}$$

$$y_{h,i} = y + d\sin(\theta) \tag{2-24}$$

The CoM of the proximal leg segment of any leg is situated at $(x_{1,i}, y_{1,i})$, where

$$x_{1,i} = x_{h,i} + l_{cm1}\sin(\theta + \theta_{1,i}) \tag{2-25}$$

$$y_{1,i} = y_{h,i} - l_{cm1}\cos(\theta + \theta_{1,i}) \tag{2-26}$$

The CoM of the distal leg segment of any leg is situated at $(x_{2,i}, y_{2,i})$, where

$$x_{2,i} = x_{h,i} + l_1\sin(\theta + \theta_{1,i}) + l_{cm2,i}\sin(\theta + \theta_{2,i}) \tag{2-27}$$

$$y_{2,i} = y_{h,i} - l_1\cos(\theta + \theta_{1,i}) - l_{cm2,i}\cos(\theta + \theta_{2,i}) \tag{2-28}$$

The position of the toe for any leg is described by,

$$x_{t,i} = x_{h,i} + l_1\sin(\theta + \theta_{1,i}) + l_{2,i}\sin(\theta + \theta_{2,i}) \tag{2-29}$$

$$y_{t,i} = y_{h,i} - l_1\cos(\theta + \theta_{1,i}) - l_{2,i}\cos(\theta + \theta_{2,i}) \tag{2-30}$$

Then the kinetic energy of the system is written in the form,

$$\begin{aligned} T_L = \frac{1}{2} I_b \dot{\theta}^2 + \frac{1}{2} m_b (\dot{x}^2 + \dot{y}^2) + \sum_{i=1}^{4} \Bigg[\frac{1}{2} (I_1 + n_1^2 \mathrm{I}_{r1}) \dot{\theta}_{1,i}^2 + \frac{1}{2} m_1 (\dot{x}_{1,i}{}^2 + \dot{y}_{1,i}{}^2) + \\ + \frac{1}{2} (I_{2,i} + n_2^2 \mathrm{I}_{r2}) \dot{\theta}_{2,i}^2 + \frac{1}{2} m_2 (\dot{x}_{2,i}{}^2 + \dot{y}_{2,i}{}^2) \Bigg] \end{aligned} \tag{2-31}$$

The potential energy of the system is calculated as,

$$U_L = mgy + \sum_{i=1}^{4} \left[m_{1,i} g y_{1,i} + m_{2,i} g y_{2,i} + \frac{1}{2} k \delta l_{2,i}^2 \right] \tag{2-32}$$

where g is the gravitational acceleration and $\delta l_{2,i}$ is the deformation of the passive element defined as,

$$\delta l_{2,i} = l_{2,i} - l_{20} \tag{2-33}$$

The Lagrangian is calculated from the difference of the kinetic and the potential energy,

$$L = T_L - U_L \tag{2-34}$$

The power from actuation torques and external forces are expressed in the form,

$$P = \sum_{i=1}^{4}\left[\tau_{1,i}\dot{\theta}_{1,i} + \tau_{2,i}\dot{\theta}_{2,i} + c_i(F_{x,i}\dot{x}_{t,i} + F_{y,i}\dot{y}_{t,i})\right] \tag{2-35}$$

where c_i is a variable equal to 1 when the i-th leg is in contact with the ground, and 0 otherwise,

$$c_i = \begin{cases} 1, & t \in \left[t_{td,i}, t_{to,i}\right] \\ 0, & t \in [0,T] - \left[t_{td,i}, t_{to,i}\right] \end{cases} \tag{2-36}$$

The EoM are then calculated from Equation (2-37),

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{\mathbf{q}}}\right) - \frac{\partial L}{\partial \mathbf{q}} = \frac{\partial P}{\partial \dot{\mathbf{q}}} \tag{2-37}$$

Finally, following the Euler - Lagrange formulation the EoM are written as,

$$\mathbf{M}\ddot{\mathbf{q}} + \mathbf{C}\dot{\mathbf{q}} + \mathbf{K}\mathbf{q} + \mathbf{J}^T\mathbf{F} = [\mathbf{0} \;\; \boldsymbol{\tau}]^T \tag{2-38}$$

where $\mathbf{M}$ is the mass matrix of the system, $\mathbf{C}$ is the centrifugal/Coriolis terms matrix, $\mathbf{K}$ is the stiffness matrix, $\mathbf{J}$ is the Jacobian matrix of the legs, $\mathbf{F}$ is the vector of the external forces and $\boldsymbol{\tau}$ is the actuation torques vector consisting of the hip and knee actuation torques $\tau_{1,i}$, $\tau_{2,i}$ respectively.

2.6 Simplified centroidal dynamics

With the centroidal dynamics approach, the dynamics of the system are projected to its CoM, see Figure 2-8. This approach introduces great simplicity, as for the description of the system only the position of the CoM, the orientation of the system and the position of the footfalls during locomotion are needed. As the centroidal dynamics of the system are much simpler, the role of the ground forces and position of footfalls in the periodicity of the movement gets explicit, and by tuning these parameters it is easy to regulate the trajectory of the CoM of the robot.

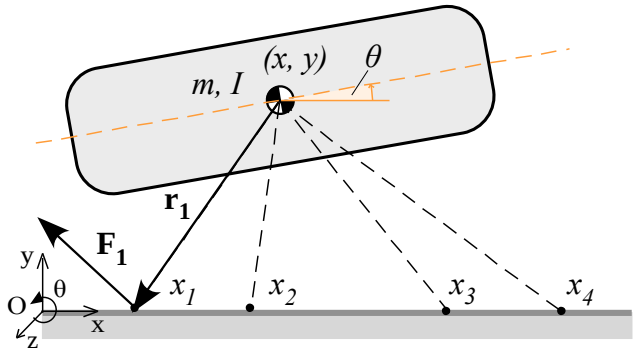


Figure 2-8. Centroidal dynamics model and its properties.

In the literature, this approach has been used for CoM trajectory optimization and postural balance control of humanoid [67], [68] and quadruped robots [69]. In the case of humanoid robots, much of the robot's mass is concentrated at its limbs (arms, legs). Therefore, the posture of the limbs affects significantly the position of the CoM of the system. On the other hand, in the case of performance quadruped robots, the legs are lightweight in comparison with the main body. As a result, change of posture of the legs during locomotion does not significantly affect the position of the CoM of the system, and thus the formulation of the centroidal dynamics is simplified, [69]. With such an approach, the CoM

of the quadruped robot coincides with the CoM of the quadruped robot's main body at every time instant, the mass m of the centroidal model is equal to the total mass of the quadruped robot and the inertia of the centroidal model I is approximately constant and equal to that of the quadruped robot's main body,

$$m = m_b + \sum_i (m_{1,i} + m_{2,i}),\ I \simeq I_b \tag{2-39}$$

The system is described by the position of the CoM (x, y) and the pitch angle θ. With such generalized coordinates, the EoM of the centroidal model are expressed in the form,

$$\ddot{x} = F_x / m \tag{2-40}$$

$$\ddot{y} = F_y / m - g \tag{2-41}$$

$$\ddot{\theta} = \tau_\theta / \mathrm{I} \tag{2-42}$$

where F_x, F_y are the total forces and τ_θ is the total torque acting on the CoM due to interaction with the ground defined as,

$$F_x = \sum_i c_i F_{x,i} \tag{2-43}$$

$$F_y = \sum_i c_i F_{y,i} \tag{2-44}$$

$$\tau_\theta = \sum_i c_i \mathbf{r}_i \times \mathbf{F}_i \cdot \hat{\mathbf{z}} = \sum_i c_i \left[(x_i - x) F_{y,i} - (y_i - y) F_{x,i} \right] \tag{2-45}$$

where the variable c_i denotes when the i-th leg leg comes in contact with the ground, see (2-36), $\mathbf{r}_i$ is the vector connecting the CoM position (x, y) with the position of the toe during stance phase (x_i, y_i), $\mathbf{F}_i$ is the vector of the horizontal and vertical forces $F_{x,i}$, $F_{y,i}$ exerted on the i-th toe and $y_i = 0$ with the assumption that locomotion takes place on even terrain, see Figure 2-8.

To move periodically in the y and θ directions with a steady net horizontal velocity (x direction), the robot should have no net acceleration in any direction in one stride circle. This can be expressed in terms of total forces and torques from ground interaction, in the form of Equations (2-46)-(2-48):

$$\int_0^T F_x dt = 0 \tag{2-46}$$

$$\int_0^T F_y dt = mgT \tag{2-47}$$

$$\int_0^T \tau_\theta dt = 0 \tag{2-48}$$

Combining Equations (2-3), (2-36) and (2-43), it becomes obvious that Equation (2-46) is satisfied by the definition of the horizontal force,

$$\int_0^T F_x dt = \sum_i \int_{t_{td,i}}^{t_{to,i}} F_{x,\max} \sin(\omega_{x,i} t + \varphi_{x,i}) dt == \frac{F_{x,\max}}{\omega_{x,i}} \sum_i \int_\pi^{3\pi} \sin(\varphi)\, d\varphi = \frac{F_{x,\max}}{\omega_{x,i}} \left[-\cos(\varphi) \right]_\pi^{3\pi} = 0 \tag{2-49}$$

For the vertical direction, from Equations (2-2), (2-8), (2-36), (2-44), and taking into account (2-50),

$$\begin{aligned} &\sum_i \int_0^T c_i \sin(\omega_{y,i} t + \varphi_{y,i})\, dt = \sum_i \int_{t_{td,i}}^{t_{to,i}} \sin(\omega_{y,i} t + \varphi_{y,i})\, dt = \\ &= \sum_i \int_{t_{td,i}}^{t_{to,i}} \left[\frac{-\cos(\omega_{y,i} t + \varphi_{y,i})}{\omega_{y,i}} \right]' dt = \sum_i \int_0^\pi \left[\frac{-\cos(\varphi)}{\omega_{y,i}} \right]' d\varphi = \\ &= \frac{1}{\omega_{y,i}} \sum_i \left[-\cos(\varphi) \right]_0^\pi = \frac{8}{\omega_{y,i}} = \frac{8\delta t_s}{\pi} \end{aligned} \tag{2-50}$$

Equation (2-47) is written in the form,

$$F_{y,\max}\frac{8\delta t_s}{\pi}=mgT \tag{2-51}$$

From which,

$$F_{y,\max}=\frac{\pi mgT}{8\delta t_s} \tag{2-52}$$

Taking also into account Equation (2-1), the amplitude of the half sine vertical force is expressed as,

$$F_{y,\max}=\frac{\pi mg}{8DF} \tag{2-53}$$

Equation (2-47) is only a necessary condition for periodical movement. In order to ensure that the robot moves periodically in a stride circle, also a supplementary condition is introduced, which requires that the CoM of the robot body at the start of the stride is at apex,

$$\dot{y}(0)=0 \tag{2-54}$$

As a result, with zero initial vertical velocity (2-54) and no net acceleration in the y direction (2-47), periodicity in the y direction in one stride is ensured,

$$\dot{y}(0)=\dot{y}(T)=0 \tag{2-55}$$

$$y(0)=y(T) \tag{2-56}$$

Finally, to ensure periodical movement in the θ direction in one stride, we demand conditions (2-57), (2-58) to hold.

$$\theta(0)=\theta(\mathrm{T}) \tag{2-57}$$

$$\dot{\theta}(0)=\dot{\theta}(\mathrm{T}) \tag{2-58}$$

From Equation (2-45), it is clear that the coordinates of the footfalls are crucial for the satisfaction of Equations (2-57), (2-58). Note that if this pair of equations is satisfied, no net acceleration in the θ direction exists and thus Equation (2-48) also holds.

2.7 Inverse kinematics

Although the centroidal dynamics introduce great simplicity, they provide no information on the leg joint variables during locomotion, see (2-40)-(2-42). Nevertheless, with the assumption that the CoM of the quadruped robot, coincides with the CoM of the robot's main body, by knowing the full geometry of the robot between it's CoM and the toes and also the position of the toes, the leg joint variables can be related with the variables of the CoM, through inverse kinematics.

The positions of the hind and front hip joints are related to the position and orientation of the robot's body through Equations (2-21)-(2-24). For each leg the position of its toe is related to the position of its hip through Equations (2-29), (2-30). Let $x_{hr,i}$, $y_{hr,i}$ be the relative hip to toe coordinates defined as,

$$x_{hr,i}=x_{h,i}-x_{t,i} \tag{2-59}$$

$$y_{hr,i}=y_{h,i}-y_{t,i} \tag{2-60}$$

then by replacing Equations (2-59), (2-60) in Equations (2-29), (2-30),

$$x_{hr,i}+l_{2,i}\sin(\theta+\theta_{2,\mathrm{i}})=-l_1\sin(\theta+\theta_{1,i}) \tag{2-61}$$

$$y_{hr,i}-l_{2,i}\cos(\theta+\theta_{2,i})=l_1\cos(\theta+\theta_{1,i}) \tag{2-62}$$

Squaring,

$$\left[x_{hr,i}+l_{2,i}\sin(\theta+\theta_{2,i})\right]^2=\left[-l_1\sin(\theta+\theta_{1,i})\right]^2 \tag{2-63}$$

$$\left[y_{hr,i} - l_{2,i}\cos(\theta + \theta_{2,i}) \right]^2 = \left[l_1 \cos(\theta + \theta_{1,i}) \right]^2 \tag{2-64}$$

Adding (2-63), (2-64) we result in,

$$x_{hr,i}^2 + y_{hr,i}^2 + l_{2,i}^2 + 2x_{hr,i}l_{2,i}\sin(\theta + \theta_{2,i}) - 2y_{hr,i}l_{2,i}\cos(\theta + \theta_{2,i}) = l_1^2 \tag{2-65}$$

$$x_{hr,i}\sin(\theta + \theta_{2,i}) - y_{hr,i}\cos(\theta + \theta_{2,i}) = \frac{l_1^2 - x_{hr,i}^2 - y_{hr,i}^2 - l_{2,i}^2}{2l_{2,i}} \tag{2-66}$$

$$x_{hr,i}\sin(\theta + \theta_{2,i}) - y_{hr,i}\cos(\theta + \theta_{2,i}) = a_{hr,i} \tag{2-67}$$

where $a_{hr,i}$ is defined as,

$$a_{hr,i} = \frac{l_1^2 - x_{hr,i}^2 - y_{hr,i}^2 - l_{2,i}^2}{2l_{2,i}} \tag{2-68}$$

Let $\varphi_{2,i}$ be,

$$\varphi_{2,i} = \theta + \theta_{2,i} \tag{2-69}$$

Then from trigonometry,

$$\sin\varphi_{2,i} = \frac{2\tan\frac{\varphi_{2,i}}{2}}{1 + \tan^2\frac{\varphi_{2,i}}{2}} \tag{2-70}$$

$$\cos\varphi_{2,i} = \frac{1 - \tan^2\frac{\varphi_{2,i}}{2}}{1 + \tan^2\frac{\varphi_{2,i}}{2}} \tag{2-71}$$

Replacing (2-69)-(2-71) in (2-67),

$$(y_{hr,i} - a_{hr,i})\tan^2\frac{\varphi_{2,i}}{2} + 2x_{hr,i}\tan\frac{\varphi_{2,i}}{2} - y_{hr,i} - a_{hr,i} = 0 \tag{2-72}$$

Solving this second order polynomial w.r.t. $\tan(\varphi_{2,i}/2)$,

$$\tan\frac{\varphi_{2,i}}{2} = \frac{-2x_{hr,i} \pm \sqrt{4x_{hr,i}^2 + 4y_{hr,i}^2 - 4a_{hr,i}^2}}{2(y_{hr,i} - a_{hr,i})} = \frac{-x_{hr,i} \pm \sqrt{x_{hr,i}^2 + y_{hr,i}^2 - a_{hr,i}^2}}{(y_{hr,i} - a_{hr,i})} \tag{2-73}$$

where

$$x_{hr,i}^2 + y_{hr,i}^2 - a_{hr,i}^2 \geq 0 \tag{2-74}$$

Replacing (2-69) in (2-73) and solving w.r.t. variable $\theta_{2,i}$,

$$\theta_{2,i} = 2\,\mathrm{atan2}\left[-x_{hr,i} \pm \sqrt{x_{hr,i}^2 + y_{hr,i}^2 - a_{hr,i}^2},\ y_{hr,i} - a_{hr,i} \right] - \theta \tag{2-75}$$

Note that the solution with the positive sign corresponds to the knee backward configuration and that with negative sign to a knee forward configuration. Solving Equations (2-61), (2-62) w.r.t variable $\theta_{1,i}$,

$$\theta_{1,i} = \mathrm{atan2}\left[-x_{hr,i} - l_{2,i}\sin(\theta + \theta_{2,i}),\ y_{hr,i} - l_{2,i}\cos(\theta + \theta_{2,i}) \right] - \theta \tag{2-76}$$

Inequality (2-74) ensures that the toe remains in the workspace of the leg with segment lengths l_1, $l_{2,i}$. Substituting (2-68) in (2-74) and factoring we obtain,

$$\left[x_{hr,i}^2 + y_{hr,i}^2 - (l_1 - l_{2,i})^2 \right]\left[-(x_{hr,i}^2 + y_{hr,i}^2) + (l_1 + l_{2,i})^2 \right] \geq 0 \tag{2-77}$$

Solving inequality (2-77), we obtain the pair of inequalities (2-78), which describe the workspace of the leg as a bounded annular area, with inner radius $|l_1 - l_{2,i}|$ and outer radius $l_1 + l_{2,i}$.

$$(l_1 - l_{2,i})^2 \le x_{hr,i}^2 + y_{hr,i}^2 \le (l_1 + l_{2,i})^2$$
$$|l_1 - l_{2,i}| \le \sqrt{x_{hr,i}^2 + y_{hr,i}^2} \le l_1 + l_{2,i} \quad (2\text{-}78)$$

From the inverse kinematics Equations (2-75), (2-76), the joint angles $\theta_{1,i}, \theta_{2,i}$ are related through the instantaneous leg geometry l_1, $l_{2,i}$ to hip to toe relative coordinates. Thus, even though the centroidal dynamics are described only by the position of the CoM and the orientation of the robot's body, through inverse kinematics, also information regarding the leg joint variables during locomotion becomes available, if the position of the toes at every time instant is given.

2.8 Hardware constraints

To further approach reality the physical system described in the previous sections is subject to realistic hardware constraints. The hardware constraints presented here are divided in *leg strength constraints* and in *actuation constraints*. Leg strength constraints refer to the constraints in strength introduced by the material and the geometry of the tubular leg segments. Given a material for the leg segments, the leg strength constraints determine the permissible leg segment length and cross-sectional area. Actuation constraints, on the other hand, are related to the thermal constraints of DC motors and strength constraints of their gearbox. Actuation constraints are expressed in terms of maximum values for the actuation torques and angular velocities, which must not be exceeded to avoid damage in the actuation system.

The leg segments are stressed during stance phase, when the impulsive ground forces are exerted to the leg. Ground forces have a compressive component acting on the longitudinal direction of each leg segment F_l and a shear/bending component acting on the direction vertical to the leg segment F_v. Those two components are found by projecting the ground forces on the directions collinear to each leg segment $\hat{\boldsymbol{l}}$, and vertical to each leg segment $\hat{\boldsymbol{v}}$, see Figure 2-9. With the assumption that the weight of the leg segments are insignificant in comparison to the ground forces, the forces applied on the proximal leg segment are approximately the same with the forces applied to the toe and equal to $F_{x,i}$, $F_{y,i}$.

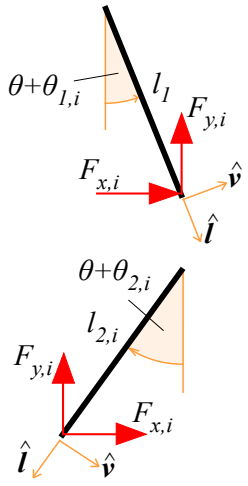


Figure 2-9. Positive directions $\hat{\boldsymbol{l}}$, $\hat{\boldsymbol{v}}$ and the forces applied on the leg segments.

Projecting the ground forces on the $\hat{\boldsymbol{l}}$ and $\hat{\boldsymbol{v}}$ directions for the proximal to the body leg segment,

$$F_{l,1,i} = F_{x,i}\sin(\theta + \theta_{1,i}) - F_{y,i}\cos(\theta + \theta_{1,i}) \tag{2-79}$$

$$F_{v,1,i} = F_{x,i}\cos(\theta + \theta_{1,i}) + F_{y,i}\sin(\theta + \theta_{1,i}) \tag{2-80}$$

and for the distal leg segment,

$$F_{l,2,i} = F_{x,i}\sin(\theta + \theta_{2,i}) - F_{y,i}\cos(\theta + \theta_{2,i}) \tag{2-81}$$

$$F_{v,2,i} = F_{x,i}\cos(\theta + \theta_{2,i}) + F_{y,i}\sin(\theta + \theta_{2,i}) \tag{2-82}$$

Let A, I_a be the cross-sectional area and the second moment of area of the tubular leg segments defined as,

$$A = \frac{\pi}{4}(d_{out}^2 - d_{in}^2) \tag{2-83}$$

$$I_a = \frac{\pi}{64}(d_{out}^4 - d_{in}^4) \tag{2-84}$$

Then the compressive stress σ_c, the shear stress τ_s and the bending stress σ_b on the j-th leg segment $j \in \{1,2\}$ of the i-th leg $i \in \{1,2,3,4\}$ are found by Equations (2-85)-(2-87).

$$\sigma_{c,j,i} = \frac{F_{l,j,i}}{A} \tag{2-85}$$

$$\tau_{s,j,i} = \frac{F_{v,j,i}}{A} \tag{2-86}$$

$$\sigma_{b,j,i} = \frac{M_{b,j,i}}{I_a}\frac{d_{out}}{2} \tag{2-87}$$

where $M_{b,j,i}$ is the bending moment of the vertical to the j-th leg segment force component $F_{v,j,i}$,

$$M_{b,j,i} = F_{v,j,i} l_{j,i} \tag{2-88}$$

The equivalent normal stress on the cross-sectional area of each leg segment is:

$$\sigma_{n,j,i} = \left|\sigma_{c,j,i}\right| + \left|\sigma_{b,j,i}\right| \tag{2-89}$$

To respect the strength constraints of the tubular leg segments, the equivalent normal and shear stresses should be less or equal to the permissible values of the material,

$$\begin{aligned} \sigma_{n,j,i} &\le \frac{S_{c,u}}{s_f} \\ \tau_{s,j,i} &\le \frac{\tau_{s,u}}{s_f} \end{aligned} \tag{2-90}$$

where $S_{c,u}$, $\tau_{s,u}$ are the ultimate compressive and shear strengths of the material of the leg segments and s_f is a user defined safety factor.

The constraints of the actuating DC motors are divided in thermal and strength constraints. To better understand the purpose of these constraints, the basic principles of DC motors are explained briefly. The torque of a DC motor τ_m is directly analogous to the intensity of the current i_a on its field windings [70],

$$\tau_m = K_T i_a \tag{2-91}$$

where K_T is the torque constant. The power of the thermal losses on the field windings P_{th} is analogous to the square of the current i_a^2,

$$P_{th} = i_a^2 R \tag{2-92}$$

where R is the resistance of the field windings. As thermal losses increase, so does the temperature of the field windings. To protect the circuit of the DC motor from overheating, maximum permissible continuous and short term values for the current are introduced, $i_{a,ct}$ and $i_{a,st}$, corresponding to maximum permissible continuous and short term torques $\tau_{m,ct}$ and $\tau_{m,st,th}$,

$$\tau_{i_a,ct} = K_T i_{a,ct} \tag{2-93}$$

$$\tau_{i_a,st} = K_T i_{a,st} \tag{2-94}$$

To increase the output torque of a DC motor without increasing the current on its windings reduction mechanisms such as gearboxes are used. The reduction mechanism, reduces the output angular velocity of the motor n_t times (reduction ratio) and since the total power of the DC motor is constant, the output torque is increased correspondingly by n_t times, see Equations (2-95)-(2-97).

$$\dot{\theta}_{m,out} = \dot{\theta}_{m,in} / n_t \tag{2-95}$$

$$\tau_{m,out} = n_t \tau_{m,in} \tag{2-96}$$

$$P_m = \tau_{m,out}\, \dot{\theta}_{m,out} = \tau_{m,in}\, \dot{\theta}_{m,in} \tag{2-97}$$

Nevertheless, the reduction mechanism consists of mechanical parts (bearings, gears) that should be protected from overstressing. Therefore, maximum permissible output torques and angular velocities are set for the exit of the gearbox $\tau_{\max}$ and $\dot{\theta}_{\max}$. As a result, a total maximum short term torque should be allowed at the exit of the reduction, so that the DC motors system is protected from overheating and overstressing,

$$\tau_{st} = \min(n_t \tau_{i_a,st}, \tau_{\max}) \tag{2-98}$$

The permissible continuous torque is also calculated at the exit of the reduction,

$$\tau_{ct} = n_t \tau_{i_a,ct} \tag{2-99}$$

By applying these upper bound values to the torques and angular velocities of the quadruped robot's leg joints,

$$\begin{gathered} \dot{\theta}_{j,i} \le \dot{\theta}_{\max,j},\ t \in [0,T] \\ \tau_{j,i} \le \tau_{st,j},\ t \in [0,\ T] \\ \mathrm{rms}(\tau_{j,i}) \le \tau_{ct,j},\ t \in [0,\ T] \end{gathered} \tag{2-100}$$

where

$$\mathrm{rms}(\tau_{j,i}) = \sqrt{\frac{1}{N}(\tau_{j,i,1}^2 + \tau_{j,i,2}^2 + \ldots + \tau_{j,i,N}^2)} \tag{2-101}$$

and N is the number of sampling points of the torque $\tau_{j,i}$ in the duration of a stride period T.

By defining the hardware constraints, (2-90) and (2-100), the description of the physical system is concluded. The gaits, leg architecture, dynamics, inverse kinematics and constraints applying to the quadruped robot presented in this chapter, are to be used in the development of the leg design method presented in the next chapter.

3 Optimal Leg Design Method

3.1 Overview

Given some overall specifications for the quadruped robot's inertial properties the purpose of the developed method is to determine optimal leg properties (leg segment lengths, configuration, leg segment cross sectional area and passive element stiffness) coupled with optimal gait properties (gait, stride period, CoM apex running height and amplitude of horizontal ground forces), in order to achieve *maximum horizontal velocity*. The method consists of one outer and two inner stages.

At the beginning of the method the gait is defined by determining the time instants of touch down $t_{td,i}$ and take off $t_{to,i}$ of every leg. Furthermore, the initial conditions of the motion $x(0)$, $\dot{y}(0)$, $\theta(0)$, $\dot{\theta}(0)$ are set. These values are provided as inputs to the following stages. In the Outer Stage the gait parameter space is spanned in the form of an exhaustive search and alternative *gait parameters* $\dot{x}(0)$, $y(0)$, a_{fx}, T are provided as inputs to the first and second inner stages.

In the first inner stage (Stage 1), using the centroidal dynamics of the robot, optimal footfalls x_i are sought through exhaustive search and optimization, which achieve periodical movement in one stride for the robot's CoM and minimization of the hip joint torques necessary to withstand the vertical ground forces exerted on each leg. Stage 1 provides as outputs the optimal footfalls x_i and the optimized CoM trajectory of the robot, which are used as inputs to the second inner stage.

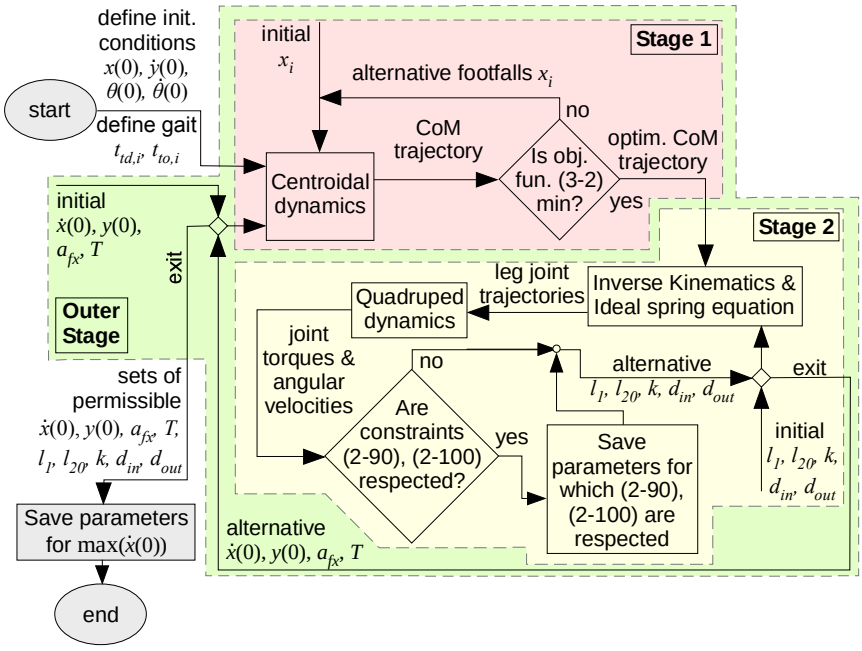


Figure 3-1. Flow chart indicating the discrete stages of the leg design method.

The main operation conducted in the second inner stage (Stage 2) is to exhaustively search the *leg parameters* l_1, l_{20}, k, d_{in}, d_{out} for which the hardware constraints are respected. At the beginning of Stage 2, a trajectory is determined for the toe of each leg to follow during flight phase. Thus, with the position of the toe and that of the CoM determined throughout the stride, given the robot's geometry, the leg joint trajectories can be calculated throughout the stride, using inverse kinematics. The joint

trajectories are found for alternative leg parameters, and through the dynamics of the quadruped robot the joint torques are calculated and the actuation constraints are evaluated. From the posture of each leg and the forces exerted from the ground, the stresses applied on the cross sectional area of each leg segment are calculated and subjected to the strength constaints incurred by the material of the leg segments. If both actuation and strength constraints are satisfied, then the set of leg parameters l_1, l_{20}, k, d_{in}, d_{out} and the corresponding gait parameters $\dot{x}(0)$, $y(0)$, a_{fx}, T that satisfy the constraints are saved. This procedure is repeated until the whole leg parameter space is spanned, concluding thus Stage 2.

When the whole gait parameter space is spanned, the outer stage comes to an end. From the saved parameters that satisfy the hardware constraints, those with which the robot could run with maximum velocity, is the set of optimum parameters.

3.2 Initialization

At the beginning of the method, the model properties, the initial conditions and the gait are defined. The parameters intended for initialization and their respective description are presented in Table 3-1. Those parameters determine the systems specifications and remain constant throughout the execution of the algorithm. Note that the time instants defining the touch down and take off of each leg, (thus defining the gait) are expressed as a percentage of the period,

$$t_{td,i}\% = \frac{t_{td,i}}{T}100\% \qquad t_{to,i}\% = \frac{t_{to,i}}{T}100\% \tag{3-1}$$

As a result, each time the period T takes a value in the following outer stage, Equations (3-1) are solved to $t_{td,i}$, $t_{to,i}$.

Table 3-1. Parameters and initial conditions determined in the initialization stage.

Parameter	Decription	Parameter	Description
m [kg]	overall mass of the robot	b_e [m]	maximum leg clearance
d [m]	half hip separation distance	μ	Coulomb friction coefficient
g [m/s^2]	gravitational acceleration	n_i	transmission ratio of the motor acting on the j-th joint
a [m]	height of the robot body	$I_{r,j}$ [kg m^2]	rotor inertia of the motor acting on the j-th joint
b [m]	length of the robot body	$\tau_{ct,j}$ [N/m]	max permissible short term torque on the j-th joint
I [kg m^2]	overall inertia of the robot	$\tau_{ct,j}$ [N/m]	max permissible continuous torque on the j-th joint
ρ [kg/m^2]	density of the leg segment's material	$\dot{\theta}_{\max,j}$ [rad/s]	max permissible continuous angular velocity on the j-th joint

Parameter	Decription	Parameter	Description
$S_{c,u}$ [MPa]	compressive strength	$t_{td,i}$ %	touch down time instant of the i-th leg, as a percentage of the period
$\tau_{s,u}$ [MPa]	shear strength	$t_{to,i}$ %	take off time instant of the i-th leg, as a percentage of the period
s_f	safety factor	$x(0)$ [m]	initial horizontal position of CoM
m_{j1} [kg]	mass of hip joint	$\dot{y}(0)$ [m/s]	initial vertical velocity of CoM
m_{j2} [kg]	mass of knee joint	$\theta(0)$ [rad]	initial pitch angle of centroidal model
m_s [kg]	spring mass	$\dot{\theta}(0)$ [rad/s]	initial pitch rate of centroidal model
m_t [kg]	toe mass		

3.3 Outer stage

In the outer stage of the method, an exhaustive search takes place spanning the space of the gait parameters $\dot{x}(0)$, $y(0)$, a_{fx}, T . These parameters are provided as inputs to Stage 1 and Stage 2. The exhaustive search taking place at this stage is shown in the form of pseudocode in Figure 3-2. The subscripts lb, ub correspond to the lower and upper bounds determining the parameter space in which the parameters were sought. At the end of every inner loop of this stage the sets of gait parameters $\dot{x}(0)$, $y(0)$, a_{fx}, T that satisfy the hardware constraints, and the corresponding sets of leg parameters l_1, l_{20}, k, d_{in}, d_{out}, found in Stage 2, which satisfy the hardware constraints are saved. Note that for the same set of gait parameters, alternative sets of leg parameters may satisfy the constraints. The outer stage is concluded when the whole gait parameter space has been spanned.

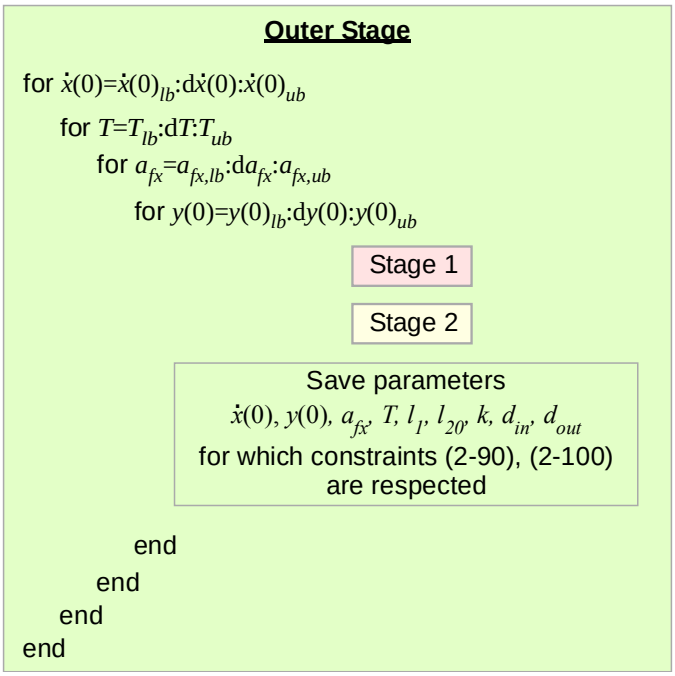


Figure 3-2. Pseudocode showing the exhaustive search conducted in the Outer Stage.

3.4 Stage 1: CoM trajectory and footfall optimization

In this stage, the positions of the legs' footfalls are sought which optimize the trajectory of the robot's CoM, so that its movement is periodical in one stride and the hip torques necessary to withstand the ground forces are minimized. The optimization procedure that takes place in this stage consists of two steps. It starts with an exhaustive search step that narrows down the parameter space in which the optimal values of x_i lie. An optimization step follows, where the optimal values x_i are found with fine accuracy in the narrow parameter space of the previous step. If a single step gradient based optimization was used instead of this two step approach, the algorithm would most probably converge to a local optimum. In the remaining section, the role of the footfalls in the movement of the quadruped robot is explained, the optimization problem is formulated and the two step optimization procedure is thoroughly explained.

Since the locomotion of the robot takes place in even terrain $(y_i = 0)$, the position of each footfall is determined exclusively from its abscissa x_i. The role of the position of the footfalls is twofold. On the one hand, it regulates the pitch and pitch rate of the centroidal model, see (2-42), (2-45). On the other hand, its distance from the respective hip determines the magnitude of the necessary torque to withstand the vertical ground forces. An example is depicted in Figure 3-3, where for the same vertical force $F_{y,1}$ and traversed distance during stance δx, footfall x_1 constantly has a shorter moment arm with respect to the hip joint than footfall $x_1^{'}$, thus resulting to smaller hip torques for the same ground force.

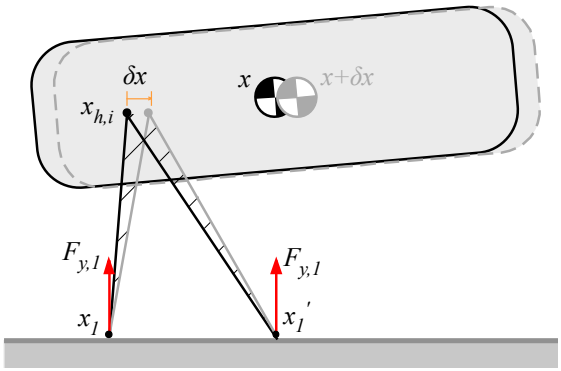


Figure 3-3. Alternative footfalls x_1, $x_1^{'}$ for the same traversed distance δx and vertical force $F_{y,1}$.

With the twofold role of the footfalls x_i in mind, we proceed to find the optimal values $x_{i,o}$ which meet the following requirements; periodicity of movement in the θ direction and minimization of the horizontal moment arms from the respective hip joints at every time instant. To meet these requirement a minimization problem is formulated,

$$\min_{x_i} Q = w_1 \left|\theta(\mathrm{T}) - \theta(0)\right| + w_2 \left|\dot{\theta}(\mathrm{T}) - \dot{\theta}(0)\right| + w_3 \sum_{i=1}^{4} \int_{t_{td,i}}^{t_{to,i}} \left|x_{h,i} - x_i\right| dt \qquad (3\text{-}2)$$

where w_1, w_2, w_3 are the weighing factors of the optimized terms. The first two weighing factors w_1, w_2, are selected so that the first two terms are of approximately the same magnitude and the third weighing factor w_3 is determined through trial and error. By minimizing the first two terms, we ensure

that periodicity conditions in the θ direction (2-57), (2-58) hold. The minimization of the third term ensures that the horizontal moment arms of the vertical force $F_{y,i}$ from the hip joint at every time instant are minimized, thus also minimizing the necessary hip joint torques. Note that the component of the horizontal force $F_{x,i}$ and the vertical moment arm $\left|y_{h,i} - y_i\right|$ is not included in the hip joint torques, as the vertical position of the hip joint $y_{h,i}$ depends on the height of running $y(0)$ which is optimized in the Outer Stage.

In the first step of the optimization procedure, an exhaustive search is conducted to narrow down the parameter space in which the optimal footfall abscissas x_i lie. At every loop of the exhaustive search, the EoM of the centroidal dynamics (2-40)-(2-42) are solved, the position of the hip joints is calculated from Equations (2-21)-(2-24) and the objective function is evaluated. At the end of the exhaustive search (es) a crude estimation of the optimal footfalls $x_{i,es}$ is available. As this step preceeds the optimization step, the requirement of this step is not accuracy but speed of solution. To achieve this requirement the following measures are proposed. The footfall abscissas x_i are sought in a vicinity of the corresponding hip joints at the time instant of touch down $x_{h,i}(t_{td,i})$, defined using the half hip separation distance d,

$$x_i \in [x_{h,i}(t_{td,i}) - d/2,\ x_{h,i}(t_{td,i}) + 2d] \tag{3-3}$$

with discretization step dx_i. Note that this interval is set through trial and error, so that no optimal value $x_{i,es}$ is truncated by its upper and lower bounds. At every loop of the exhaustive search, the EoM of the centroidal dynamics are solved numerically using three point finite difference expressions for N = 100 integration points in the time interval $[0, T]$. A pseudocode example for the case of the trotting gait of Figure 2-2 (d) is displayed in Figure 3-4. In this example, the leg pair {1, 3} is in contact with the ground for the first half of the period $(n = 1:50)$, succeeded by the leg pair {2, 4} for the second half of the period $(n = 51:100)$. In the beginning of the exhaustive search, given the initial position of the CoM $x(1)$ the abscissas of the hip joints $x_{h,1}(1)$, $x_{h,3}(1)$ are calculated from (2-21)-(2-24). Values are set for x_1, x_3 in (3-3) and the centroidal dynamics EoM (2-40)-(2-42) are solved for the first half of the period. The supplementary hip joint abscissas $x_{h,2}(50)$, $x_{h,4}(50)$ are calculated from the position of the CoM at the end of the first half of the period $x(50)$, the boundaries of the intervals (3-3) are found, values are set for x_2, x_4 and the EoM of the centroidal dynamics are solved until the end of the period. Having solved the centroidal EoM in one period for the candidate abscissas x_1, x_2, x_3, x_4, the objective function value is calculated from Equation (3-2), and is evaluated. At the end of the exhaustive search, a crude estimation of the minimum value of the objective function and the footfall abscissas $x_{i,es}$ corresponding to the minimum value of the objective function are found.

The values of the footfalls $x_{i,es}$ found in the first step of Stage 1, are used as initial values for the optimization taking place in the second step of Stage 1. The upper and lower bounds in which the optimal leg abscissas are sought are defined with respect to the discretization step dx_i of the preceding exhaustive search as,

$$x_i \in [x_{i,es} - dx_i,\ x_{i,es} + dx_i] \tag{3-4}$$

Stage 1/Step 1

```
minQ=1000;

for x_1=x_h,1(1)-d/2:dx_1:x_h,1(1)+3*d/2
   for x_3=x_h,3(1)-d/2:dx_3:x_h,3(1)+3*d/2

      for n=1:50
         calculate F_x, F_y, τ_θ  (2-42)-(2-44)
         solve EoM (2-39)-(2-41)
      end

      for x_2=x_h,2(1)-d/2:dx_2:x_h,2(1)+3*d/2
         for x_4=x_h,4(1)-d/2:dx_4:x_h,4(1)+3*d/2

            for n=51:100
               calculate F_x, F_y, τ_θ  (2-42)-(2-44)
               solve EoM (2-39)-(2-41)
            end

            calculate the objective function Q (3-2)

            if Q<minQ
              save Q as minQ
              save currently optimum abscissas as x_i,es
            end

         end
      end

   end
end
```

Figure 3-4. Pseudocode of the exhaustive search in the first step of Stage 1.

The minimization problem (3-2) is solved using the default interior point algorithm of the `fmincon` MATLAB function. At every loop of the optimization process, for the candidate footfall abscissas x_i the EoM of the centroidal dynamics (2-40)-(2-42) are solved with fine absolute error tolerance 1e-6, using the `ode45` solver of MATLAB, based on the Runge-Kutta 45 integration method. The value of the objective function is calculated and evaluated. At the end of the optimization process, the optimal values $x_{i,o}$ that minimize the objective function (3-2) are provided as outputs.

Using the optimal footfall abscissas $x_{i,o}$ the EoM of the centroidal dynamics are solved for one more time. The EoM at this point are solved with the fixed step MATLAB function `ode3`, based on the Runge-Kutta algorithm of third order, for N =1000 integration points to achieve the desired accuracy of results. The choice of a fixed step integration algorithm is convenient for two reasons. On one hand, each row of the solution vectors is intuitively related with its corresponding time instance in the stride. For instance, for the trotting gait of Figure 2-2 (d), it is easy to understand that the solution at the 250th time instant, corresponds to the 25% of the period, where legs HL, FR are at midstance and legs FL, HR are at midflight. On the other hand, the use of a fixed step integration algorithm facilitates the use of the MATLAB functions used in the next stage that need fixed step vectors as inputs.

For the robot to safely remain away from the ground throughout the stride, a maximum permissible pitch constraint is introduced in the form of inequality (3-5).

$$y\big|_{\theta_{\max}} - \frac{a}{2}\sin(\theta_{\max}) - \frac{b}{2}\cos(\theta_{\max}) \geq \frac{y\big|_{\theta_{\max}}}{2} \tag{3-5}$$

This constraint ensures that the lowest part of the robot body at maximum pitch, is not situated lower than half the vertical distance of the CoM from the ground at max pitch, see Figure 3-5. If constraint

(3-5) is satisfied, the external forces $F_{x,i}$, $F_{y,i}$ and the optimized CoM trajectories $\mathbf{q}_C = [x\, y\, \theta]$, $\dot{\mathbf{q}}_C$, $\ddot{\mathbf{q}}_C$ at discrete time instants $t_n \in [0, T]$, $n \in \{1, 2, ..., 1000\}$ are saved for use in Stage 2.

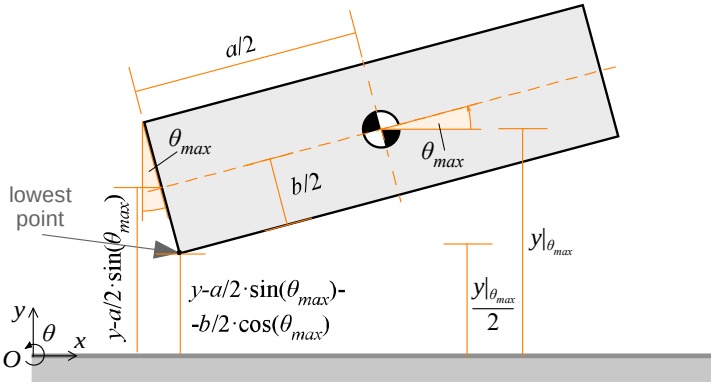


Figure 3-5. Position of the lowest part of the robot body at max pitch in comparison with half the vertical distance of the CoM from the ground.

3.5 Stage 2: Spanning the leg parameter space

In the second stage of the method, alternative leg geometries are tested for use in the robot. The optimized CoM trajectory $\mathbf{q}_C$, $\dot{\mathbf{q}}_C$, $\ddot{\mathbf{q}}_C$, the footfall abscissas $x_{i,o}$ and the ground forces $F_{x,i}$, $F_{y,i}$ corresponding to gait parameters $\dot{x}(0)$, $y(0)$, a_{fx}, T are provided as inputs. The position of the toe of each leg during its stance phase is determined from the pair $(x_{i,o}, 0)$. To fully define the trajectory of the toes during the whole stride, a hybrid elliptical/cubic polynomial trajectory for the toes is adopted throughout flight phase. The position of the hip joints throughout the stride are found by inserting the position of the CoM $\mathbf{q}_C$, in Equations (2-21)-(2-24). Given the position of the hip joint $(x_{h,i}, y_{h,i})$ and that of the toe of each leg $(x_{t,i}, y_{t,i})$ at discrete time instants t_n, for alternative leg segment lengths and spring stiffnesses l_1, l_{20}, k, from inverse kinematics (2-75), (2-76) and the equation of the ideal spring, the leg joint variables $\theta_{1,i}$, $\theta_{2,i}$, $l_{2,i}$ are found at every time instant $t_n \in [0, T], n \in \{1, 2, ..., 1000\}$. By numerically differentiating the vector of leg joint variables $\mathbf{q}_i = [\theta_{1,i}\, \theta_{2,i}\, l_{2,i}]$, the vectors of leg joint velocities $\dot{\mathbf{q}}_i$ and accelerations $\ddot{\mathbf{q}}_i$ are found. For each set of l_1, l_{20}, k, corresponding to leg postures $\mathbf{q}_i$, the ground forces $F_{x,i}$, $F_{y,i}$ are projected on the leg segments (Equations (2-79)-(2-82)) and for alternative leg segment diameters d_{in}, d_{out} strength constraints (2-90) are tested. Every permissible set of l_1, l_{20}, k, d_{in}, d_{out} is inserted in the EoM of the quadruped robot (2-38), along with the ground forces $F_{x,i}$, $F_{y,i}$ the CoM trajectory $\mathbf{q}_C$, $\dot{\mathbf{q}}_C$, $\ddot{\mathbf{q}}_C$, the joint trajectories $\mathbf{q}_i$, $\dot{\mathbf{q}}_i$, $\ddot{\mathbf{q}}_i$ and the model parameters, and the EoM (2-38) are solved to the leg joint torques $\tau_{j,i}$. The angular velocities $\dot{\theta}_{j,i}$ and torques $\tau_{j,i}$ are subjected to actuation constraints (2-100). If actuation constraints are also satisfied then the set of leg parameters l_1, l_{20}, k, d_{in}, d_{out} is saved. After the whole leg parameter space is spanned, Stage 2 is terminated and all the permissible sets of l_1, l_{20}, k, d_{in}, d_{out}, corresponding to each set of gait parameters $\dot{x}(0)$, $y(0)$, a_{fx}, T are saved. In the following analysis each of the above steps is described in detail.

Stage 2 is structured in the form of an exhaustive search for the leg parameters l_1, l_{20}, k. From the optimized positions of the robot's CoM $\mathbf{q}_C$, corresponding to footfalls $(x_{i,o}, y_i)$ and the positions of the hip joints (2-21)-(2-24), the hip to toe distance of every leg during stance phase is determined,

$$l_{h2t,i} = \sqrt{(x_{h,i} - x_i)^2 + (y_{h,i} - y_i)^2} \tag{3-6}$$

The upper bound of each leg's workspace (inequalities (2-78)) is the maximum value of $l_{h2t,i}$. As all legs have the same leg segment lengths l_1, l_{20} and compliance k, the maximum value of $l_{h2t,i}$ for every leg i and time instant t_n determines the minimum permissible total leg length $l_1 + l_{2,i}$,

$$\begin{aligned} l_1 + l_{2,i}(t_n) &\geq l_{\max}, \\ l_{\max} &= \max_i [\max_{t_n} (l_{h2t,i})] \end{aligned} \tag{3-7}$$

If the intervals in which l_1, l_{20} are sought are set in the form,

$$\begin{aligned} l_1 &\in [l_{1,lb},\ l_{1,ub}] \\ l_{20} &\in [l_{20,lb},\ l_{20,ub}] \end{aligned} \tag{3-8}$$

many of the candidate solution combinations that lay near the lower bounds of the search do not satisfy inequality (3-7) and are rejected making the exhaustive search algorithm slow and counterproductive. On the other hand, candidate solutions near $l_{1,lb}$, $l_{20,lb}$ should not be excluded from the search, as perhaps inequality (3-7) is satisfied for candidate combinations in vicinities of pairs $(l_{1,lb}, l_{20,ub})$ or $(l_{1,ub}, l_{20,lb})$. An example is presented to better understand the disadvantage of formulation (3-8). Let's suppose that

$$\begin{aligned} l_{\max} &= 0.5m \\ l_1 &\in [0.1m, 0.5m] \\ l_{20} &\in [0.1m, 0.5m] \end{aligned} \tag{3-9}$$

It is clear that many of the combinations near the lower bounds of l_1, l_{20} do not satisfy (3-7) and should be rejected. Nevertheless, combinations of (l_1, l_{20}) in vicinities of $(0.1, 0.5)$ or $(0.5, 0.1)$ respect (3-7) and therefore truncation of the lower bounds of the search intervals is not justified. To overcome this obstacle, the exhaustive search problem is reformed by introducing the ε parameter, as shown in (3-10).

$$\begin{aligned} l_1 &\in [l_{1,lb}, \varepsilon l_{\max} - l_{20,lb}] \\ \varepsilon &\in [1, \varepsilon_{ub}] \\ l_{20} &= \varepsilon l_{\max} - l_1 \end{aligned} \tag{3-10}$$

As the distal leg segment with free length l_{20} is compliant, the formulation (3-10) does not necessarily guarantee that (3-7) is respected if $\varepsilon \simeq 1$. Nevertheless, the number of the rejected combinations (l_1, l_{20}) of (3-10) is particularly smaller than that of (3-8), and thus the speed of the exhaustive search algorithm is significantly improved.

In every inner loop of the exhaustive search, the leg joint trajectories are found from the optimized CoM trajectory, given the robot's geometry. The positions of the hips $(x_{h,i}, y_{h,i})$ and the toes $(x_{t,i}, y_{t,i}) \equiv (x_{i,o}, 0)$ during stance phase, are available at every time instant t_n from Stage 1. As the distal leg segment is compliant, the leg geometry changes throughout stance phase. To predict the deflection of the spring throughout stance the ideal spring equation is introduced,

$$F_{s,i} = k(l_{2,i} - l_{20}) \tag{3-11}$$

As the toe remains in contact with the ground throughout stance,

$$F_{s,i} = F_{l,2,i} \tag{3-12}$$

Then from Equations (2-81), (3-11), (3-12) the length of the compliant leg segment is found at every time instant,

$$\mathrm{l}_{2,i} = l_{20} + \frac{1}{k}\left[F_{x,i} \sin(\theta + \theta_{2,i}) - F_{y,i} \cos(\theta + \theta_{2,i}) \right] \tag{3-13}$$

Combining the inverse kinematics Equations (2-75), (2-76) with (3-13), the leg joint variables $\theta_{1,i}$, $\theta_{2,i}$, $l_{2,i}$ are found at every time instant throughout the stance phase of each leg, $t_n \in [t_{td,i}, t_{to,i}]$. By numerically differentiating the leg joint variables, the joint velocities $\dot{\theta}_{1,i}$, $\dot{\theta}_{2,i}$, $\dot{l}_{2,i}$ and accelerations $\ddot{\theta}_{1,i}$, $\ddot{\theta}_{2,i}$, $\ddot{l}_{2,i}$ are found during stance phase. For first order and second order differentiation the MATLAB functions `gradient` and `del2` were used. At this point the knee joint is checked to not collide with the ground through stance phase, expressed in the form of an inequality,

$$\min_{t_n}\left[90^o - \left|\theta + \theta_{2,i}\right| \right] \geq 5^o \tag{3-14}$$

where $90^o - \left|\theta + \theta_{2,i}\right|$ is the absolute angle of the distal leg segment from the ground, see Figure 3-6. If inequality (3-14) is respected, then we proceed to calculate the leg joint trajectories throughout flight phase. If not, the current combination of the exhaustive search l_1, $l_{20,i}$, k is rejected and the procedure is repeated for the next combination of l_1, $l_{20,i}$, k provided from the exhaustive search algorithm.

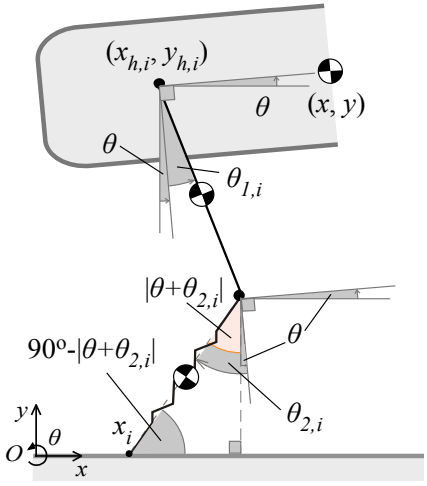


Figure 3-6. Trigonometry for the distal leg segment.

Throughout the flight phase, a mechanical stop is considered to keep the length of the distal leg segment constant,

$$l_{2,i} = l_{20},\ t_n \in [t_{to,i},\ t_{td,i} + T] \tag{3-15}$$

The requirement for each leg during flight phase is to avoid collision with the ground by creating the necessary toe to ground clearance, following a first order continuous trajectory; non-continuous angular velocities would require infinite torques. Inspired by control schemes in which the reference toe trajectory during flight is an ellipse [71], a hybrid cubic polynomial - elliptical flight phase trajectory is selected. The cubic polynomial part ensures continuity for angular velocities, and the elliptical part provides realistic boundary conditions at midflight.

Having found the leg joint trajectories during the preceding stance phase from $t_{td,i}$ to $t_{to,i}$, and because the stride is a periodical phenomenon, the joint angles and angular velocities at the beginning $t_{to,i}$ and the end of the flight phase $t_{td,i}+T$ are available,

$$\begin{aligned}
&\theta_{1,i}(t_{to,i}),\ \theta_{1,i}(t_{td,i}+\mathrm{T})=\theta_{1,i}(t_{td,i})\\
&\dot{\theta}_{1,i}(t_{to,i}),\ \dot{\theta}_{1,i}(t_{td,i}+\mathrm{T})=\dot{\theta}_{1,i}(t_{td,i})\\
&\theta_{2,i}(t_{to,i}),\ \theta_{2,i}(t_{td,i}+\mathrm{T})=\theta_{2,i}(t_{td,i})\\
&\dot{\theta}_{2,i}(t_{to,i}),\ \dot{\theta}_{2,i}(t_{td,i}+\mathrm{T})=\dot{\theta}_{2,i}(t_{td,i})
\end{aligned} \tag{3-16}$$

At midflight $t_{mf,i}$ the toe is considered to be at the apex of an elliptical trajectory, described by Equations (3-17), (3-18),

$$x_{e,i}=x_{c,i}+a_e\cos(\omega_e t+\varphi_{e,i}) \tag{3-17}$$

$$y_{e,i}=y_{c,i}+b_e\sin(\omega_e t+\varphi_{e,i}) \tag{3-18}$$

where $(x_{c,i},y_{c,i})$ are the coordinates of the center of the ellipse, a_e and b_e are the semi-axes of the ellipse, ω_e the circular frequency in which the ellipse is traversed and $\varphi_{e,i}$ the phase difference that determines the starting point on the ellipse. The center of the ellipse is situated at ground level, at distance a_e from the footfall of the preceding stance phase,

$$(x_{c,i},y_{c,i})\equiv(x_i(t_{to,i})+a_e,0) \tag{3-19}$$

In one stride, the horizontal distance traversed from the CoM of the robot and each toe are the same and equal to $x(T)-x(0)$, see Figure 3-7. The horizontal ellipse semi axis is equal to half that distance,

$$a_e=(x(T)-x(0))/2 \tag{3-20}$$

and the vertical semi axis b_e is equal to maximum desired clearance from the ground. The ellipse is traversed in the counter clockwise direction with a circular frequency ω_e defined in (3-21),

$$\omega_e=-\frac{2\pi}{T_e}=-\frac{2\pi}{2(T-\delta t_s)}=-\frac{\pi}{T(1-DF)} \tag{3-21}$$

where T_e is the period in which the ellipse is traversed. From the value of the argument $\omega_e t+\varphi_{e,i}$ at the time instant of take off the phase difference is found,

$$\begin{aligned}
&\omega_e t_{to,i}+\varphi_e=\pi\\
&\varphi_e=\pi-\omega_e t_{to,i}
\end{aligned} \tag{3-22}$$

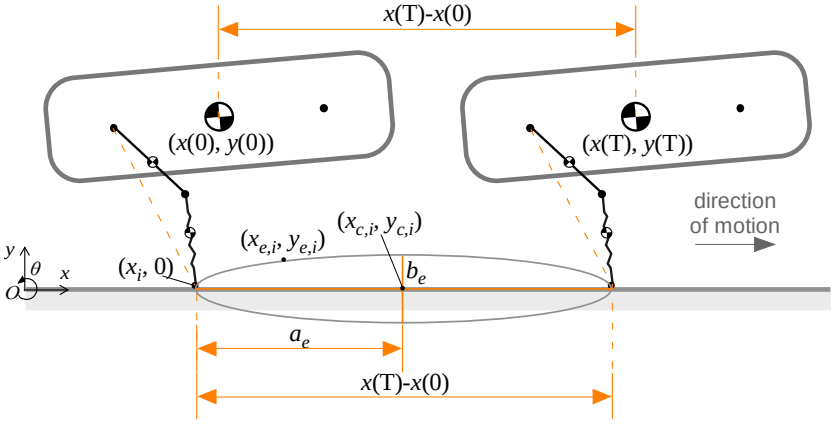


Figure 3-7. The elliptical flight phase trajectory and its properties.

From inverse kinematics (2-75), (2-76) and the relative position of the hip joints (2-21)-(2-24) to the position of the toes during flight phase (3-17), (3-18), the leg joint angles $\theta_{1e,i}$, $\theta_{2e,i}$ of the elliptical

trajectory are found. By numerically differentiating $\theta_{1e,i}$, $\theta_{2e,i}$, the joint angular velocities $\dot{\theta}_{1e,i}$, $\dot{\theta}_{2e,i}$ are found. Because the joint angular velocities at take off and touch down of the elliptical trajectory are not continuous, to introduce first order continuity only the values of joint variables and their corresponding angular velocities at midflight are kept,

$$\begin{aligned}\theta_{1,i}(t_{mf,i}) &= \theta_{1e,i}(t_{mf,i}), \dot{\theta}_{1,i}(t_{mf,i}) = \dot{\theta}_{1e,i}(t_{mf,i}) \\ \theta_{2,i}(t_{mf,i}) &= \theta_{2e,i}(t_{mf,i}), \dot{\theta}_{2,i}(t_{mf,i}) = \dot{\theta}_{2e,i}(t_{mf,i})\end{aligned} \tag{3-23}$$

and the states at touch down, midflight and take off (3-16), (3-23) are interpolated using cubic polynomials which can ensure first order continuity. One cubic polynomial is used from take off to midflight,

$$\begin{aligned}\theta_{j,i} &= a_1 t^3 + b_1 t^2 + c_1 t + d_1 \\ \dot{\theta}_{j,i} &= 3a_1 t^2 + 2b_1 t + c_1\end{aligned}, \; t \in [t_{to,i}, t_{mf,i}] \tag{3-24}$$

and another from midflight to touch down,

$$\begin{aligned}\theta_{j,i} &= a_2 t^3 + b_2 t^2 + c_2 t + d_2 \\ \dot{\theta}_{j,i} &= 3a_2 t^2 + 2b_2 t + c_2\end{aligned}, \; t \in [t_{mf,i}, t_{td,i} + T] \tag{3-25}$$

By applying boundary conditions at take off, midflight and touch down (3-16), (3-23), the coefficients of the polynomials in (3-24), (3-25) are found. By differentiating the angular velocities, the joint accelerations with time are found too,

$$\begin{aligned}\ddot{\theta}_{j,i} &= 6a_1 t + 2b_1, \; t \in [t_{to,i}, t_{mf,i}] \\ \ddot{\theta}_{j,i} &= 6a_2 t + 2b_2, \; t \in [t_{mf,i}, t_{td,i} + T]\end{aligned} \tag{3-26}$$

As it can be seen for the example of a leg with $l_1 = 0.32\,\mathrm{m}$ and $l_{20} = 0.3\,\mathrm{m}$ (Figure 3-8), the non-continuities observed in the hip joint angular velocity with the elliptical toe trajectory are eliminated with the hybrid elliptical - cubic polynomial approach.

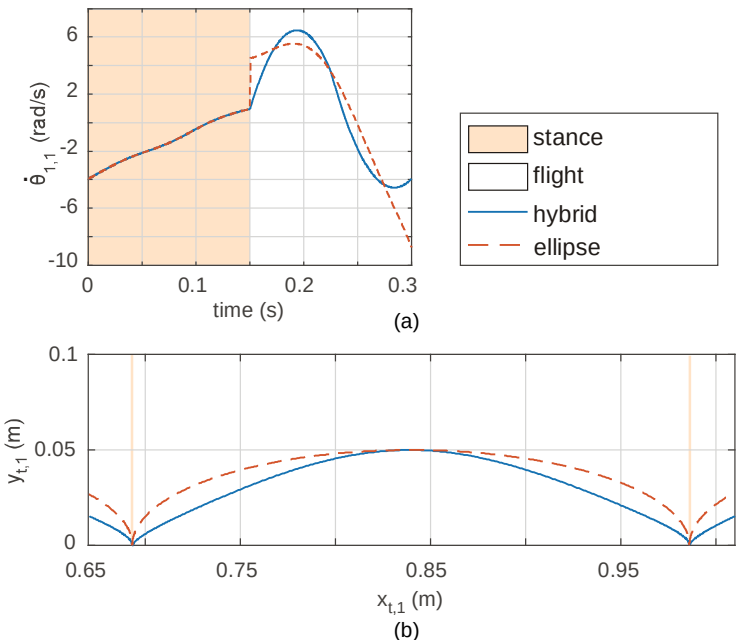


Figure 3-8. Examples of the (a) evolution of hip angular velocity with time and the (b) toe path, for elliptical and hybrid toe trajectories.

At the end of the procedure described above, the vectors of leg joint variables $\mathbf{q}_i = [\theta_{1,i}\ \theta_{2,i}\ l_{2,i}]$, joint velocities $\dot{\mathbf{q}}_i$ and accelerations $\ddot{\mathbf{q}}_i$ are available throughout the stride. After the leg joint trajectories are found, hardware constraints (2-90), (2-100) are checked. From the posture of each leg during stance phase found in $\mathbf{q}_i$, the ground forces applied on the toes $F_{x,i}$, $F_{y,i}$ and Equations (2-79) -(2-82), the longitudinal and vertical forces exerted on the leg segments are calculated, $F_{l,j,i}$, $F_{v,j,i}$. Alternative combinations of d_{in}, d_{out} for standard carbon tubes are tested, giving priority to those with thinner cross sectional area, resulting in lighter leg segments. For every option of d_{in}, d_{out} the stresses on the leg segments are calculated (2-85)-(2-87) and are subjected to strength constraints (2-90). If strength constraints are respected, the combination of d_{in}, d_{out} is accepted, otherwise we proceed to a combination resulting to a thicker cross sectional area, until strength constraints (2-90) are respected.

The set of leg parameters l_1, l_{20}, k, d_{in}, d_{out} and the corresponding leg joint trajectories $\mathbf{q}_i$, $\dot{\mathbf{q}}_i$, $\ddot{\mathbf{q}}_i$ and CoM trajectories $\mathbf{q}_C$, $\dot{\mathbf{q}}_C$, $\ddot{\mathbf{q}}_C$ are replaced in the EoM of the quadruped (2-38), from which the vector of the necessary actuation torques $\boldsymbol{\tau}$ for the locomotion of the robot in one stride are calculated. For every actuated joint of every leg, the actuation torques $\tau_{j,i}$ found in $\boldsymbol{\tau}$ and the angular velocities $\dot{\theta}_{j,i}$ found in $\dot{\mathbf{q}}_i$, are subjected to the actuation constraints (2-100). If the actuation constraints are respected, then the combination of parameters l_1, l_{20}, k, d_{in}, d_{out} is saved. At the end of the exhaustive search of the leg parameters, all the persmissible sets of l_1, l_{20}, k, d_{in}, d_{out} corresponding to the current set of gait parameters $\dot{x}(0)$, $y(0)$, a_{fx}, T are saved and the Stage 2 of the method exits to the Outer Stage.

3.6 Concluding the method

The method concludes after exiting the Outer Stage. At the end of the method, all sets of gait parameters $\dot{x}(0)$, $y(0)$, a_{fx}, T and their corresponding permissible sets of leg parameters l_1, l_{20}, k, d_{in}, d_{out} are saved. In this way, trends of every parameter can be drawn with increasing horizontal CoM velocity $\dot{x}(0)$, and the effect of every parameter in $\dot{x}(0)$ can be studied. The set of gait parameters and the corresponding set or sets of leg parameters for which maximum velocity $\dot{x}(0)$ is achieved are considered the optimal solutions of this method.

4 Results

4.1 Introduction

The proposed leg design methodology is applied in this chapter for two alternative running gaits, bounding and trotting, and two alternative knee configurations, the knee backward (KB) and the knee forward (KF) configuration for the same model parameters. Two identical motors are acting on the hip and the knee joint of each leg. The purpose of this chapter is to find out which gait, knee configuration, leg and gait parameters result in the fastest locomotion velocity for a robot of the overall mass and inertia of *NTUA Laelaps*, while respecting constraints (2-90), (2-100). For each case, the evolution of the permissible gait and leg parameters with horizontal CoM velocity is presented and studied. The optimal leg and gait parameters for which maximum velocity is achieved are found and the locomotion of the robot with these optimal sets is presented. The optimal results for alternative gaits and knee configurations are compared, and the combination of gait, knee configuration, gait parameters and leg parameters for which the maximum horizontal velocity is reached is found. This overall optimal solution is evaluated using an independent to the method quadruped model including damping, a realistic ground model and a P-V controller.

4.2 System parameters

In this section, the values of the system's parameters for which the method is applied are presented. These parameters are used for both gaits and knee configurations. The overall mass and geometry properties are taken similar to these found in the current version of the Laelaps quadruped, see Table 4-1. The density of the leg segments is taken similar to those of carbon fiber tubes and the masses of the leg joints and the toes are estimated based on the density of the alluminium alloy 7075-T6. For the application in this chapter, both DC motors acting on the leg joints are the same and their properties correspond to Maxxon's DC motor RE 50 200W, with a reduction ratio of 53:1 and supply voltage 60 V.

Table 4-1. Values of system parameters.

Parameter	Value	Parameter	Value
m [kg]	42.0	b_e [m]	0.05
d [m]	0.30	g [m/s^2]	9.81
a [m]	0.15	μ	0.65
b [m]	1.00	$S_{c,u}$ [MPa]	200.00
I [kg m^2]	3.58	s_f	3.00
ρ [10^2 kg/m^2]	14.66	n_j	53.00
m_{j1} [kg]	0.40	$I_{r,j}$ [10^{-5} kg m^2]	5.42
m_{j2} [kg]	0.40	$\tau_{st,j}$ [N/m]	45.00
m_s [kg]	0.17	$\tau_{ct,j}$ [N/m]	14.96
m_t [kg]	0.10	$\dot{\theta}_{\max,j}$ [rad/s]	11.21

From preliminary runs of the method, the shear stresses on the leg segments were found to be insignificant in comparison with the normal stresses, therefore the shear component of (2-90) is omitted. The alternative inner and outer diameters for which the strength of the tubular segments was tested, were taken from standard carbon fiber tube sizes, see Table 4-2.

Table 4-2. Candidate sizes of carbon fiber tubes.

a/a	d_{in} [mm]	d_{out} [mm]
1	23	25
2	26	28
3	28	30
4	26	30

4.3 The bounding gait

The method was firstly applied for the bounding gait of Figure 4-1. The initial conditions were taken as $x(0)=1\text{m},\ \dot{y}(0)=0\,\text{m/s}$ and $\theta(0)=0\,\text{rad}$. The initial pitch rate of the bounding gait is crucial for the stability of the movement, [72]. In this approach the initial pitching rate $\dot{\theta}(0)$ is found from the maximum pitch angle of the body $\theta_{\max}$,

$$\dot{\theta}(0)=\frac{2\pi}{T}\theta_{\max} \tag{4-1}$$

where the maximum accepted pitch angle of the body was taken as $\theta_{\max}=10\pi/180$ rad.

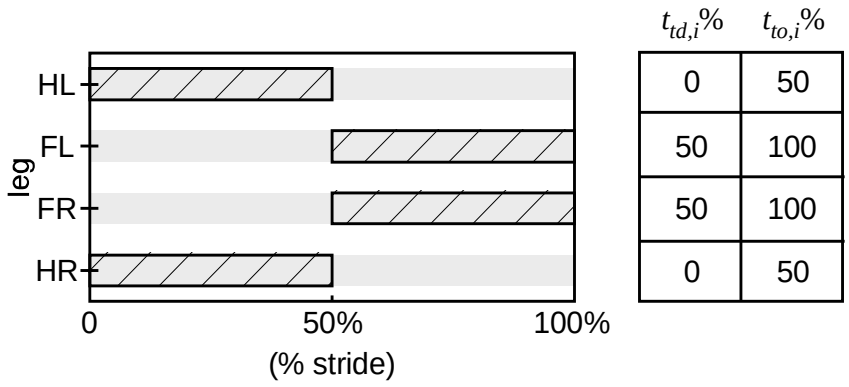


Figure 4-1. Gait graph of the bounding gait and respective time instants of touch down and take off as a percentage of the stride period.

The bounds in which the gait and leg parameters were sought and the respective discretization steps are summarized in Table 4-3. Note that these bounds were set through trial and error to avoid truncating any of the optimal solutions in maximum speed. The reader is reminded that the upper bounds of leg parameters $l_1,\ l_{20}$ are set by introducing the parameter ε, see (3-10).

For bounding with increasing horizontal velocity $\dot{x}(0)$ from 0.20 m/s to 1 m/s, the permissible combinations of gait and leg parameters for which constraints (2-90), (2-100) were valid decreased until convergence, see Figure 4-2. The maximum horizontal velocity reached for the knee backward (KB) configuration was 0.48 m/s, whereas for the knee forward (KF) configuration was 0.54 m/s. The

parameters for which the maximum velocities were reached are shown in Table 4-4. For the optimal parameters for both configurations, the resulting bounding movement is shown in Figure 4-3 and the corresponding to the motion CoM trajectories are shown in Figure 4-4. It is obvious that the movement in the y and θ directions is periodical in one stride and that the robot runs with a steady net horizontal velocity (x direction), as predicted from the form of the horizontal and vertical forces $F_{x,i}$, $F_{y,i}$ and the optimization of the objective function (3-2).

Table 4-3. Bounds and discretization steps of the parametric search for the bounding gait.

Parameter	Lower Bound	Upper Bound	Discr. Step
$\dot{x}(0)$ [m/s]	0.20	1.00	variable
$y(0)$ [m]	0.45	0.85	0.05
a_{fx}	0.00	0.60	0.10
T [s]	0.20	0.60	0.05
k [N/m]	1250	5000	1250
ε	1.00	1.10	0.025
l_1 [m]	0.10	$\varepsilon l_{\max} - l_{20,lb}$	0.01
l_{20} [m]	0.10	$\varepsilon l_{\max} - l_{1,lb}$	0.01

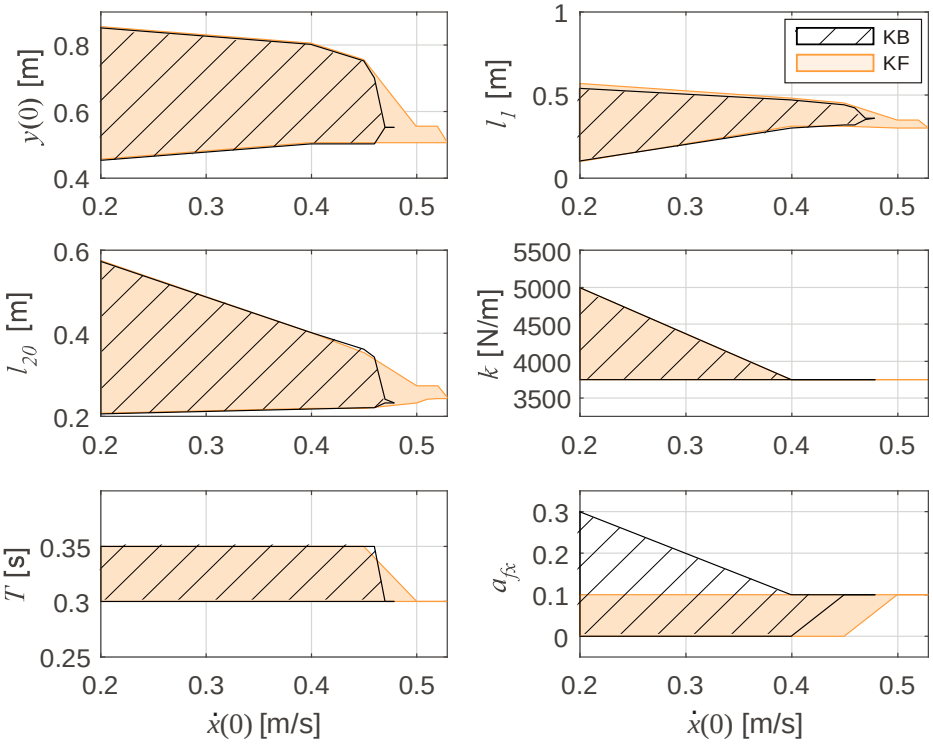


Figure 4-2. Evolution of parameters with horizontal bounding velocity for the knee backward (KB) and the knee forward (KF) configurations.

Table 4-4. Optimal parameters for maximum bounding velocity per configuration.

Config.	$y(0)$ [m]	a_{fx}	T [s]	l_1 [m]	l_{20} [m]	k [N / m]	d_{in} [mm]	d_{out} [mm]
KB	0.55	0.10	0.30	0.36	0.23	3750	26	28
KF	0.50	0.10	0.30	0.30	0.24	3750	26	28

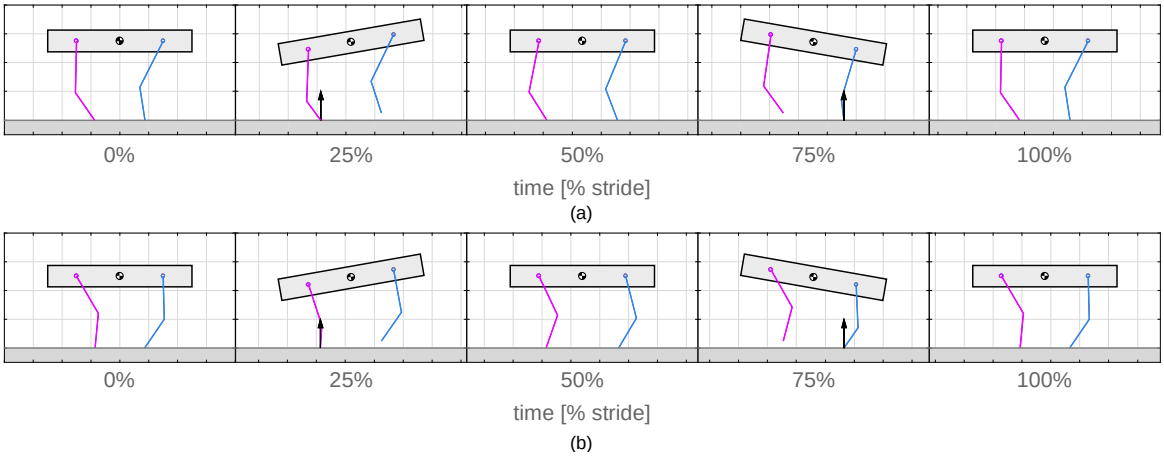


Figure 4-3. Snapshots of the quadruped's bounding motion for the optimal solutions in the (a) KB and the (b) KF configurations.

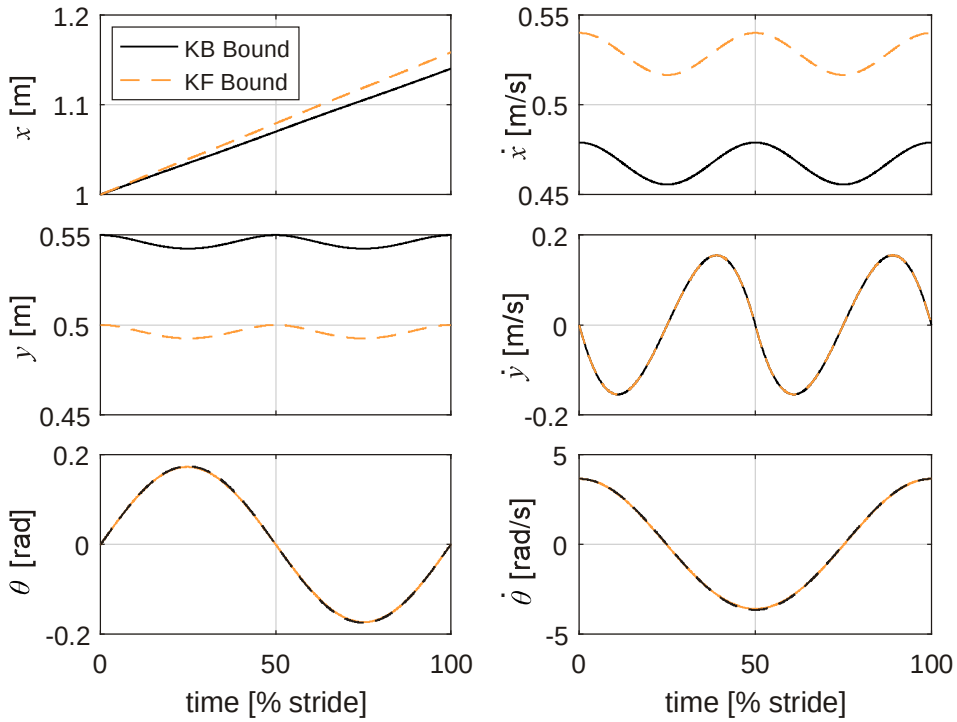


Figure 4-4. Robot body CoM trajectories for bounding with the optimal solutions in the KB and KF configurations.

It can be observed that for the KB and KF configurations, the proximal leg segment of the optimal legs is slightly larger than the distal, and the total length of both segments does not exceed the hip to

hip distance $2d = 0.6\,\mathrm{m}$. For both knee configurations, the reason for which the robot cannot run any faster with the bounding gait is the violation of the max continuous torque constraint of (2-100); in the KB configuration this happens firstly for the torque of the hip joints of the front legs ($\tau_{1,2}$, $\tau_{1,3}$) and for the KF configuration for the torque of the knee joints of the front legs ($\tau_{2,2}$, $\tau_{2,3}$). For the KB configuration in maximum speed (0.48 m/s), $\tau_{1,2}$ is increased mainly during the flight phase ($\mathrm{rms}(\tau_{1,2}) = 14.96\,Nm$), see Figure 4-5 (a). That is probably due to the reduced stride period (0.3 s) in which the max velocity is achieved and the increased output rotor inertia ($\sim \mathrm{n}_j^2 I_{r,j}$) due to the high transmission ratio ($n_j = 53$) of the motor; the leg joint should accelerate and decelerate a significant inertia in a short time. A further increase in the horizontal velocity (see direction of arrows) leads to an overall increase in torque requirement, thus exceeding the maximum permissible continuous torque. For the case of the KF configuration in maximum speed (0.54 m/s), more demanding in torque $\tau_{2,2}$ seems to be the stance phase ($\mathrm{rms}(\tau_{2,2}) = 14.93\,Nm$), see Figure 4-5 (b). This is assumed to happen, due to the infavorable direction of the ground force exerted on the toe of the leg (Figure 4-3 at 75% of the stride). Also in this case, a further increase in horizontal velocity leads to an increase in torque requirement throughout the stride, due to which the max continuous permissible torque constraint is violated.

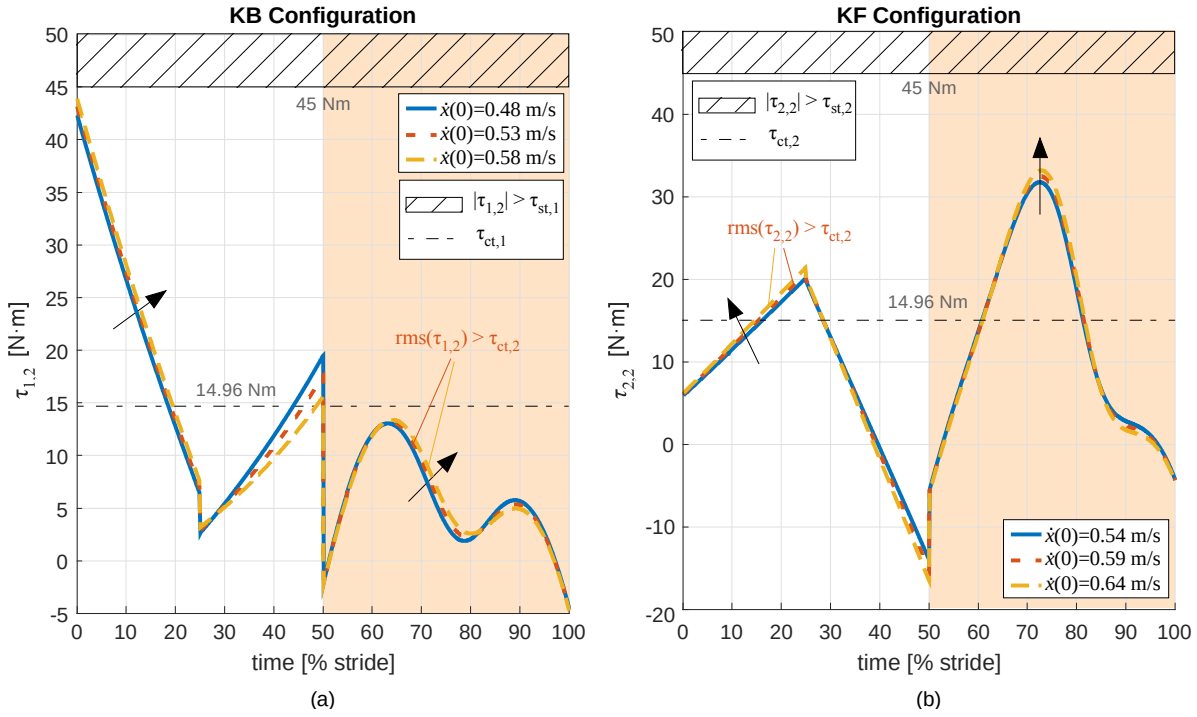
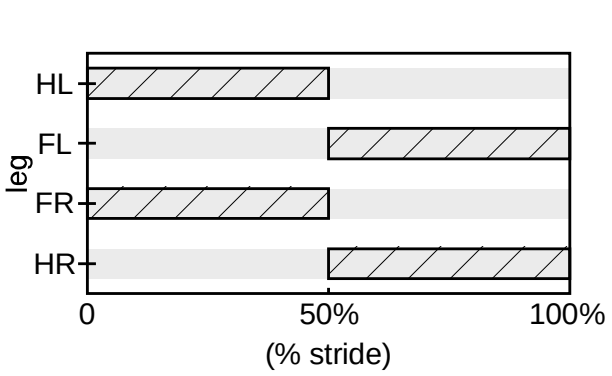


Figure 4-5. Effect of velocity increase on $\tau_{2,1}$ and $\tau_{2,2}$; the colored area indicates the stance phase of each leg.

4.4 The trotting gait

The leg design methodology was also applied for the trotting gait depicted in Figure 4-6. The initial conditions for locomotion with the trotting gait were taken as $x(0) = 1\,\mathrm{m}$, $\dot{y}(0) = 0\,\mathrm{m/s}$, $\theta(0) = 0\,\mathrm{rad}$. In contrast with the bounding gait, the pitching rate for locomotion with trotting remains relatively small [73], therefore the initial pitching rate for trotting was taken as $\dot{\theta}(0) = 0\,\mathrm{rad/s}$.



$t_{td,i}$%	$t_{to,i}$%
0	50
50	100
0	50
50	100

Figure 4-6. Gait graph of the trotting gait and respective touch down and take off time instants as a percentage of the stride period.

The bounds of the parametric search conducted for the trotting gait were set through trial and error as shown in Table 4-5, ensuring that the optimal parameters found were not truncated by them. The discretization step for every parameter was chosen taking into account the magnitude of every parameter and the trade off between a rich parametric space and the duration of solution of the parametric search problem.

Table 4-5. Bounds and discretization steps of the parametric search for the trotting gait.

Parameter	Lower Bound	Upper Bound	Discr. Step
$\dot{x}(0)$ [m/s]	1.00	3.00	variable
$y(0)$ [m]	0.55	1.30	0.05
a_{fx}	0.10	0.70	0.10
T [s]	0.35	0.75	0.05
k [N/m]	1250	7500	1250
ε	1.000	1.100	0.025
l_1 [m]	0.10	$\varepsilon l_{\max} - l_{20,lb}$	0.01
l_{20} [m]	0.10	$\varepsilon l_{\max} - l_{1,lb}$	0.01

The evolution of the permissible gait and leg parameters with increasing horizontal speed is shown in Figure 4-7. The serrated shape of the convergence curves indicates that some solution sets available in greater velocities do not respect some of the actuation constraints in lower velocities, for instance see Figure 4-7 for $\dot{x}(0) = 2.50\,\mathrm{m/s}$ and $\dot{x}(0) = 2.51\mathrm{m/s}$. It was discovered that the actuation constraint that is marginally violated at 2.50 m/s is that of the continuous joint torque, due to a less favorable placement of the leg on the ground during stance phase, see Figure 4-8. Nevertheless, this serrated convergence behavior poses no problem as solution sets than are marginally acceptable in lower velocities, significantly bias actuation constraints in maximum horizontal velocities.

The maximum velocity reached for trotting while respecting constraints (2-90), (2-100), was 2.90 m/s for the knee backward configuration and 2.94 m/s for the knee forward configuration. The optimal parameters for which maximum velocities were reached for both configurations are shown in Table 4-6. The resulting trotting movement of the quadruped robot with the optimal sets of parameters for both

knee configurations is shown in Figure 4-9. In both cases, the robot trots periodically (y, θ directions) with a steady overall horizontal velocity (x direction), see Figure 4-10.

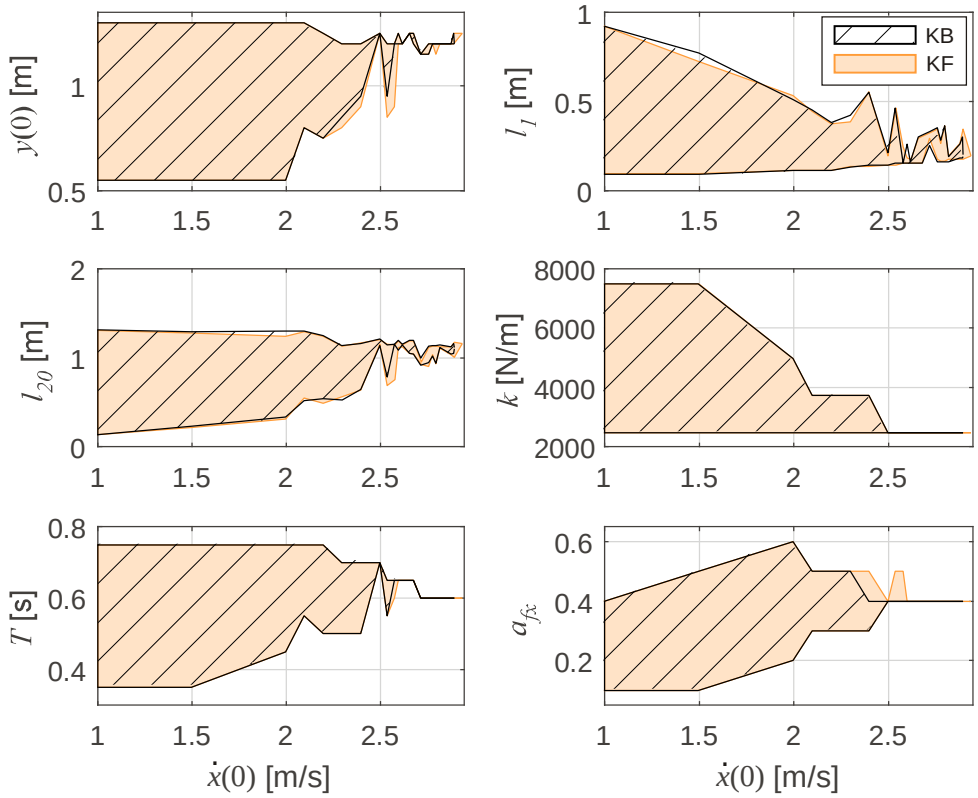


Figure 4-7. Evolution of parameters with horizontal trotting velocity for the knee backward (KB) and the knee forward (KF) configurations.

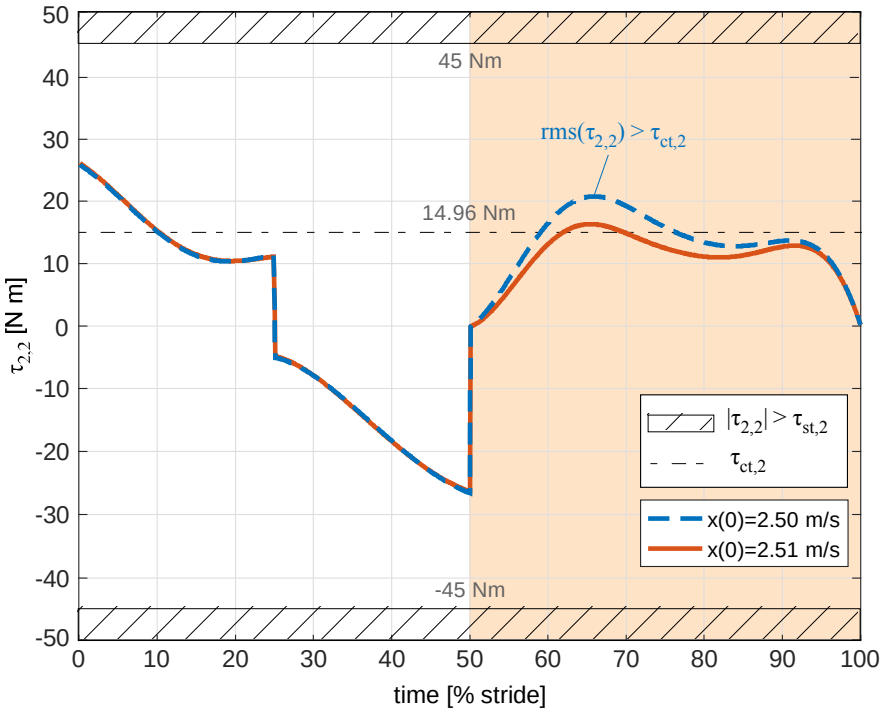


Figure 4-8. Knee torque requirement of a front left leg candidate solution at two successive running velocities.

The great difference between the maximum velocity in the bounding and the trotting gait is probably due to the much greater pitch observed in the former, compare pitch in figures Figure 4-4, Figure 4-10. With increased body pitch, the legs in contact with the ground receive the ground forces in a more crouched position, resulting in greater moment arms and joint torque requirements. Furthermore, the robot moves faster with a trotting rather with a bounding gait, as the passive dynamics of its body facilitate gaits with small pitch ratio. Chatzakos studied in [74] the role of the dimensionless body inertia I^*,

$$I^* = \frac{I_b}{m_b d^2} \tag{4-2}$$

in the pitching rate of movement. He concluded that for $I^* < 1$, the robot body rotates more easily than it translates, facilitating movements with large pitch ratios, as bounding. If on the other hand $I^* \geq 1$, the translational movement of the robot body prevails over rotational, therefore gaits with small pitch ratio, as pronking or trotting, are more favorable. In the case studied here $I^* \simeq 1$, thus to achieve the large pitch ratios necessary for the bounding gait, excessive effort from the legs' motors is required.

Table 4-6. Optimal parameters for maximum trotting velocity per configuration.

Config.	$y(0)$ [m]	a_{fx}	T [s]	l_1 [m]	l_{20} [m]	k [N / m]	d_{in} [mm]	d_{out} [mm]
KB	1.20	0.40	0.60	0.20	1.10	2500	23	25
KF	1.25	0.40	0.60	0.20	1.16	2500	23	25

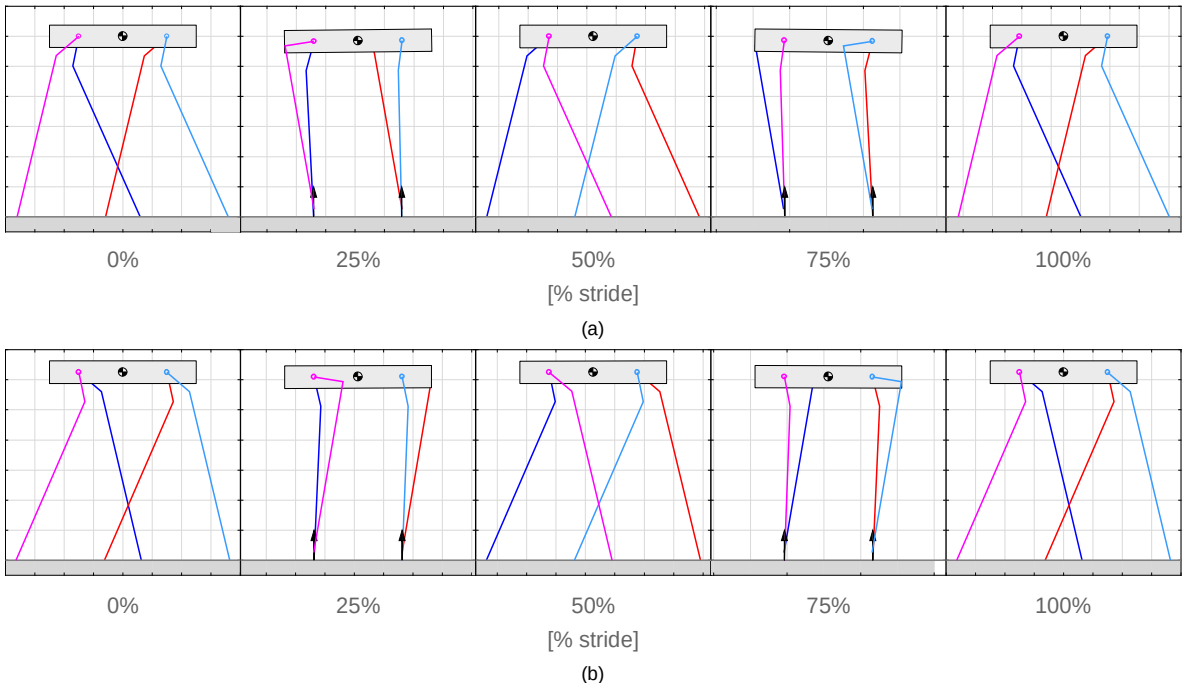


Figure 4-9. Snapshots of the quadruped's trotting motion for the optimal solutions in the (a) KB and the (b) KF configuration.

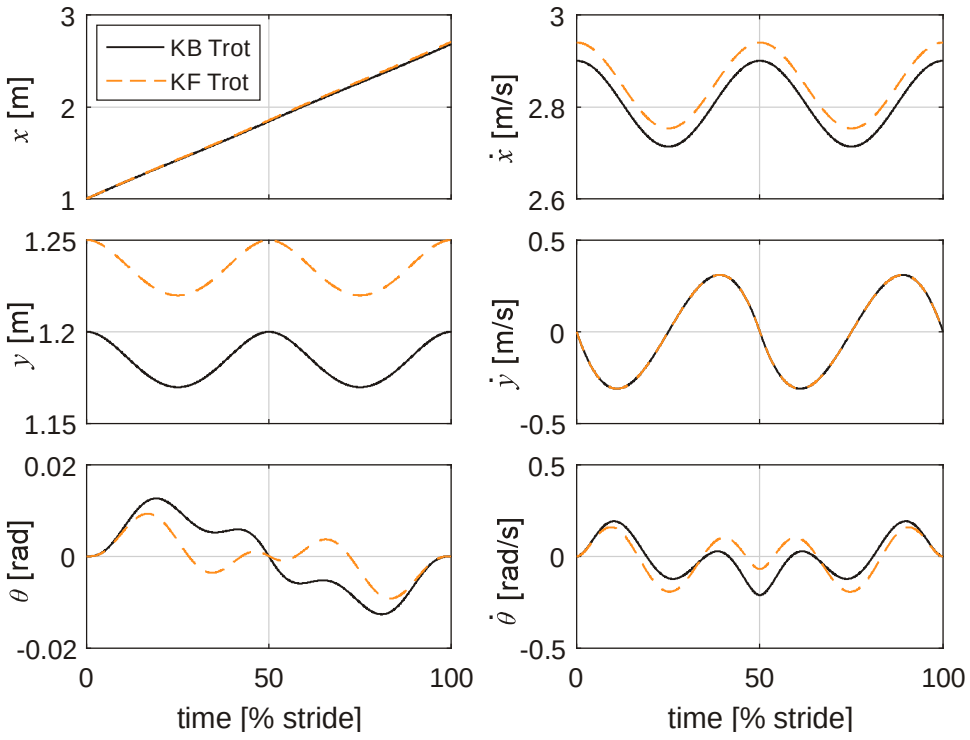


Figure 4-10. Robot body CoM trajectories for trotting with the optimal solutions in the KB and KF configurations.

The optimal leg solutions for both configurations have a really short proximal leg segment and a long distal leg segment with a soft spring, so that the total leg length is larger than twice the hip to hip distance, $2 \cdot 2d = 1.2\,m$. With these legs, during stance most of the leg deflection is undertaken by the passive element, allowing the robot to withstand ground forces in a near singular configuration. Having long legs, the quadruped can cover a larger distance in one stride, even for a large stride period. The short proximal leg segment keeps the heavy knee joint close to the robot body, thus reducing the total inertia of an overall long leg. Furthermore, having a small proximal segment l_1 and a long distal leg segment l_{20}, $l_1 \ll l_{20}$, the hip motor actuates on the total leg length $l_1 + l_{20} \cong l_{20}$ and the knee motor acts on the segment with length l_{20}. As a result, the acceleration/deceleration of approximately the total leg is equally distributed to both motors. Nevertheless, the proximal leg segment can't get too small, or else it will be impossible for the leg's toe to create the desired clearance from the ground.

In any knee configuration, if the robot tries to trot in an even greater horizontal velocity, the max continuous torque constraint is violated. For the case of the KB configuration, this firstly happens as velocity increases at the hip joint of the HL leg (torque $\tau_{1,1}$). At maximum horizontal velocity (2.90 m/s), the hip joint torque requirement is already demanding ($\mathrm{rms}(\tau_{1,1}) = 14.95\,Nm$), mainly due to the task of swinging a long leg, see Figure 4-11 (a). By further increasing the forward velocity, the overall torque $rms(\tau_{1,1})$ exceeds the permissible continuous torque value. For the KF configuration, the max permissible continuous torque constraint is firstly violated at the knee joint of the hind left leg (torque $\tau_{2,1}$). At maximum horizontal velocity, the knee joint torque requires much torque to support the long distal segment l_{20} during the stance phase and to swing it during the flight phase ($\mathrm{rms}(\tau_{2,1}) = 14.92\,Nm$

), see Figure 4-11 (b). With increasing horizontal velocity, a torque increase mainly throughout stance phase leads to a non permissible $\mathrm{rms}(\tau_{2,1})$.

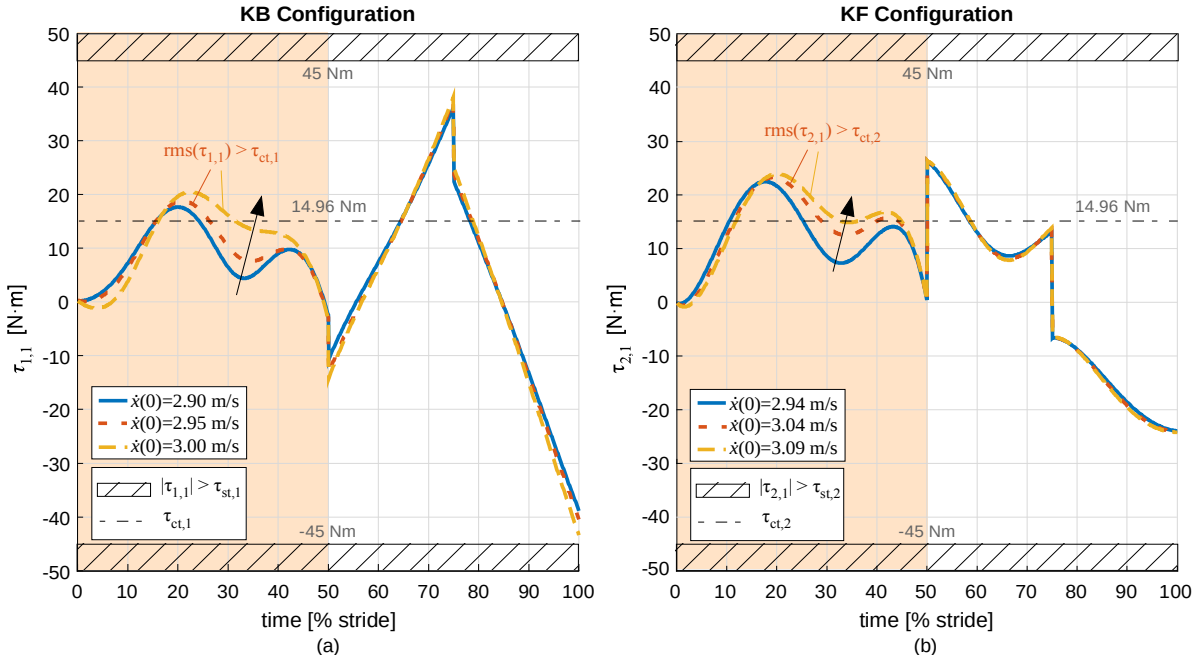


Figure 4-11. Effect of velocity increase on torques $\tau_{1,1}$ and $\tau_{2,1}$.

4.5 Comparison with the popular equally segmented leg

So far, the optimal parameters that achieve the overall maximum horizontal velocity are these of the trotting gait in a KF configuration, see Table 4-6. The legs to achieve this velocity are long ($l_1 + l_{20} \approx 4d$), with a short proximal leg segment, a long distal leg segment ($l_1 \ll l_{20}$) and a soft spring. This leg structure is much different that the current most popular two-segment leg solution used by the current version of *NTUA Laelaps*, the *StarlETH*, the *HyQ* and many of *Boston Dynamics*' robots; an equally segmented leg ($l_1 = l_{20}$), with its total leg length being approximately equal to the hip to hip distance ($l_1 + l_{20} \approx 2d$). In this Section, these two leg structures are evaluated using literature elements and results of the proposed method.

There are some advantages of having an equally two-segment leg with a total length of $l_1 + l_{20} \approx 2d$. Firstly, by equal segmentation ($l_1 = l_{20}$) the workspace of the leg is maximized. By substituting in (2-78) $l_1 = l_{2,i} = l$ the workspace is expanded from an annular to a circular area with radius $2l$, thus maximizing the workspace. This property is usefull for robots targeted to perform stunts like moving with crouched legs under obstacles or evaluating alternative control schemes, where the clearance of the leg is not predetermined. Secondly, for an equally segmented two-segment leg the manipulability ellipsoid is maximized, [75]. The manipulability ellipsoid is a Jacobian based measure of manibulability, traditionally used for design and motion planning of manipulators. It shows the area in which the end effector (toe) can move with specific velocities or exert specific forces, with unit leg joint angular velocities or torques respectively. Furthermore, having a total leg length approximately equal to the hip to hip distance instead of a much larger leg improves stability against perturbations; given a

force perturbation on its body, it is much easier for a top heavy robot to collapse than for one with short legs.

Although the equally segmented leg poses some advantages that justify its use by many research teams, the proposed method and results focus on the ability of a quadruped robot to run fast given certain actuation constraints; to achieve this, it may be required for the workspace or the manipulability of the leg to be reduced. To prove this, the set of parameters most close to the popular equally segmented leg that could achieve maximum velocity while respecting constraints (2-90), (2-100) is brought forward, see Table 4-7.

Table 4-7. Set of parameters maximizing the performance of the equally segmented leg.

Config.	$y(0)$ [m]	a_{fx}	T [s]	l_1 [m]	l_{20} [m]	k [N / m]	d_{in} [mm]	d_{out} [mm]
KF	0.55	0.60	0.60	0.32	0.33	2500	23	25

The maximum velocity that can be achieved with this leg by trotting in the KF configuration is $\dot{x}(0) = 2.03\,\mathrm{m/s}$, approximately $0.9\,\mathrm{m/s}$ less than that achieved with the optimal KF leg solution found in Section 4.4. This probably happens for two reasons. Firstly, the torque requirement is more equally distributed to the hip and the knee joint motors of the long inequally segmented leg (LISL), than to the short equally segmented leg (SESL). As explained in Section 4.4, if $l_1 \ll l_{20}$ then both motors cooperate to accelerate and decelerate the total leg. If $l_1 \approx l_{20}$ on the other hand, the hip motor acts on the whole leg length $l_1 + l_{20}$, while the knee motor acts on the distal leg segment l_{20}. As a result, it is expected the hip joint motor of the SESL leg to be more loaded than the knee joint motor. Indeed, at maximum velocity $\dot{x}(0) = 2.03\,\mathrm{m/s}$, the rms torque value of the HL SESL leg hip joint motor in one stride was found to be larger $(\mathrm{rms}(\tau_{1,1}) = 14.90\,\mathrm{Nm})$ than the rms torque value of the knee joint $(\mathrm{rms}(\tau_{2,1}) = 12.92\,\mathrm{Nm})$ at the same time. As expected, the rms torque values were more equally distributed to the leg joints of LISL, i.e. $\mathrm{rms}(\tau_{1,1}) = 14.53\,\mathrm{Nm}$ for the hip and $\mathrm{rms}(\tau_{2,1}) = 14.92\,\mathrm{Nm}$ for the knee joint at maximum velocity $\dot{x}(0) = 2.94\,\mathrm{m/s}$. Secondly, longer legs result in longer stride length λ. As a result, for the same period T, the mean velocity $\bar{v} = \lambda / T$ is expected to be larger. We assume that for both leg solutions LISL and SESL, the robot ran with the same period $(\mathrm{T} = 0.6\,\mathrm{s})$ and the same leg stiffness $(k = 2500\,\mathrm{N/m})$ to exploit passive dynamics throughout stance phase. Indeed, for two legs in contact with the ground in each stance phase the quadruped robot can be simplified in a system of a mass $m = 42\,\mathrm{kg}$ and two springs in parallel $k_t = 2k = 5000\,\mathrm{N/m}$. Then the free oscillation period of the system is found as,

$$T_n = 2\pi\sqrt{\frac{m}{k_t}} = 0.58\,\mathrm{s} \tag{4-3}$$

which is really close to the value of the stride period, $T = 0.6\,\mathrm{s}$.

4.6 Results in a nutshell

In this chapter, the leg design methodology was implemented for a robot with the body inertial characteristics of *NTUA Laelaps*, two identical DC motors acting on the hip and the knee joints of each

leg, for legs either in the KB or in the KF configuration, both for the bounding and the trotting gaits. The results showed that the quadruped robot could run faster with the trotting gait, with long legs that have a much shorter proximal leg segment in comparison with the distal leg segment. Furthermore it was shown that with this long leg the robot could move faster by trotting in comparison with the popular two-segment leg that has a total length approximately equal to the hip separation and segments of equal length. In all cases, the reason the robot could not move any faster was the violation of the maximum continuous torque constraint. To facilitate comparison, the results for each case are summarized in Table 4-8.

Table 4-8. Optimal leg attributes and max rms joint torque requirement for achieving the maximum velocity in every examined case.

Gait	Leg Config.	l_1 [m]	l_{20} [m]	$\frac{(l_1+l_{20})}{2d}$	k [N/m]	$\max_{j,i}(\mathrm{rms}(\tau_{j,i}))$ [Nm]	$\max(\dot{x}(0))$ [m/s]
Bound	KB	0.36	0.23	0.98	3750	14.96	0.48
Bound	KF	0.30	0.24	0.90	3750	14.93	0.54
Trot	KB	0.2	1.10	2.17	2500	14.95	2.90
Trot	KF	0.2	1.16	2.27	2500	14.92	**2.94**
Trot	KF	0.32	0.33	1.08	2500	14.90	2.03

4.7 Validation of the optimal leg

The leg design method and its results presented in the previous sections are based upon certain assumptions, concerning the profile of the ground forces and the extent to which the CoM of the system is affected by the movement of the legs. The optimal legs were found for ideal passive elements with no damping and for steady state running without net acceleration in each stride. The long size of the optimal leg for the trotting gait, results in a top heavy design that causes stability concerns. To this end, an independent to the proposed method validation strategy is considered necessary for the validation of the results. The proposed validation strategy consists of more realistic physical descriptions for the quadruped robot and the ground, and a control scheme for quadrupedal trotting. The leg evaluated is the optimal leg for trotting in the KF configuration, for which the overall maximum running velocity of $2.94\,\mathrm{m/s}$ is achieved.

The physical description of the quadruped robot was modified to also include damping b in the prismatic compliant elements. The interaction of the leg with the ground was modeled as a point contact. Each time the toe of a leg collides with the ground a normal and a frictional contact force are exerted to the toe of the leg. The normal component of the ground force F_n is described by the continuous, non - sticking Hunt - Crossley impact model [76], [77],

$$F_n = k_g \delta_y^{\nu} + b_g \delta_y^{\nu} \dot{\delta}_y^{\nu} \tag{4-4}$$

where δ_y and $\dot{\delta}_y$ are the local identation of the ground and its rate respectively, $\nu = 1.5$ is the Hertzian coefficient for non-adhesive contact, k_g is the ground stiffness coefficient depending on the materials in contact and b_g is the ground damping coefficient calculated as a function of stiffness,

$$b_g = 1.5 c_a k_g \tag{4-5}$$

The frictional force component consists of a static friction term and an exponential term describing the friction - velocity curve when slippage occurs [78],

$$F_t = \begin{cases} -\text{sgn}(\dot{x}_t)\mu F_n, & |\dot{x}_t| \le u_e \\ -\text{sgn}(\dot{x}_t)F_n(\mu_c + (\mu - \mu_c)\text{e}^{-|\dot{x}_t/u_s|}), & |\dot{x}_t| > u_e \end{cases} \tag{4-6}$$

where μ_c is the kinetic friction coefficient, μ is the static friction coefficient, $\dot{x}_t$ is the tangential to the toe velocity, u_s is the Stribeck velocity, F_n the normal force of the Hunt - Crossley impact model and u_e is the toe velocity threshold for which slippage occurs. Trotting was accomplished by applying to the system a P-V control scheme with an elliptical reference toe trajectory, [71].

The trotting experiment was conducted on stiff ground with $k_g = 400{,}000\,\text{N/m}$, $b_g = 120{,}000\,\text{Ns/m}$, [79]. The spring damping coefficient was considered equal to $b_l = 0.3\,\text{Ns/m}$. The kinematic friction coefficient was set to $\mu_c = 0.8$, the static coefficient to $\mu = 0.65$, the threshold velocity to $u_e = 10^{-3}\,\text{m/s}$ and the Stribeck velocity to $u_s = 10^{-2}\,\text{m/s}$. By properly tuning the gains of the controller k_p, k_v to initial values and gradually increasing k_p, a stable accelerating movement was achieved with the optimal leg in the KF configuration $(l_1 = 0.2\,\text{m},\ l_{20} = 1.16\,\text{m},\ k = 2500\,\text{N/m})$, see Figure 4-12.

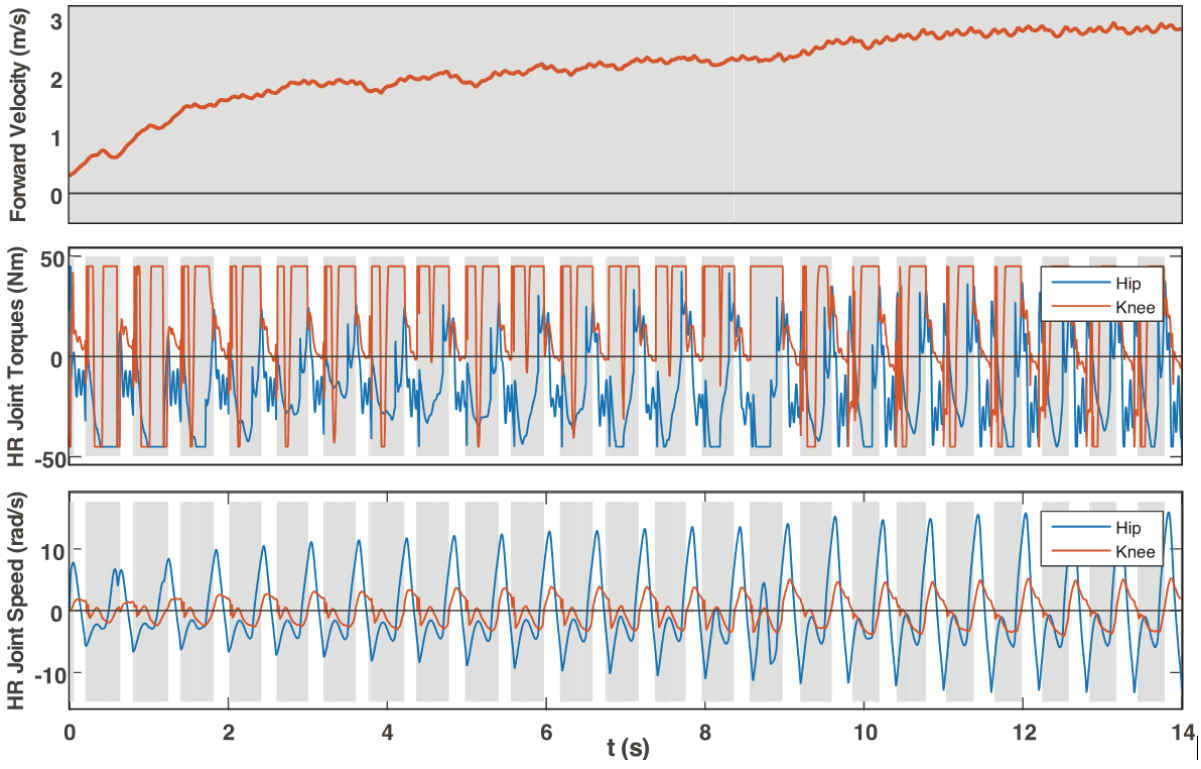


Figure 4-12. Top to bottom: Evolution of robot forward velocity, HR leg joint torques and angular velocities with time for the validation experiment of the optimal leg; the gray backround denotes stance phases.

The optimal leg solution performed well in this realistic trotting experiment, as the robot succeeded in running stably (without collapsing), while respecting the max short term torque constraint $(\tau_{st,j} = 45\,\text{N/m},\ j = 1,2)$ for any joint motor of all legs for the total duration of the experiment. The maximum angular velocity constraint $(\dot{\theta}_{\max,j} = 11.21\,\text{rad/s},\ j = 1,2)$ was also respected for the whole

duration of the experiment for the knee joint motor and up to a forward velocity of approximately $2\,\mathrm{m/s}$ for the hip joint. Depending on the values of (k_p, k_v) the toe in flight phase performed an ellipsoid like movement with clearance larger than that in the application of Section 4.4 $(b_e = 0.05\mathrm{m})$, thus probably leading to an increase in hip joint angular velocity requirement for lower forward velocity. Finally, due to the incurred losses in the leg springs and the ground and the accelerating movement of the robot, the *rms* torque requirement was increased for most motors.

5 Case studies aimed to improve the running performance of *NTUA Laelaps*

5.1 Introduction

In the previous chapter general results where presented concerning the optimal gait and leg configuration and the respective gait and leg parameters for which maximum locomotion velocity could be reached. For convenience two identical motors were considered to act on the actuated joints of each leg. The overall maximum horizontal velocity was reached for the trotting gait in the KF configuration. In this chapter, using this gait and knee configuration, we aim to find out the optimal leg and gait parameters for maximum locomotion velocity, focusing on the existing actuation system of the *NTUA Laelaps* quadruped. This consists of two different DC motors, a brushed and a brushless, each introducing its own actuation constraints, see Table 5-1. By investigating alternative scenarios in which the brushed and the brushless motors act on the hip and the knee joints alternately, the motor - leg joint correspondence is figured out for maximizing the quadruped robot's running performance. As the *NTUA Laelaps* is not only intended for sagittal plane running (e.g. on a treadmill, where the rolling motion is fixed), a stability criterion is introduced for movement in the frontal plane and optimal results respecting this criterion are presented. Finally, the role of a supplementary reduction ratio is studied in the running performance of the *NTUA Laelaps*.

Table 5-1. Properties of the available *NTUA Laelaps* motors.

Type	Rotor inertia [kg m^2]	Reduction ratio	Max cont. torque [N m]	Max short term torque [N m]	Max angular velocity [rad/s]
RE 50 200 W (Brushed)	$542 \cdot 10^{-7}$	53:1	14.96	45	11.21
RE 45 250 W (Brushless)	$209 \cdot 10^{-7}$	43:1	10.6	28	14.66

5.2 Brushed motor acting on the hip joint

In the first case study, the brushed motor is considered acting on the hip joint and the brushless motor on the knee joint. This scenario is introduced to the physical system description, by tuning properly the corresponding actuation related parameters, see Table 5-2. The rest of the model parameters and the initial conditions for the simulations conducted in this section remain the same as in Section 4.4.

Table 5-2. Actuation parameters for the motors acting on the hip and knee joints.

j	$I_{r,j}$ [kg m^2]	n_j	$\tau_{\mathrm{ct,j}}$ [N m]	$\tau_{\mathrm{st,j}}$ [N m]	$\dot{\theta}_{\mathrm{max,j}}$ [rad/s]
1	$542 \cdot 10^{-7}$	53	14.96	45	11.21
2	$209 \cdot 10^{-7}$	43	10.6	28	14.66

The boundaries of the parametric search were also in this case properly tuned in order to avoid truncation of an optimal set of solutions. Apart from the boundaries displayed in Table 5-3, the rest were kept the same as those presented in Table 4-5.

Table 5-3. Tuned boundaries for the parametric search of Section 5.2.

Parameter	Lower Bound	Upper Bound	Discr. Step
$\dot{x}(0)$ [m/s]	2.00	2.80	variable
$y(0)$ [m]	0.8	1.30	0.05
k [N/m]	1250	5000	1250

The maximum velocity for which constraints (2-90), (2-100) were respected is 2.585 m/s for the leg and gait parameters shown in Table 5-4. In maximum velocity, the angular velocity of the hip joints is considerably increased, reaching 95.46 % of the permissible at the hip joint of the FR leg. The *rms* torque of the knee joints is also close to the permissible continuous torque, with a value of 98.63 % of the permissible observed in the knee joint of the FL leg. If the horizontal velocity is further increased, the maximum continuous torque constraint is violated for the knee joint of the FL leg, due to a torque requirement increase oberverd mainly during the stance phase of the leg, see Figure 5-1.

Table 5-4. Set of optimal parameters for maximum velocity trotting for the brushed motor acting on the hip joint.

Config.	$y(0)$ [m]	a_{fx}	T [s]	l_1 [m]	l_{20} [m]	k [N / m]	d_{in} [mm]	d_{out} [mm]
KF	1.15	0.40	0.65	0.18	1.07	2500	23	25

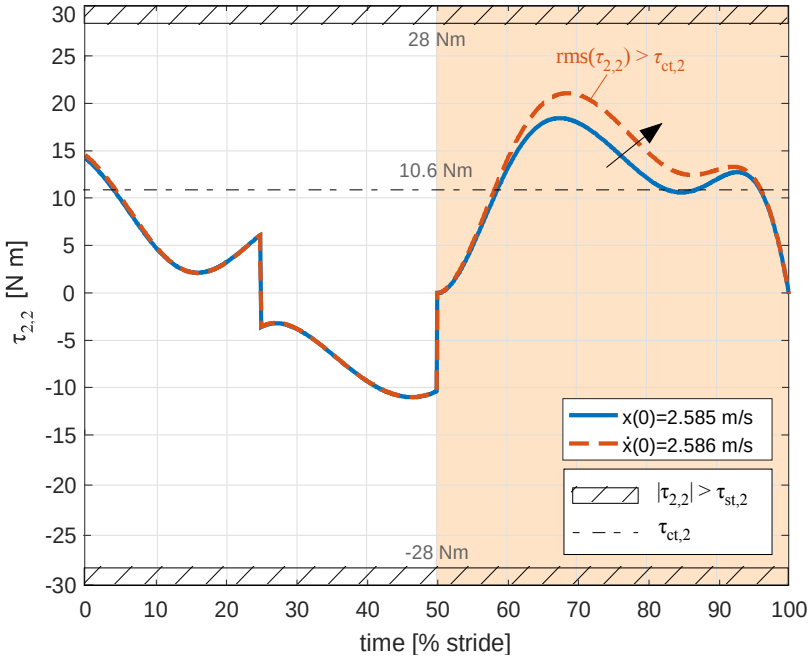


Figure 5-1. Effect of velocity increase on hip joint torque $\tau_{2,2}$.

The observed increased requirement of the hip joints for angular velocity and the knee joints for *rms* torque for legs in the KF configuration is consistent with the results presented in Sections 4.3 - 4.4; in these cases also, max velocity running in the KF configuration required higher *rms* torque on the knee joints. This observation comes in contradiction with the brushed motor actuating the hip joint. The brushed motor provides higher continuous and short term torque than the brushless, and lower angular velocity, see Table 5-1. Consequently, there is a strong indication that by alterating the position of the brushed and the brushless motors, the robot may achieve an even greater horizontal velocity. This assumption is checked in the following section.

5.3 Brushed motor acting on the knee joint

The brushed motor in this case is considered acting on the knee joint and the brussless motor on the hip joint. This is introduced in the description of the physical system by changing the actuation parameters accordingly, see Table 5-5. The parametric set boundaries that were tuned differently in comparison with Table 4-5 are presented in Table 5-6.

Table 5-5. Actuation parameters for the motors acting on the hip and knee joints.

j	$I_{r,j}$ [kg m^2]	n_j	$\tau_{\mathrm{ct,j}}$ [N m]	$\tau_{\mathrm{st,j}}$ [N m]	$\dot{\theta}_{\mathrm{max,j}}$ [rad/s]
1	$209 \cdot 10^{-7}$	43	10.6	28	14.66
2	$542 \cdot 10^{-7}$	53	14.96	45	11.21

Table 5-6. Tuned boundaries for the parametric search of Section 5.3.

Parameter	Lower Bound	Upper Bound	Discr. Step
$\dot{x}(0)$ [m/s]	2.00	3.60	variable
$y(0)$ [m]	0.90	1.40	0.05
T [s]	0.45	0.90	0.05
k [N/m]	1250	5000	1250
ε	1.000	1.125	0.025

The maximum horizontal velocity achieved is 3.039 m/s, greater than the one achieved in the case in which the brushed motor was acting on the hip joint, as expected from the discussion in the previous section. The set of optimal leg and gait parameters for which the maximum velocity is reached is shown in Table 5-7. In maximum trotting velocity, the requirement for hip joint angular velocity is significantly increased, reaching a fraction of 98.02 % the maximum permissible angular velocity $\dot{\theta}_{\mathrm{max},2}$ for the hip joint of the FR leg. The requirement in knee joint *rms* torque is even more considerable, being at 99.79% of the maximum continuous permissible torque $\tau_{ct,2}$ for the knee joint of the FL leg. If the robot runs at a larger horizontal velocity, the maximum continuous torque constraint is violated for the motor actuating on the FL knee joint, due to an increase in torque requirement during stance phase, see Figure 5-2.

Table 5-7. Set of optimal parameters for maximum velocity trotting for the brushed motor acting on the hip joint.

Config.	$y(0)$ [m]	a_{fx}	T [s]	l_1 [m]	l_{20} [m]	k [N / m]	d_{in} [mm]	d_{out} [mm]
KF	1.25	0.40	0.6	0.14	1.21	2500	23	25

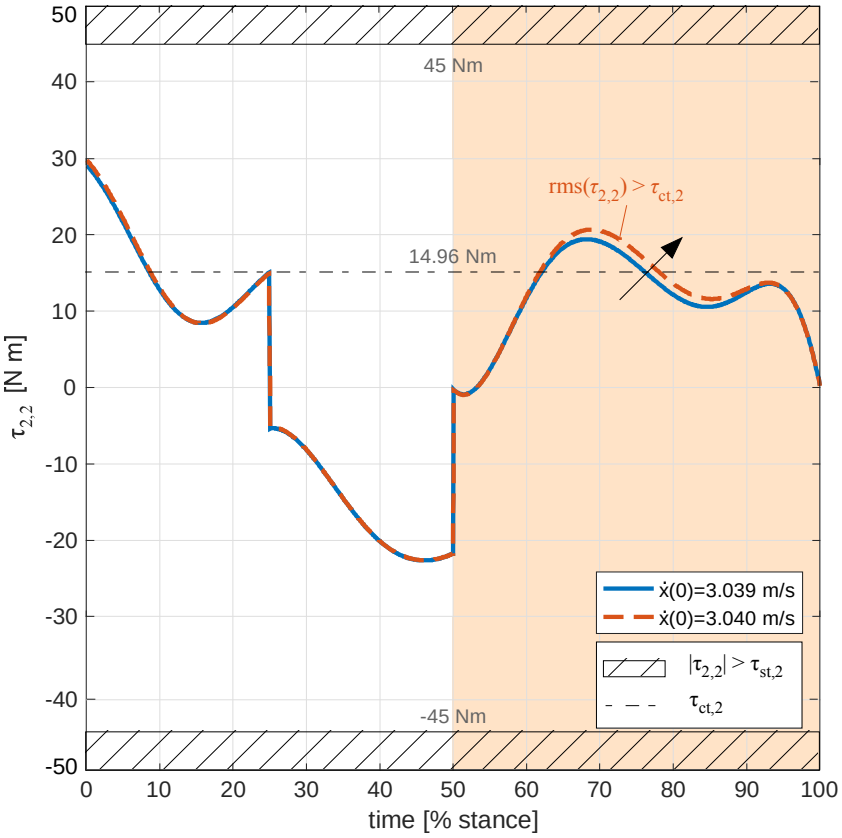
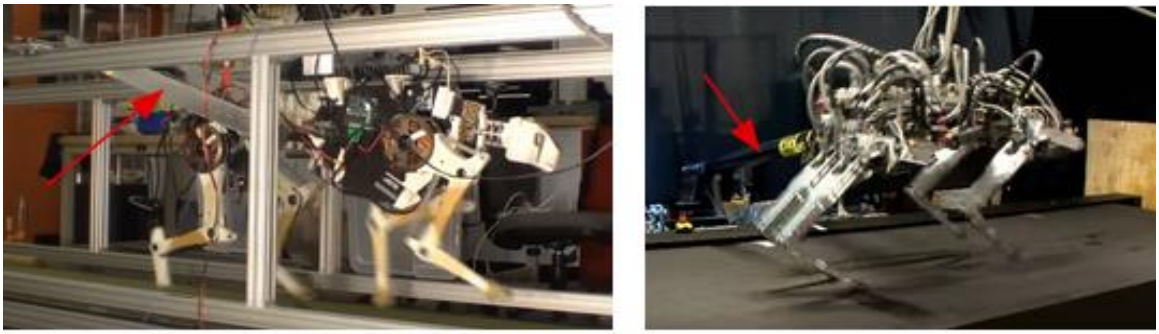


Figure 5-2. Effect of velocity increase on hip joint torque $\tau_{2,2}$.

5.4 Extending stability in the frontal plane

All the optimal solutions for performance trotting presented so far consist of long legs $(l_1 + l_{20} > 4d)$. Large horizontal velocities can be achieved with long legs, as with these large stride length is achieved. Although long legs introduce top heaviness to the system of the quadruped robot, in Section 4.7 it was shown that with an appropriate control scheme, dynamic stability can be achieved even for top heavy robots, for sagittal plane trotting (planar movement). This ensures that the quadruped robot will be able to achieve performance trotting, given that the rolling motion of its body is insignificant, as for instance in the case where the robot runs on a treadmill with a mechanism fixing its rolling motion (e.g. a boom or a x-y-θ stage). In the literature, the fastest running performances of quadruped robots have been achieved for in - lab treadmill running, see Figure 5-3. Nevertheless, there are many examples of robots that have succeeded in no roll treadmill running but have failed in out of laboratory running (an example of the FastRunner robot is shown in [80]); this is probably due to frontal plane instability. Top heavy robot designs amplify frontal plane instability, as small perturbating forces on the robot's body result in large tipover moments due to the high position of the CoM. In the following analysis, the relationship

between the robot's CoM height and the tipover stability is studied in the frontal plane, utilizing standard tipover stability principles introduced in [81]. By assuming a perturbation force and for the geometry of the *NTUA Laelaps*, the maximum permissible height of running is obtained. Introducing this constraint in the method, the optimal parameters are found, for which the robot runs in maximum velocity in the sagittal plane, while on the same time tipover stability is ensured in the frontal plane.



(a) (b)

Figure 5-3. Snapshots of the (a) *MIT Cheetah* and (b) *Boston Dynamics*' Cheetah robots high speed running on treadmills (videos available in [82], [83]) ; the red arrows indicate support mechanisms that fix rolling motion.

For the quadruped robot depicted in Figure 5-4 (a), let F_{tot} be the total force exerted on the CoM of the robot, (AC) be the line segment connecting the toe of the left leg with the CoM of the robot, γ_1 be the angle between the total force vector and (AC) and d_1 be the moment arm between the total force and the toe - ground contact point A. Then, as long as $\gamma_1 \geq 0$ stability from tipover is ensured. If on the other hand $\gamma_1 < 0$, the moment of the force F_{tot} with respect to the contact point A tends to tip the robot in the counterclockwise direction. We assume that the total force F_{tot}, consists of a horizontal perturbation force component F_p and the robot's weight component F_w, see Figure 5-4 (b). We also assume that a local perturbation in the terrain's geometry creates a small slope corresponding to γ_ε. Let y be the distance of the robot's CoM from the ground and w be the half hip separation in the frontal plane. Then from Figure 5-4 (b),

$$\tan(\gamma_1 + \gamma_2 + \gamma_\varepsilon) = \frac{w}{y} \tag{5-1}$$

$$\tan(\gamma_2) = \frac{F_p}{F_w} \tag{5-2}$$

Then solving (5-1), (5-2) to γ_1,

$$\gamma_1 = \tan^{-1}(\frac{w}{y}) - \gamma_2 - \gamma_\varepsilon, \qquad \gamma_2 = \tan^{-1}(\frac{F_p}{F_w}) \tag{5-3}$$

By demanding that $\gamma_1 \geq 0$ then from (5-3),

$$\tan^{-1}(\frac{w}{y}) \geq \tan^{-1}(\frac{F_p}{F_w}) + \gamma_\varepsilon \tag{5-4}$$

Solving Equation (5-4) to y,

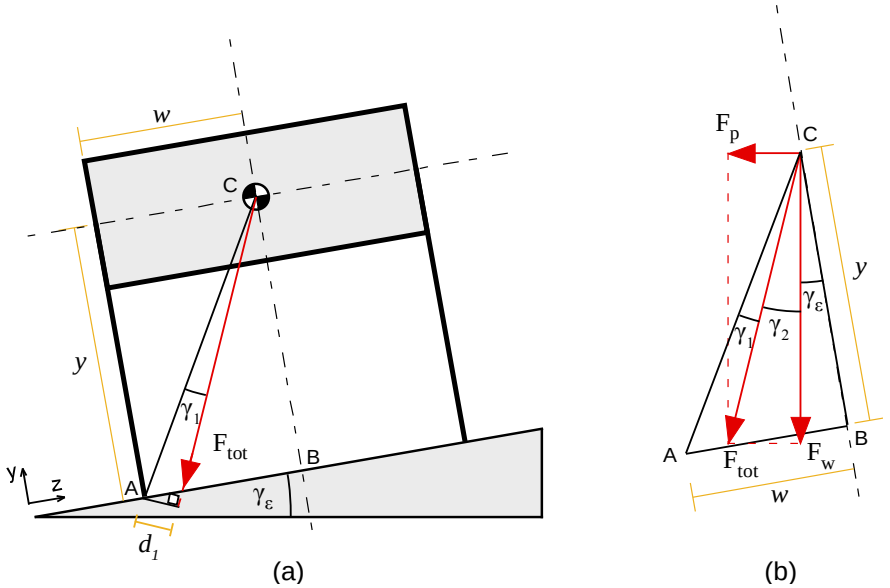


Figure 5-4. (a) Total force applied on the CoM of the quadruped robot in the frontal plane and (b) Correlation of the robot's geometry with the forces exerted on its CoM.

$$y \leq w / \tan(\tan^{-1}(\frac{F_p}{F_w}) + \gamma_\varepsilon) \tag{5-5}$$

Equation (5-5) states that to achieve tipover stability in the frontal plane, the CoM height of the robot y should be upperly bound, depending on the frontal plane half hip separation distance w of the robot, the robot's weight F_w, the perturbation force F_p and the slope of the terrain γ_ε.

Assuming a perturbation force of 10 % the robot's body weight and a local terrain slope of 5^o, for the frontal half hip separation distance of *NTUA Laelaps* $w = 0.15\text{m}$, the permissible CoM heights in which the robot should run to avoid tipover instabilities should be less or equal than 0.8 m. For the actuation system of Table 5-5 and tuning properly the upper and lower bounds of the parametric search (Table 5-8), optimal parameters are sought for trotting in maximum velocity, for a maximum running height of 0.8 m. The maximum velocity that can be achieved by trotting in this state is 2.817 m/s for the parameters shown in Table 5-9. The resulting leg morphology and motion of the robot in one stride is shown in Figure 5-5. Although the maxiumum velocity achieved is 0.22 m/s (~7%) less than the velocity achieved with the leg of Section 5.3, with the much shorter leg (40 % shorter) tipover stability is ensured in the frontal plane.

Table 5-8. Upper and lower bounds of parametric search of Section 5.4.

Parameter	Lower Bound	Upper Bound	Discr. Step
$\dot{x}(0)$ [m/s]	2.00	2.90	variable
$y(0)$ [m]	0.5	0.8	0.05
T [s]	0.35	0.85	0.05
k [N/m]	1250	5000	1250
ε	1.000	1.125	0.025

Table 5-9. Optimall parameters for maximum velocity trotting, for $y(0) \leq 0.8\,\mathrm{m}$.

Config.	$y(0)$ [m]	a_{fx}	T [s]	l_1 [m]	l_{20} [m]	k [N / m]	d_{in} [mm]	d_{out} [mm]
KF	0.7	0.60	0.55	0.20	0.61	2500	23	25

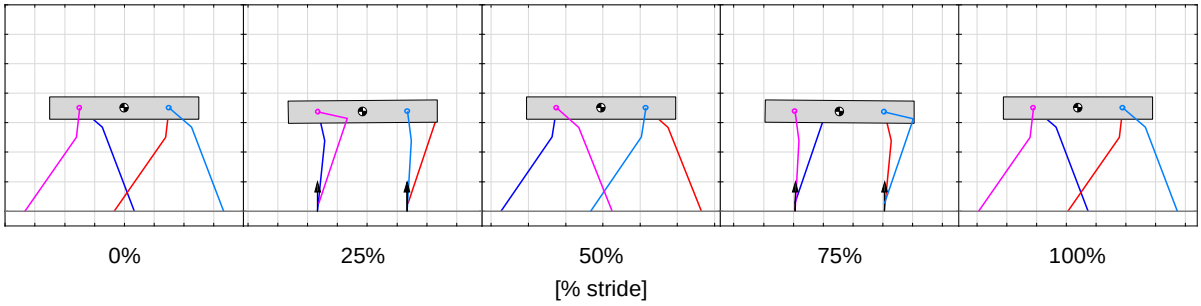


Figure 5-5. Snapshots of the trotting quadruped robot with the optimal legs for tipover stability in the frontal plane.

In maximum velocity, the maximum *rms* torque requirement of the hip joints is at 93.07 % of the maximum permissible continuous hip torque and the maximum *rms* torque of the knee joints is at 98.52 % of the maximum permissible continuous knee torque. The requirement in hip joint angular velocity is also increased, with a maximum observed reaching 99.40 % of the maximum permissible. A further increase in horizontal velocity cannot be achieved with the optimal leg solution, as the required increase in stride length results in footfalls out of the leg's workspace.

5.5 Introducing a supplementary reduction ratio

For many of the evaluated sets of solutions of the previous sections, a higher value of horizontal velocity could not be reached without violating some of the actuation constraints. In this section, it is investigated if by tuning any of the reduction ratios n_j, the running velocity of the quadruped robot could be further increased. The issue is not straightforward, as by introducing a supplementary reduction ratio $n_{j,e}$ (for instance using a belt transmission mechanism) to the gearbox reduction ratio n_j (resulting in a total reduction of $n_{j,e}n_j$), the maximum permissible torques and angular velocity and the rotor inertia scale as,

$$\tau'_{ct,j} = n_{j,e}\tau_{ct,j}$$
$$\tau'_{st,j} = n_{j,e}\tau_{st,j} \quad (5\text{-}6)$$

$$\dot{\theta}'_{\max,j} = \dot{\theta}_{\max,j} / n_{j,e} \quad (5\text{-}7)$$

$$I'_{r,j} = n^2_{j,e}(n^2_j I_{r,j}) \quad (5\text{-}8)$$

Note that while torques are multiplied by $n_{j,e}$, the rotor inertia is multiplied by $n^2_{j,e}$. As the motion studied is highly dynamic with significant joint angular acceleration and rotor inertia at the exit of the gearbox $n^2_j I_{r,j}$ $(n_1 = 43, n_2 = 53)$, a potential increase in torque constraints by $n_{j,e}$ may lead to an overall increase in torque requirement that scales by $n^2_{j,e}$, hence leading to a decrease in the performance of the robot. To investigate the role of a supplementary reduction ratio in the maximum running velocity of the robot, two experiments are conducted. Firstly, for the parameter space of Section 5.4, a

supplementary reduction ratio of $(48:26)$ is introduced in both motors, to evaluate the hypothesis that an extra reduction ratio worsens the maximum achievable velocity of the robot. Note that $(48:26)$ is the extra reduction ratio the previous version of the *NTUA Laelaps* quadruped had for every actuating motor. Then to determine if the increase or the decrease of the reduction ratio leads to a better running velocity, various combinations of supplementary reductions are sought, with $n_{j,e} > 1$ or $0 < n_{j,e} \leq 1$ for the parameter space of Section 5.4.

With a supplementary reduction ratio of $(48:26)$ introduced to both motors, the resulting actuation parameters are shown in Table 5-10. The maximum velocity that could be reached was 2.083 m/s for the parameter set shown in Table 5-11. For a larger velocity, both the continuous torque and the maximum angular velocity constraints at the knee joints were violated. This is due to the increase of torque requirement due to the increased rotor inertia and the decrease of the maximum angular velocity constraint, incurred by the supplementary reduction ratio $(48:26)$.

Table 5-10. Actuation parameters at the exit of the extra reduction with $n_{j,e} = (48:26)$.

j	$I'_{r,j}$ [kg m^2]	$\tau'_{\mathrm{ct,j}}$ [N m]	$\tau'_{\mathrm{st,j}}$ [N m]	$\dot{\theta}'_{\mathrm{max,j}}$ [rad/s]
1	$386 \cdot 10^{-7}$	19.56	51.69	7.94
2	$1001 \cdot 10^{-7}$	27.62	83.08	6.07

Table 5-11. Set of optimal parameters for $n_{j,e} = (48:26)$.

Config.	$y(0)$ [m]	a_{fx}	T [s]	l_1 [m]	l_{20} [m]	k [N / m]	d_{in} [mm]	d_{out} [mm]
KF	0.80	0.40	0.60	0.27	0.65	2500	23	25

The non linear dependence of the rotor's inertia in the exit of the extra reduction $I'_{r,j}$, on the extra reduction ratio $n_{j,e}$, creates the need for a more systematic approach. To better understand the relationship between the extra reduction ratio $n_{j,e}$ and the maximum achievable horizontal velocity, the method was implemented for various reduction ratios in the vicinity of the nominal reduction ($n_{j,e} = 1$, $n_1 = 43$, $n_2 = 53$). The candidate values for $n_{1,e}$ were taken in the interval $\{0.6, 0.8, 0.9, 1.0, 1.2\}$ and those for $n_{2,e}$ in the interval $\{0.8, 1.0, 1.2\}$. The maximum velocities found for each of the above 15 combinations and the corresponding optimal set of parameters are shown in Table 5-12.

It is obvious that the maximum achievable velocity is found for the sets of $(n_{1,e}, n_{2,e}) = (0.8, 1)$ or $(0.9, 1)$ or $(1, 1)$, see also Figure 5-6. For combinations of extra reduction ratios near $n_{1,e} = 1.2$ or $n_{2,e} = 1.2$, robot velocities less than 2.760 m/s could be achieved, with the minimum value found (2.760 m/s) for $(n_{1,e}, n_{2,e}) = (1.2, 0.8)$. For combinations of smaller reduction ratios, velocities from 2.756 to 2.817 m/s could be achieved.

The results from this parametric search provide strong indications that maximum velocities can be achieved for total reduction ratios for the motors acting on the hip and the knee joints respectively,

$$n_{1,e} n_1 = (0.90 \pm 0.15)43 = 38.70 \pm 6.45 \quad (5\text{-}9)$$

$$n_{2,e} n_2 = (1.00 \pm 0.05)53 = 53 \pm 2.65 \quad (5\text{-}10)$$

In any case, the reduction ratios of the actuating motors should not further increase, as this would drastically limit the maximum horizontal velocity the quadruped robot could reach.

Table 5-12. Optimal set of parameters and velocity for various combinations of $n_{1,e}, n_{2,e}$.

a / a	$n_{1,e}$	$n_{2,e}$	max($\dot{x}$) [m / s]	$y(0)$ [m]	a_{fx}	T [s]	l_1 [m]	l_{20} [m]	k [N / m]	d_{in} [mm]	d_{out} [mm]
1	0.6	0.8	2.775	0.70	0.6	0.55	0.11	0.70	2500	23	25
2	0.6	1.0	2.775	0.70	0.6	0.55	0.11	0.70	2500	23	25
3	0.6	1.2	2.737	0.70	0.6	0.55	0.12	0.69	2500	23	25
4	0.8	0.8	2.756	0.70	0.6	0.55	0.13	0.68	2500	23	25
5	0.8	1.0	**2.817**	0.70	0.6	0.55	0.15	0.66	2500	23	25
6	0.8	1.2	2.748	0.70	0.6	0.55	0.13	0.68	2500	23	25
7	0.9	0.8	2.761	0.70	0.6	0.55	0.16	0.65	2500	23	25
8	0.9	1.0	**2.817**	0.70	0.6	0.55	0.18	0.63	2500	23	25
9	0.9	1.2	2.755	0.70	0.6	0.55	0.16	0.65	2500	23	25
10	1.0	0.8	2.749	0.70	0.6	0.55	0.18	0.63	2500	23	25
11	1.0	1.0	**2.817**	0.70	0.6	0.55	0.20	0.61	2500	23	25
12	1.0	1.2	2.658	0.80	0.7	0.80	0.17	0.84	1250	23	25
13	1.2	0.8	2.616	0.65	0.6	0.55	0.27	0.49	2500	23	25
14	1.2	1.0	2.716	0.70	0.8	0.80	0.23	0.68	1250	23	25
15	1.2	1.2	2.735	0.75	0.8	0.85	0.22	0.74	1250	26	28

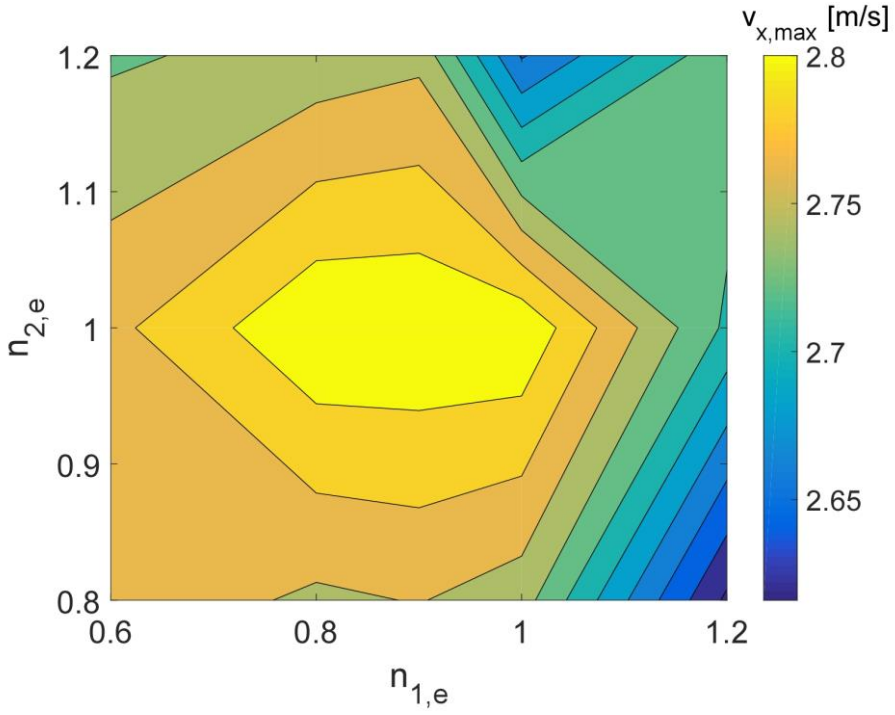


Figure 5-6. Maximum horizontal velocity as a function of the extra reduction ratios $n_{1,e}, n_{2,e}$.

6 Conclusions and Future Work

6.1 Conclusions

The work conducted in this thesis was inspired by the potential of the quadruped robots to improve the life quality in modern societies. Their great advantage in rugged terrain compared to wheeled or tracked vehicles, could be exploited in saving human lives in search and rescue missions or improve productivity and safety in agricultural or industrial environment. Due to the challenging nature of the quadrupedal locomotion, many issues concerning the design of the quadruped robot and escpecially the leg design are still considered unresolved in existing research literature.

In an effort to enrich current research, a leg design methodology was devised intended to improve and maximize the running performance of a quadruped. The proposed leg design methodology, took into account the estimated mass and inertia of the robot, the candidate leg architecture (3 dof leg), the desired materials for the leg components, the available actuation system and and estimated terrain parameters and provided design (leg parameters) and control (gait parameters) indications for maximizing running performance.

Utilizing this method, for robot parameters close to these of the *NTUA Laelaps*, the running performance of the robot for alternative gaits, knee configurations, acting joints for the available motors, and reduction ratios was optimized and the various results were compared. Although the robot physical descriptions used were expressed in the sagittal plane, introducing a frontal plane tipover stability criterion, solutions were also found capable for fast, stable motion against terrain or force perturbations in the frontal plane. To understand to which end the proposed results are reasonable, a set of optimal solutions was evaluated in an independent platform consisting of more realistic physical descriptions and controllers and the robot's running performance was validated.

The results showed that for the *NTUA Laelaps* robot body and actuation system, the maximum running performance could be achieved with the trotting gait for legs in the knee forward configuration. For this gait and configuration, it was found which of the two available motors should actuate on which leg joint, thus succeeding a maximum running velocity of 3.039 m/s. Making reasonable assumptions for perturbations in the terrain's slope and for external perturbation forces and by demanding frontal plane stability, the robot was proposed to run with a 40% shorter leg, reducing thus the maximum running velocity only by 7%. Investigating the effect of introducing a supplementary reduction ratio, the reduction ratio combinations that maximized running velocity were found and it was concluded that by further increasing the motors' reduction ratios, the running performance of the robot is more probable to deteriorate.

The previous version of the *NTUA Laelaps* has legs with equal segments and with a total length equal to the hip separation distance, made of duralumin and steel components. The actuators system is so arranged, that the brushed DC motor acts on the hip joint and the brushless on the knee joint, both having an additional reduction ratio of (48:26). The results of this thesis indicated that by altering some of the attributes of the existing system the running performance of the robot can be drastically increased.

To achieve a large running velocity the robot should move with a trotting gait instead of a bounding gait. The duralumin and steel components should be replaced by carbon fiber tubes where possible to reduce the leg weight and inertia. The total leg length should be greater than the hip separation distance to achieve a larger stride length but in the same time adequately short to facilitate frontal plane stability. The proximal to the body leg segment should be shorter to the distal leg segment for a better distribution of the torque requirement between the hip and the knee joint. Morever, it was shown that with the brushed motor acting on the knee joint and the brushless on the hip joint the robot can move faster in comparison with the existing arrangement. Finally, the supplementary reduction ratio should be kept close to one, as for a larger reduction ratio the rotor inertia and the torque requirement increase to a much larger scale than the maximum permissible torque.

6.2 Future work

In the presented thesis a leg design methodology was devised that given the inertial properties of the robot, the leg architecture and leg materials, the running gait, ground properties and the actuation system, optimized the running performance of the robot by proper selection of certain leg parameters (length, proportions, cross sectional area and stiffness) and gait parameters (foot placement, CoM height of running, amplitude of horizontal force, stride period). Future work can focus on studying the effect that alternative inputs can have on the robot performance predicted by the employed method and on further validating the results through leg construction and experimentation. If the results are experimentally validated, the method could be extended also to other tasks instead of running through substraction.

An interesting issue related to the inputs of the method is the DF of the selected gaits. For a running velocity greater than a threshold, quadrupedal animals chose to run in gaits with *full flight* phases, i.e. phases where no leg is in contact with the ground (DF<0.5). For instance, cheetahs and dogs run in maximum speed in duty factors less than 0.2 [63]. For the results presented in this thesis, at least two legs are in contact with the ground at every time instant (DF=0.5). By further decreasing the DF of each leg during the gait, the magnitude of the sinusoidal impulsive force exerted on each leg would increase (see eq. (2-53)), and so would the cumulative time the leg spends in flight. As a result, the leg joint torque requirements during stance phase would increase, but the required accelerations, angular velocities and torques throughout flight phase would probably decrease (same joint trajectories during full flight for longer flight phase duration). This could potentially lead to a better distribution of joint torque requirements throughout the stride, leading to a better running performance. Another opportunity presented by full flight phases is the disengagement of the stride length from the leg length. This attribute could probably help offset the trade-off between sagittal plane tipover stability and maximum running velocity.

Another issue that should be addressed is the costruction of the legs. The leg joints should be properly designed to withstand the exerted stresses, while respecting the weight budget set in this thesis. Carbon fiber tubes with the desired density and inner diameter/outer diameter combination

should be used, and the leg segment lengths should be taken as prescribed by the results of this thesis. A proper mechanism for the prismatic degree of freedom should be considered, that can accommodate the compression spring of the desired stiffness. Kevlar ropes could be used as a leightweight solution to transmit power from the motor to the knee joint. If a belt transmission mechanism is used for power transmission between the motors and the joints, the optimal combinations of extra transmission ratios $n_{j,e}$ should be taken into consideration.

After the legs are constructed, they should be mounted on the robot's body and running experiments should be conducted. The control scheme utilized for the running task should take into consideration the proposed gait, apex running height $y(0)$ and positioning of the robot's legs x_i resulting in the desired stride length λ, leg clearance b_e and stride period T. By conducting running experiments, the gaps between methodology and hardware results should be identified, which may lead in re-designing loops. Another proposition for experimental future work, would be to run in a terrain with perturbations in the frontal plane and measure through toe mounted sensors, the total perturbation force F_p exerted on the robot. This could lead to a more realistic correlation of terrain incurred perturbations with maximum permissible running height $y(0)$ for tipover stability in the frontal plane.

Finally, after the results of the method are experimentally validated, the leg design method could also be applied through abstraction for other tasks, e.g. stair climbing. Although many changes would be necessary (description of the terrain, stability consitions, performance criteria), the basic structure of the method would remain intact. An input stage where the robot and actuation properties, the terrain, the task and the footfall sequence to achieve it would be defined; a first stage, where the CoM trajectory of the robot will be optimized based on some performance criterion, a second stage that finds permissible leg solutions, where the joint trajectories are determined and constraints (actuation, strength) are evaluated and a final stage where alternative parameters are provided as inputs in the other two stages, and the optimal set of solutions for the performance criterion are determined.

7 References

[1] F. Hadarson, "Locomotion for difficult terrain," *Internal Report*, Department of Machine Design, Royal Institute of technology, Sweden, 1998.

[2] M. Raibert, "Trotting, pacing and bounding by a quadruped robot," *Journal of Biomechanics,* vol. 23, no. 1, 1990, pp. 79-81.

[3] http://www.bostondynamics.com/robot_cheetah.html.

[4] H. W. Park, P. M. Wensing and S. Kim, "Online Planning for Autonomous Running Jumps Over Obstacles in High-Speed Quadrupeds," in *Robotics: Science and Systems*, 2015.

[5] E. Ackerman, "Spot Is Boston Dynamics' Nimble New Quadruped Robot," *IEEE SPECTRUM*, 2015.

[6] E. Ackerman, "Whoa: Boston Dynamics Announces New WildCat Quadruped Robot," *IEEE SPECTRUM*, 2013.

[7] E. Z. Moore, "Leg Design and Stair Climbing Control for the RHex Robotic Hexapod", *ME Thesis*, Department of Mechanical Engineering, McGill University, 2002.

[8] A. Saunders, D. I. Goldman, R. J. Full and M. Buehler, "The RiSE Climbing Robot: Body and Leg Design," in *Unmanned Systems Technology VIII, Proc. of SPIE Vol. 6230*, 2006.

[9] S. Kim, J. Clark and M. Cutkosky, "iSprawl: Design and tuning for high-speed autonomous open-loop running," *The International Journal of Robotics Research,* vol. 25, no. 9, 2006, pp. 903-912.

[10] S. Seok, A. Wang, M. Y. Chuah, D. Otten, J. Lang and S. Kim, "Design Principles for Highly Efficient Quadrupeds and Implementation on the *MIT Cheetah* Robot," in *2013 IEEE International Conference on Robotics and Automation (ICRA)*, Karlsruhe, Germany, 2013.

[11] J. A. Smith and I. Poulakakis, "Rotary gallop in the untethered quadrupedal robot *Scout II*," in *Intelligent Robots and Systems (IROS)*, 2004.

[12] N. Cherouvim and E. Papadopoulos, "Energy saving passive-dynamic gait for a one-legged," *Robotica,* vol. 24, 2006, pp. 491-498.

[13] M. Haberland and S. Kim, "On extracting design principles from biology : I. Method–General answers to high-level design questions for bioinspired robots," *Bioinspiration & Biomimetics,* vol. 10, 2015, pp. 1-14.

[14] M. Hutter, "*StarlETH* & Co. – Design and Control of Legged Robots with Compliant Actuation", *MSc Thesis*, ETH Zurich, 2013.

[15] C. Semini, "*HyQ* - Design and Development of a Hydraulically Actuated Quadruped Robot", *PhD Thesis*, University of Genoa and Italian Institute of Technology, 2010.

[16] J. Rummel and A. Seyfarth , "Stable Running with Segmented Legs," *The International Journal of Robotics Research ,* vol. 27, no. 8, 2008, pp. 919-934.

[17] H. Witte, R. Hackert, K. E. Lilije, N. Schilling, D. Voges, G. Klauer, W. Lig, J. Albiez, A. Seyfarth, D. Germann, M. Hiller, R. Dillmann and M. S. Fischer, "Transfer of Biological Principles into the Construction of Quadruped Walking Machines," in *Second Workshop on Robot Motion and Control*, 2001.

[18] H. Witte, R. Hackert, W. Ilg, J. Biltzinger, N. Schilling, F. Biedermann, M. Jergas, H. Preuschoft and M. S. Fischer, "Quadrupedal mammals as paragons for walking machines," in *Proceedings of AMAM - Adaptive Motions in Animals and Machines*, Kyoto, 2003.

[19] A. Sprowitz, A. Tuleu, M. Vespignani, M. Ajallooeian, E. Badri and A. J. Ijspeert, "Towards dynamic trot gait locomotion: Design, control, and experiments with Cheetah-cub, a compliant quadruped robot," *The International Journal of Robotics Research,* vol. 32, no. 8, 2013, pp. 932-950.

[20] D. Ajilo, "Mechanical Design of a Quadruped Robot", *BSc Thesis*, Department of Mechanical Engineering, MIT Press, 2015.

[21] "Spring-mass Walking with Atrias in 3D: Robust Gait Control Spanning Zero To 4.3 kph on a Heavily Underactuated Bipedal Robot," in *Proceedings of ASME 2015 Dynamic Systems and Control Conference*, Columbus, Ohio, 2015.

[22] G. Kenneally, A. De and D. E. Koditschek, "Design Principles for a Family of Direct-Drive Legged Robots," *IEEE Robotics and Automation Letters,* vol. 1, no. 2, 2016, pp. 900-907.

[23] J. Grimes and J. Hurst, "THE DESIGN OF ATRIAS 1.0 A UNIQUE MONOPOD, HOPPING ROBOT," in *CLAWAR 2012 – Proceedings of the Fifteenth International Conference on Climbing and Walking Robots and the Support Technologies for Mobile Machines*, Baltimore, 2012.

[24] R. M. Murray, Z. Li and S. S. Sastry, "A mathematical introduction to robotic manipulation.", CRC Press, 1994.

[25] M. Schmidt and M. Fischer, " Morphological integration in mammalian limb proportions: Dissociation between function and development.," *Evolution,* vol. 63, no. 3, 2009, pp. 749-766.

[26] L. McDowell, "The Dog in Action", Orange Judd Publishing Company Inc., 1950.

[27] M. Hutter, D. C. Remy, M. A. Hoepflinger and R. Siegwart, "ScarlETH: Design and control of a planar running robot," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2011.

[28] P. Chatzakos and E. Papadopoulos, "Bio-inspired design of electrically-driven bounding quadrupeds," *Mechanism and Machine Theory,* vol. 44, 2009, p. 559–579.

[29] J. McKenzie, "Design of Robotic Quadruped Legs", *BSc Thesis*, MIT, 2012.

[30] : http://www.ai.mit.edu/projects/leglab/robots/quadruped/quadruped.html.

[31] X. Zhou and S. Bi, "A survey of bio-inspired compliant legged," *Bioinspiration & Biomimetics,* vol. 7, 2012.

[32] D. V. Lee and S. G. Meek, "Directionally compliant legs influence the intrinsic pitch behaviour of a trotting quadruped," *Proceedings of the Royal Society B: Biological Sciences,* vol. 272, no. 1563, 2005, pp. 567-572.

[33] S. Meek, S. Kim and M. Anderson, "Stability of a trotting quadruped robot with passive, underactuated legs," in *IEEE International Conference on Robotics and Automation (ICRA)*, Pasadena, 2008.

[34] Z. Xiuli, Z. Haojun, G. Xu, C. Zhifeng and Z. Liyao, "A Biological Inspired Quadruped Robot: Structure and Control," in *IEEE International Conference on Robotics and Biomimetics*, 2005.

[35] B. Li, Y. Li, X. Rong, J. Meng and H. Chai, "The Effects of Leg Configurations on Trotting Quadruped Robot," in *Chinese Intelligent Automation Conference*, 2013.

[36] M. Jones and J. Hurst, "Effects of leg configuration on running and walking robots," in *Fifteenth International Conference on Climbing and Walking Robots and the Support Technologies for Mobile Machines (CLAWAR)*, Baltimore, 2012.

[37] M. S. Fischer and R. Blickhan, "The Tri-Segmented Limbs of Therian Mammals: Kinematics, Dynamics, and Self-stabilization— A Review," *Journal of Experimental Zoology,* vol. 305, no. A, 2006, pp. 935-952.

[38] http://biorob.epfl.ch/cheetah.

[39] http://news.mit.edu/2015/cheetah-robot-lands-running-jump-0529.

[40] https://biomimetics.mit.edu/research/mit-super-mini-cheetah.

[41] http://www.ghostrobotics.io/*MiniTaur*/.

[42] M. Raibert, "Legged Robots that Balance", Cambridge, MIT Press, 1986.

[43] R. J. Full, C. T. Farley and J. M. Winters, "Musculoskeletal Dynamics in Rhythmic Systems: A Comparative Approach to Legged Locomotion," in *Biomechanics and Neural Control of Posture and Movement*, Springer New York, 1999, pp. 192-205.

[44] C. Semini, V. Barasuol, T. Boaventura, M. Frigerio and J. Buchli, "Is Active Impedance the Key to a Breakthrough," in *International Symposium of Robotics Research (ISRR)*, 2013.

[45] S. Seok, A. Wang, D. Otten and S. Kim, "Actuator Design for High Force Proprioceptive Control in Fast Legged Locomotion," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Vilamoura, Algarve, Portugal, 2012.

[46] M. A. Hopkins, A. Leonessa and B. Y. Lattimer, "Optimisation-Based Whole-Body Control of a Series Elastic Humanoid Robot," *International Journal of Humanoid Robotics,* vol. 12, 2014.

[47] S. Wolf and G. Hirzinger, "A New Variable Stiffness Design: Matching Requirements of the Next Robot Generation," in *IEEE International Conference on Robotics and Automation (ICRA)*, Pasadena, 2008.

[48] L. Mooney and H. Herr, "Continuously-Variable Series-Elastic Actuator," in *IEEE International Conference On Rehabilitation Robotics*, 2013.

[49] J. Austin, A. Schepelmann and H. Geyer, "Control and Evaluation of Series Elastic Actuators with Nonlinear Rubber Springs," in *IEEE/RSJ Intelligent Robots ans Systems (IROS)*, 2015.

[50] A. Mutka, E. Koco and Z. Kovacic, "Adaptive Control of Quadruped Locomotion Through Variable Compliance of Revolute Spiral Feet," *Journal of Bionic Engineering,* vol. 12, no. 3, 2015, pp. 352-360.

[51] X. Liu, A. Rossi and A. Poulakakis, "SPEAR: A Monopedal Robot with Switchable Parallel Elastic Actuation," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2015.

[52] G. Folkertsma, S. Kim and S. Stramigioli, "Parallel stiffness in a bounding quadruped with flexible spine," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Vilamoura, Algarve, Portuga, 2012.

[53] Y. Yesilvsky, W. Xi and C. D. Remy, "A Comparison of Series and Parallel Elasticity in a Monoped Hopper," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2015.

[54] M. Hutter, C. Gehring, D. Jud, A. Lauber, C. D. Bellicoso, V. Tsounis, J. Hwangbo, K. Bodie, P. Fankhauser, M. Bloesch, R. Diethelm, S. Bachmann, A. Melzer and M. Hoepflinger, "ANYmal - A Highly Mobile and Dynamic Quadrupedal Robot," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Daejeon, 2016.

[55] https://www.youtube.com/watch?v=M8YjvHYbZ9w.

[56] https://www.youtube.com/watch?v=wE3fmFTtP9g.

[57] C. Semini, V. Barasuol, J. Goldsmith, M. Frigerio, M. Focchi, Y. Gao and D. G. Galdwell, "Design of the Hydraulically-Actuated, Torque-Controlled Quadruped Robot *HyQ*2Max," *IEEE/ASME Transactions on Mechatronics,* vol. PP, no. 99, 2016.

[58] D. Karssen, M. Haberland, M. Wisse and S. Kim, "The Optimal Swing-Leg Retraction Rate for Running," in *IEEE International Conference on Robotics and Automation (ICRA)*, Shangai, China, 2011.

[59] R. M. Walter and D. R. Carrier, "Ground forces applied by galloping dogs," *The Journal of Experimental Biology,* vol. 210, 2007, pp. 208-216.

[60] A. Ananthanarayanan, M. Azadi and S. Kim, "Towards a bio-inspired leg design for high-speed running," *Bioinspiration and Biomimetics,* vol. 7, 2012.

[61] R. M. Alexander and A. S. Jayes, "A dynamic simularity hypothesis for the gaits of quadrupedal mammals.," *J. Zool. Lond.,* vol. 201, 1983, pp. 135-152.

[62] L. D. Maes, M. Herbin, R. Hackert, V. L. Bels and A. Abourachid, "Steady locomotion in dogs: temporal and associated spatial coordination patterns and the effect of speed," *The Journal of Experimental Biology,* vol. 211, 2008, pp. 138-149.

[63] P. E. Hudson, S. A. Corr and A. M. Wilson, "High speed galloping in the cheetah (Acinonyx jubatus) and the racing greyhound (Canis familiaris): spatio-temporal and kinetic characteristics," *The Journal of Experimental Biology,* vol. 215, 2012, pp. 2425-2434.

[64] J. J. Collins and I. N. Stewart, "Coupled Nonlinear Oscillators and the Symmetries of Animal Gait," *Journal of Nonlinear Science,* vol. 3, 1993, pp. 349-392.

[65] T. McMahon and G. Cheng, "The Mechanics of Running: How does stiffness Couple with Speed," *Journal of Biomechanics,* vol. 23, no. 1, 1990, pp. 65-78.

[66] K. Koutsoukis and E. Papadopoulos, "On Passive Quadrupedal Bounding with Flexible Linear Torso," *International Journal of Robotics (Theory and Applications),* vol. 4, no. 2, 2015, pp. 1-8.

[67] H. Dai, A. Valenzuela and R. Tedrake, "Whole-body motion planning with centroidal dynamics and full kinematics," in *IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, 2014.

[68] D. E. Orin, A. Goswami and S. H. Lee, "Centroidal dynamics of a humanoid robot," *Autononomous Robots,* vol. 35, 2013, pp. 161-176.

[69] S. Ha, S. Coros, A. Alspach, J. Kim and K. Yamane, "Task-based Limb Optimization for Legged Robots," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Daejeon, 2016.

[70] E. Papadopoulos, "Electromechanical Systems for Power Conversion", Foundas Press, 2010.

[71] K. Machairas and E. Papadopoulos, "An Active Compliance Controller for Quadruped Trotting," in *24th Mediterranean Conference on Control and Automation (MED)*, Athens, 2016.

[72] I. Poulakakis, E. Papadopoulos and M. Buehler, "On the Stability of the Passive Dynamics of Quadrupedal Running with a Bounding Gait," *The International Journal of Robotics Research,* vol. 25, no. 7, 2006, pp. 669-687.

[73] M. H. Raibert, "Trotting, pacing and bounding by a quadruped robot," *Journal of Biomechanics,* vol. 23, 1990, pp. 79-98.

[74] P. Chatzakos, Parametric Analysis and Systematic Design of Legged Robots, *PhD Thesis*, (In Greek), Athens: NTUA Press , 2009.

[75] B. Siciliano, L. Sciavicco, L. Villani and G. Oriolo, Robotics: Modelling, Planning and Control, Springer, 2009.

[76] K. H. Hunt and F. E. Crossley, "Coefficient of Restitution Interpreted as Damping in Vibroimpact," *ASME Journal of Applied Mechanics,* vol. 42, no. 2, 1975, pp. 440-445.

[77] G. Gilardi and I. Sharf, "Literature survey of contact dynamics modeling," *Mechanism and Machine Theory,* vol. 37, no. 10, 2002, pp. 1213-1239.

[78] L. C. Bo and D. Pavelescu, "The friction-speed relation and its," *Wear,* vol. 82, no. 3, 1982, pp. 227-289.

[79] V. Vasilopoulos , I. Paraskevas and E. Papadopoulos, "Control and Energy Considerations for a Hopping Monopod on Rough Compliant Terrains," in *IEEE International Conference on Robotics and Automation (ICRA*), Seattle, 2015.

[80] https://www.youtube.com/watch?v=STCBHurECp4.

[81] E. Papadopoulos and D. A. Rey, "The Force-Angle Measure of Tipover Stability Margin for Mobile Manipulators," *Vehicle System Dynamics,* vol. 33, 2000, pp. 29-48.

[82] https://www.youtube.com/watch?v=UBHJqnM8RTU.

[83] https://www.youtube.com/watch?v=chPanW0QWhA.

[84] M. Stefanovska, S. Risteska, B. Samakoski, G. Maneski and B. Kostadinoska, "Theoretical and Experimental Bending Properties of Composite Pipes," *International Journal of Environmental, Chemical, Ecological, Geological and Geophysical Engineering,* vol. 9, no. 6, 2015.

[85] P. E. Hudson, S. A. Corr and A. M. Wilson, "High speed galloping in the cheetah (Acinonyx jubatus) and the racing greyhound (Canis familiaris): spatio-temporal and kinetic characteristics," *The Journal of Experimental Biology,* vol. 215, 2012, pp. 2425-2434.

Appendix A - Carbon tube properties

Available carbon fiber tube suppliers provide only approximate values for the density and the strength of their products, without certification. This motivated the author to purchase representative carbon tube samples and subject them to direct measurements and tests. The density of the specimens was calculated from their measured mass and volume and their ultimate strength in compression and bending was figured out by conducting appropriate compression and three point bending tests.

Two specimens were sampled from a carbon fiber tube with inner diameter $d_{in} = 18mm$, outer diameter $d_{out} = 20mm$ and total length $1000m$. The specimens were cut from the tube with sawing, and the specimen intended for the compression test was also subjected to turning, to ensure uniform contact of the specimen with the compression plates. The length of the compression specimen was taken equal to $l_c = 61.4mm$, to ensure that the ratio $l_c / d_{out} \simeq 3$ is small enough to avoid buckling during the compression test. The length of the bending specimen was taken equal to $l_b = 240mm$, resulting in a length to outer diameter ratio $l_b / d_{out} = 12$. The length of the bending specimen should be adequately long in comparison with its outer diameter to ensure that bending (and not compression or shear) is the main cause of stress. In literature [84], a typical satisfactory length to outer diameter ratio for bending of composite tubes is $l_b / d_{out} = 8$. The mass of the compression specimen was measured as $m_c = 0.056kg$ and that of the bending specimen as $m_b = 0.022kg$. The density of the specimens was calculated as,

$$\rho = \frac{m_{sp}}{V}, \qquad V = l_{sp}\pi(d_{out}^2 - d_{in}^2)/4 \tag{A-1}$$

where m_{sp}, l_{sp} is the mass and the length of the specimens. By replacing the masses, the lengths and the diameters of the specimens in Equations (A-1), the density of the carbon fiber composite was found as, $\rho = 1466kg/m^3$.

The universal testing system Instron 4482 was used for the compression and the bending, see Figure A-1 (a). The experimental setup for the compression tests consisted of two compression plates, Figure A-1 (b) and that for the three point bending test consisted of two support pins and a punch, see Figure A-1 (c). The curvature radii of the supports and the punch were proper ($r_c = 10mm$) to avoid local overstressing and denting of the subjects. The support span (distance between support pins) was taken equal to $200mm$ to avoid slippage of the specimen during bending. The compression experiment was conducted for a constant compression velocity of $6mm/\sec$ and the bending experiment for a constant punch velocity of $1mm/\min$. Load and deflection were recorded throughout the experiments and the results were presented in respective graphs, see Figure A-2.

From the maximum measured loads and the geometry of the carbon fiber tubes, the compression and bending strengths S_c, S_b were calculated as,

$$S_c = F_{c,\max} / A$$
$$S_b = \frac{M_{b,\max}}{I_a}\frac{d_{out}}{2}, \quad M_{b,\max} = F_{b,\max} l_s / 2 \tag{A-2}$$

where $F_{c,\max}$ is the maximum recorded compressive load, A is the cross sectional area of the carbon fiber tubes, $F_{b,\max}$ is the maximum recorded bending load, l_s is the support span length and I_a is the area moment of inertia of the carbon fiber tubes. For $F_{c,\max} = 11705\,\mathrm{N}$ and $F_{b,\max} = 650\,\mathrm{N}$ the compressive and bending strengths were found as, $S_c = 200\,\mathrm{MPa}$ and $S_b = 240\,\mathrm{MPa}$. The maximum permissible normal stress value was taken as the minimum of the two,

$$S_{c,u} = \min(S_c, S_b) = 200\,\mathrm{MPa} \tag{A-3}$$

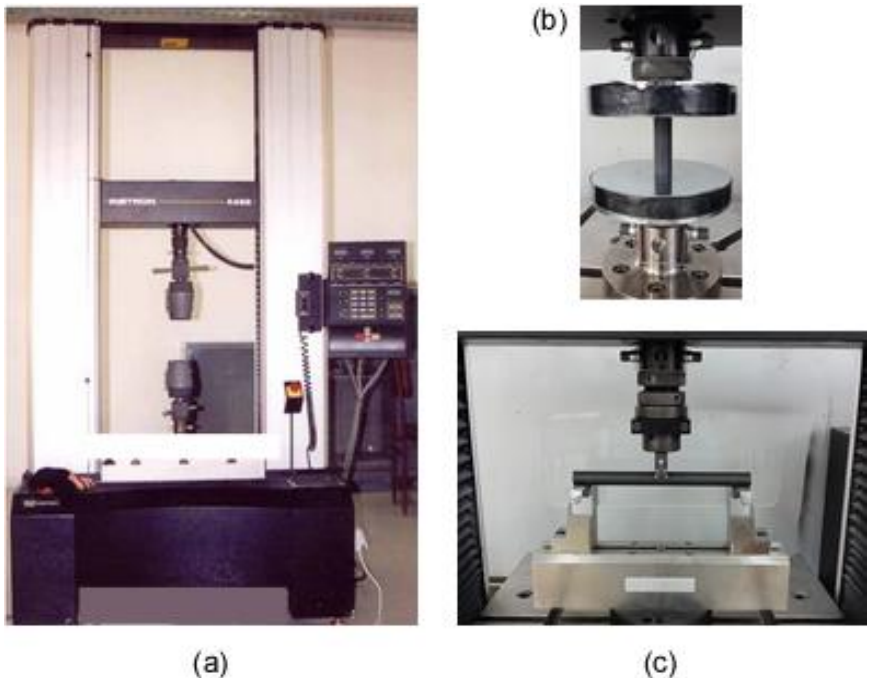


Figure A-1. (a) Instron 4482 testing system and experimental setups for (b) compression and (c) three point bending.

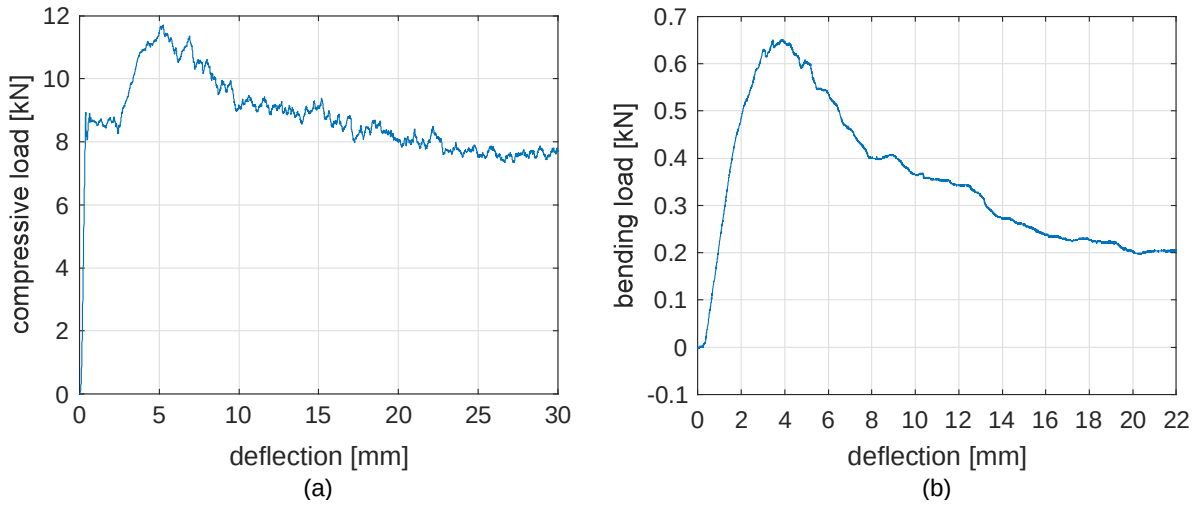


Figure A-2. Load - deflection diagrams for the (a) compression and (b) three point bending tests.

Appendix B - Codes in Matlab

Appendix B is intended to explain to the reader the main scripts and functions written and used by the author as the algorithmic implementation of the leg design method presented in the dissertation. The purpose of thoroughly explaining the codes is twofold; in case of interest, the reader should be able to reproduce the results of the method and use the codes provided for their own research interests. A list of the written and used function and scripts with a short description is provided below (Figure B-1):

- Main_vx000.m: The algorithmic implementation of the leg design method presented in Figure 3-1, corresponding to a running velocity vx ($\dot{x}(0)$). This script is intended to be run in parallel for different velocities. To run multible of these scripts in parallel, rename the script accordingly to the input velocity. For instance, for the velocities vx=2.50 m/s and vx=2.80 m/s, create two copies named Main_vx250.m and Main_vx280.m, change the respective velocities inputs and run the scripts in two different MATLAB windows. The output of this script is the PSC.mat matrix containing the permissible leg and gait parameter solutions to run in the desired velocities and indicators if the solutions are truncated by the parameter search boundaries. At the end of this script PSC.mat matrix is saved in the workspace folder as PSC_vx000.mat.
- PostProcessing.m: Loads one of the optimal solutions sets saved in the matrix PSC_vx000.mat and calculates the robot CoM and leg joint trajectories, the joint torques and stresses exerted on the tubular leg segments. The user can chose to display an animation of the robot running, plot the joint torques and angular velocities to compare them with the actuation constraints or plot the stresses applied on the tubular leg segments.
- c_i.m: Has as output the value of the variable c_i (see eq. (8-1))

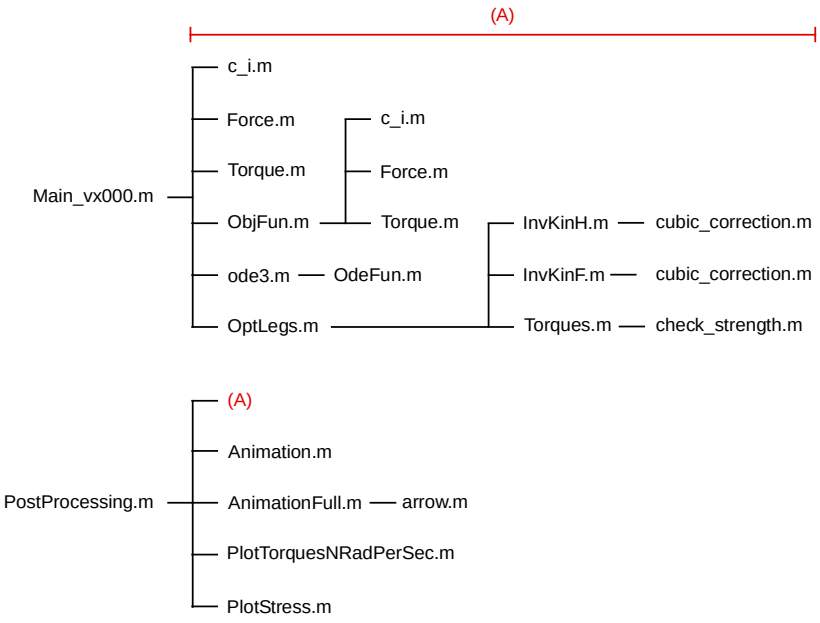


Figure B-1. Tree diagram showing the functions called from the two main scripts, Main_vx000.m and PostProcessing.m.

- Force.m: Provides as output the horizontal and vertical forces exerted on the CoM of the robot at every time instant, see eq. (2-43), (8-2).
- Torque.m: Calculates the total torque exerted on the robot body w.r.t its CoM, see eq. (2-45).
- ObjFun.m: Finds the value of the objective function (3-2) for a set of candidate footfalls x_i.
- ode3.m: A mathworks community function, implementation of the Runge - Kutta (3rd order) method with a fixed integration step.
- OdeFun.m: Provides as output the right hand side of the differential Equations system (2-40)-(2-42) at every time instant.
- OptLegs.m: Most of the leg parameter search of Stage 2 is conducted in OptLegs.m, having as outputs the leg parameters l_1, l_{20}, k that satisfy actuation constraints (2-100).
- InvKinH.m: Returns as output the leg joint trajectories of the hind legs, corresponding to a hybrid elliptical - cubic polynomial flight phase toe trajectory, see Section 2.7.
- InvKinF.m: Returns as output the leg joint trajectories of the hind legs, corresponding to a hybrid elliptical - cubic polynomial flight phase toe trajectory, see Section 2.7.
- cubic_correction.m: Takes as input the leg joint trajectories corresponding to an elliptical toe flight phase trajectory, and gives as output the joint trajectories corresponding to a hybrid elliptical - cubic polynomial flight phase toe trajectory, see conditions (3-16), (3-23) and polynomials (3-24), (3-25)
- Torques.m: Given the leg joint trajectories and dynamics of the quadruped robot the leg joint torques are calculated, eq. (2-38).
- check_strength.m: For alternative user provided sets of d_{in}, d_{out} (see Table 4-2) the strength of the tubular leg segments is evaluated, see inequalities (2-90).
- Animation.m: Given the CoM trajectories and the footfalls of the quadruped robot, an animation is displayed, showing the motion of the centroidal model.
- AnimationFull.m: Given the CoM and leg joint trajectories of the quadruped robot, an animation is displayed, showing the motion of the quadruped robot.
- arrow.m: A mathworks community function, used to plot the vectors of the ground forces with time.
- PlotTorquesNRadPerSec.m: The joint torques and angular velocities are expressed as a percentage of the corresponfing maximum permissible values and plotted vs. time in figures.
- PlotStress.m: Calculates (eq. (2-85)-(2-87)) and plots the stresses on the tubular leg segments, comparing them with the material's strength.

In the following sections, the written scripts and functions are presented in detail with helpful comments that guide the reader.

Main_vx000.m

```
%-------------------------------------------------------------------------%
%---------------------------- Main_vx000.m -------------------------------%
%-------------------------------------------------------------------------%
% This script searches for the leg and gait parameters for which hardware
% constraints are respected, while the robot runs in a user specified
% velocity. Create multible versions of this script with different names to
% run simulations with different velocities vx parallely. The output of
% this script is the matrix PSC which contains all the permissible leg and
% gait parameters for a running velocity vx and indeces warning the user if
% an upper or lower boundary has been reached.

clc
clear
tic

%-------------------------- ROBOT PROPERTIES -----------------------------%
% The legs' properties are set in Torques.m
% The legs' strength is set in check_strength.m
m=42;
d=0.3;
g=9.81;
a=1;
b=0.15;
I=1/12*m*(a^2+b^2);
%-------------------------------------------------------------------------%
%------------------ GROUND FORCE RELATED PARAMETERS ----------------------%
fs=2;
mi=0.65;
%-------------------------------------------------------------------------%
%---------------------------- GROUND LOCATION ----------------------------%
% Set desired clearance from the ground in InvKinHL.m, InvKinFL.m
yg=0;
%-------------------------------------------------------------------------%
%----------------------- ACTUATOR SPECIFICATIONS -------------------------%
% Specify rotor inertias and reduction ratios in Torques.m
% The values below are torques and ang. velocities after reduction.
tau2maxst=45;
tau2maxct=14.96;
th2dmax=11.21;
tau1maxst=45;
tau1maxct=14.96;
th1dmax=11.21;
%-------------------------------------------------------------------------%
%----------------------- TEMPORAL GAIT PARAMETERS ------------------------%
% HL-->1, FL-->2, FR-->3, HR-->4, tti=[t_td_i%, t_to_i%]
tt1=[0,50];
tt2=[50,100];
tt3=[0,50];
tt4=[50,100];
t21=tt2(1);
t22=tt2(2);
dfi=-t21*pi/(t22-t21);

% The code parts and scripts affected if gait is changed are:
% - Main_vx000.m, lines 134-228.
% - check_strength.m, CheckStrength.m replace (1:500), (501:1000) with the
% proper indeces for each leg's stance and flight.
% - cubic_correction.m replace the proper values to j0, jm, jf, t0, tm, tf,
% replace (1:500), (501:1000) with the proper indeces for each leg's stance
```

```
% midflight and flight.
% - Force.m, replace properly the arguments fi1x, fi1y and the phase
% differences dfix, dfiy.
% - InvKinFL.m, InvKinHL.m, reset wtri, fitri, replace (1:500), (501:1000)
% with the proper indeces for each leg's stance and flight.
% - PostProcessing.m, lines 58-65, 116-210
%-------------------------------------------------------------------------%
%------------------------- LOWER & UPPER BOUNDS --------------------------%
% The upper and lower bounds for the leg parametric search of Stage 2 are
% set in OptLegs.m
afxlb=0.1;
afxub=0.7;
Tlb=0.35;
Tub=0.75;
y0lb=0.55;
y0ub=1.1;
%-------------------------------------------------------------------------%
%----------------------- DESIRED INITIAL CONDITIONS ----------------------%
x0=1;
yd0=0;   % do not change yd0=0, method assumes appex as initial position.
th0=0;
thd0=0;
%-------------------------------------------------------------------------%
%--------------------- PARAMETRIC SEARCH OUTER STAGE ---------------------%
oc=0;
vx=0.5;   % change velocity manually, vx=xd0

for T=Tlb:0.05:Tub

for afx=afxlb:0.1:afxub

for y0=y0lb:0.05:y0ub

%------------------------ PARAMETER BOUND FLAGS --------------------------%
% Reconsider boundaries of parameter search if flags are equal to 1 at PSC.
bfafx=0;
bfT=0;
bfy0=0;
bfx1=0;
bfx2=0;
bfx3=0;
bfx4=0;
%-------------------------------------------------------------------------%
%----------------------- PARAMETRIC SEARCH STAGE 1 ----------------------%
%---------------------- FOOTFALL EXHAUSTIVE SEARCH ----------------------%
% initialization

dx=0.03;
n=100;
dt=T/(n-1);
minfpth=1e6;

t=zeros(n,1);
x=zeros(n,1);
xd=zeros(n,1);
xdd=zeros(n,1);
y=zeros(n,1);
yd=zeros(n,1);
ydd=zeros(n,1);
fx=zeros(n,1);
```

```
fy=zeros(n,1);
tau=zeros(n,1);
th=zeros(n,1);
thd=zeros(n,1);
thdd=zeros(n,1);


t(1)=0;
t(2)=dt;

x(1)=x0;
xd(1)=vx;
xd(2)=vx;
x(2)=x(1)+dt*xd(2);

y(1)=y0;
y(2)=y0;
yd(1)=yd0;
yd(2)=yd0;

th(1)=th0;
th(2)=th0;
thd(1)=thd0;
thd(2)=thd0;

% footfals xi exhaustive search
for x1=x(1)-d*cos(th(1)):dx:x(1)-d*cos(th(1))+2*d
    for x3=x(1)+d*cos(th(1)):dx:x(1)+d*cos(th(1))+2*d

        x2=0;
        x4=0;

        for j=1:3
            [c1,c2,c3,c4]=c_i(tt1,tt2,tt3,tt4,t(j),T);
            [fxj,fyj] = Force(c1,c2,c3,c4,tt1,dfi,afx,t(j),T,mi,m,g,fs);
            [tauj] = Torque(c1,c2,c3,c4,x1,x2,x3,x4,yg,x(j),y(j),fxj,fyj);
            fx(j)=fxj;
            fy(j)=fyj;
            tau(j)=tauj;
        end

        for j=3:tt1(2)*n/100
            t(j)=t(j-1)+dt;
            [c1,c2,c3,c4]=c_i(tt1,tt2,tt3,tt4,t(j),T);
            [fxj,fyj] = Force(c1,c2,c3,c4,tt1,dfi,afx,t(j),T,mi,m,g,fs);
            xdd(j)=1/m*(fxj);
            xd(j)=1/3*(2*dt*xdd(j)+4*xd(j-1)-xd(j-2));
            x(j)=dt^2*xdd(j)+2*x(j-1)-x(j-2);
            ydd(j)=1/m*(fyj-m*g);
            yd(j)=1/3*(2*dt*ydd(j)+4*yd(j-1)-yd(j-2));
            y(j)=dt^2*ydd(j)+2*y(j-1)-y(j-2);
            [tauj] = Torque(c1,c2,c3,c4,x1,x2,x3,x4,yg,x(j),y(j),fxj,fyj);
            thdd(j)=1/I*(tauj);
            thd(j)=1/3*(2*dt*thdd(j)+4*thd(j-1)-thd(j-2));
            th(j)=dt^2*thdd(j)+2*th(j-1)-th(j-2);
            fx(j)=fxj;
            fy(j)=fyj;
            tau(j)=tauj;
        end
```

```
j12=j;

for x2=x(j12)+d*cos(th(j12)):dx:x(j12)+d*cos(th(j12))+2*d
    for x4=x(j12)-d*cos(th(j12)):dx:x(j12)-d*cos(th(j12))+2*d

        for j=j12+1:tt2(2)*n/100
            t(j)=t(j-1)+dt;
            [c1,c2,c3,c4]=c_i(tt1,tt2,tt3,tt4,t(j),T);
            [fxj,fyj] = Force(c1,c2,c3,c4,tt1,dfi,afx,t(j),T,...
                mi,m,g,fs);
            xdd(j)=1/m*(fxj);
            xd(j)=1/3*(2*dt*xdd(j)+4*xd(j-1)-xd(j-2));
            x(j)=dt^2*xdd(j)+2*x(j-1)-x(j-2);
            ydd(j)=1/m*(fyj-m*g);
            yd(j)=1/3*(2*dt*ydd(j)+4*yd(j-1)-yd(j-2));
            y(j)=dt^2*ydd(j)+2*y(j-1)-y(j-2);
            [tauj] = Torque(c1,c2,c3,c4,x1,x2,x3,x4,yg,x(j),...
                y(j),fxj,fyj);
            thdd(j)=1/I*(tauj);
            thd(j)=1/3*(2*dt*thdd(j)+4*thd(j-1)-thd(j-2));
            th(j)=dt^2*thdd(j)+2*th(j-1)-th(j-2);
            fx(j)=fxj;
            fy(j)=fyj;
            tau(j)=tauj;
        end

        i11=round(tt1(1)*n/100+1);
        i12=round(tt1(2)*n/100);
        i21=round(tt2(1)*n/100+1);
        i22=round(tt2(2)*n/100);
        i31=round(tt3(1)*n/100+1);
        i32=round(tt3(2)*n/100);
        i41=round(tt4(1)*n/100+1);
        i42=round(tt4(2)*n/100);

        I1=dt*trapz(abs(x1-x(i11:i12)+d*cos(th(i11:i12))));
        I2=dt*trapz(abs(x2-x(i21:i22)-d*cos(th(i21:i22))));
        I3=dt*trapz(abs(x3-x(i31:i32)-d*cos(th(i31:i32))));
        I4=dt*trapz(abs(x4-x(i41:i42)+d*cos(th(i41:i42))));

        % objective function
        fpth=abs(th(n)-th(1))*100+10*abs(thd(n)-thd(1))+...
            10*(I1+I2+I3+I4);
        % if the weighing factors or the form of the objective
        % function is changed here, it should also be changed in
        % ObjFun.m

        if fpth<minfpth
            minfpth=fpth;
            x1o=x1;
            x2o=x2;
            x3o=x3;
            x4o=x4;
            xo=x;
            yo=y;
            tho=th;
            tauo=tau;
            fxo=fx;
            fyo=fy;
```

```
                end
            end
        end
    end
end
%-------------------------------------------------------------------------%
%------------------------ FOOTFALL OPTIMISATION --------------------------%
% xff = [x1 x2 x3 x4]'
dx=0.05;
x1lb=x1o-dx;
x1ub=x1o+dx;
x2lb=x2o-dx;
x2ub=x2o+dx;
x3lb=x3o-dx;
x3ub=x3o+dx;
x4lb=x4o-dx;
x4ub=x4o+dx;

lb=[x1lb x2lb x3lb x4lb];
ub=[x1ub x2ub x3ub x4ub];

xff0=[x1o x2o x3o x4o];

opt=optimset('Display', 'off');
[xff,fval,exitflag]=fmincon(@(xff)ObjFun(xff,m,I,g,x0,vx,y0,yd0,th0,...
    thd0,yg,tt1,tt2,tt3,tt4,T,dfi,afx,mi,fs,d),xff0,[],[],[],[],lb,...
    ub,[],opt);

x1=xff(1);
x2=xff(2);
x3=xff(3);
x4=xff(4);

% check if upper or lower bounds truncate solutions
if x1<=xo(1)-d*cos(tho(1))||x1>=xo(1)-d*cos(tho(1))+2*d
    bfx1=1;
end
if x2<=xo(j12)+d*cos(tho(j12))||x2>=xo(j12)+d*cos(tho(j12))+2*d
    bfx2=1;
end
if x3<=xo(1)+d*cos(tho(1))||x3>=xo(1)+d*cos(tho(1))+2*d
    bfx3=1;
end
if x4<=xo(j12)-d*cos(tho(j12))||x4>=xo(j12)-d*cos(tho(j12))+2*d
    bfx4=1;
end

%-------------------------------------------------------------------------%
%--------------- CoM TRAJECTORY WITH OPTIMAL FOOTFALLS -------------------%
n=1000;
dt=T/(n-1);
t=linspace(0,T,n);
q0=[x0 vx y0 yd0 th0 thd0];
q=ode3(@(t,q)OdeFun(t,q,m,g,I,x1,x2,x3,x4,yg,tt1,tt2,tt3,tt4,T,dfi,...
    afx,mi,fs),t,q0);

x=q(:,1);
xd=q(:,2);
y=q(:,3);
yd=q(:,4);
```

```
th=q(:,5);
thd=q(:,6);
n=length(t);

fx=zeros(n,1);
fy=zeros(n,1);
tau=zeros(n,1);
xdd=zeros(n,1);
ydd=zeros(n,1);
thdd=zeros(n,1);

for j=1:n
    [c1,c2,c3,c4]=c_i(tt1,tt2,tt3,tt4,t(j),T);

    [fxj,fyj] = Force(c1,c2,c3,c4,tt1,dfi,afx,t(j),T,mi,m,g,fs);
    [tauj] = Torque(c1,c2,c3,c4,x1,x2,x3,x4,yg,x(j),y(j),fxj,fyj);

    fx(j)=fxj;
    fy(j)=fyj;
    tau(j)=tauj;

    xdd(j)=1/m*fx(j);
    ydd(j)=1/m*(fy(j)-m*g);
    thdd(j)=1/I*tau(j);
end
%-------------------------------------------------------------------------%
%------------------- MAXIMUM PERMISSIBLE PITCH ---------------------------%
[maxabsth,ithm]=max(abs(th));
if a/2*sin(maxabsth)+b/2*cos(maxabsth)<=(y(ithm)-yg)/2&&maxabsth<=15*pi/180

%----------------------- PARAMETRIC SEARCH STAGE 2 ----------------------%
%------------------ LEG PROPERTIES EXHAUSTIVE SEARCH --------------------%

% if flagl==0 then actuation constraints are not respected.
[l1,l20,k,flagl,flags,bfeps]=OptLegs(t,x,xd,xdd,y,yd,ydd,th,...
    thd,thdd,tt1,tt2,tt3,tt4,x1,x2,x3,x4,yg,fx,fy,d,g,...
    tau1maxst,tau1maxct,th1dmax,tau2maxst,tau2maxct,th2dmax,T);

%-------------------------------------------------------------------------%
%------------------------ SAVE RESULTS IN PSC.m --------------------------%
% if flagl==1 actuation constraints are respected
% if flags==0 strength constraints are respected
if flagl==1

    pt=length(l1);

    parameters=zeros(pt,13);

    parameters(1:pt,1)=vx;
    parameters(1:pt,2)=afx;
    parameters(1:pt,3)=T;
    parameters(1:pt,4)=y0;

    % check if upper or lower bounds exclude solutions
    if afx==afxub||afx==afxlb
        bfafx=1;
    end
    if T==Tub||T==Tlb
        bfT=1;
```

```
    end
    if y0==y0ub||y0==y0lb
        bfy0=1;
    end

    parameters(1:pt,8)=bfafx;
    parameters(1:pt,9)=bfT;
    parameters(1:pt,10)=bfy0;
    parameters(1:pt,11)=bfx1+bfx2+bfx3+bfx4;

    for pc=1:pt
        parameters(pc,5)=l1(pc);
        parameters(pc,6)=l20(pc);
        parameters(pc,7)=k(pc);
        parameters(pc,12)=bfeps(pc);
        parameters(pc,13)=flags(pc);
    end


    oc=oc+1;
    if oc==1
        PSC=parameters;
    else
        PSC=[PSC;parameters];
    end

    disp('Solution found')
else
    disp('Contstraints can not be satisfied')
end
%-------------------------------------------------------------------------%
end
%-------------------------------------------------------------------------%
disp('iii')
toc
end
disp('ii')
toc
end
disp('i')
toc
end
%-------------------------------------------------------------------------%

save('PSC_vx000.mat','PSC')
```

PostProcessing.m

```
%-------------------------------------------------------------------------%
%-------------------------- PostProcessing.m -----------------------------%
%-------------------------------------------------------------------------%
% Run the PostProcessing.m script after the Main_vx000.m and if at least a
% PSC.mat matrix is available. PostProcessing.m loads the set of optimal
% solutions located on the no-th row of the PSC.mat and calculates the CoM
% and joint trajectories, joint torques and stresses on the leg segments,
% data that are not saved in the PSC.mat matrix. At the end of the script
% an animation of the robot's movement is displayed, plots of the leg joint
% torques and angular velocities and plots of the stresses on the tubular
% leg segments.

clc
clear
tic

% load saved PSC matrix
load('PSC_vx000.mat')
no=1;

%--------------------------------- INPUTS ---------------------------------%
vx=PSC(no,1);
afx=PSC(no,2);
T=PSC(no,3);
y0=PSC(no,4);
l1=PSC(no,5);
l20=PSC(no,6);
k=PSC(no,7);
%-------------------------------------------------------------------------%
%------------------------- ROBOT PROPERTIES ------------------------------%
% The legs' properties are set in Torques.m
% The legs' strength is set in check_strength.m
m=42;
d=0.3;
g=9.81;
a=1;
b=0.15;
I=1/12*m*(a^2+b^2);
%-------------------------------------------------------------------------%
%------------------ GROUND FORCE RELATED PARAMETERS ----------------------%
fs=2;
mi=0.65;
%-------------------------------------------------------------------------%
%--------------------------- GROUND LOCATION -----------------------------%
% Set desired clearance from the ground in InvKinHL.m, InvKinFL.m
yg=0;
%-------------------------------------------------------------------------%
%----------------------- ACTUATOR SPECIFICATIONS -------------------------%
% Specify rotor inertias and reduction ratios in Torques.m
% The values below are torques and ang. velocities after reduction.
tau2maxst=45;
tau2maxct=14.96;
th2dmax=11.21;
tau1maxst=45;
tau1maxct=14.96;
th1dmax=11.21;
%-------------------------------------------------------------------------%
%----------------------- TEMPORAL GAIT PARAMETERS ------------------------%
% HL-->1, FL-->2, FR-->3, HR-->4, tti=[t_td_i%, t_to_i%]
```

```
tt1=[0,50];
tt2=[50,100];
tt3=[0,50];
tt4=[50,100];
t21=tt2(1);
t22=tt2(2);
dfi=-t21*pi/(t22-t21);
%-------------------------------------------------------------------------%
%---------------------- DESIRED INITIAL CONDITIONS ----------------------%
x0=1;
yd0=0;   % do not change yd0=0, method assumes appex as initial position.
th0=0;
thd0=0;
%-------------------------------------------------------------------------%
%---------------------- PARAMETRIC SEARCH STAGE 1 ----------------------%
%---------------------- FOOTFALL EXHAUSTIVE SEARCH ----------------------%
% initialization

dx=0.03;
n=100;
dt=T/(n-1);
minfpth=1e6;

t=zeros(n,1);
x=zeros(n,1);
xd=zeros(n,1);
xdd=zeros(n,1);
y=zeros(n,1);
yd=zeros(n,1);
ydd=zeros(n,1);
fx=zeros(n,1);
fy=zeros(n,1);
tau=zeros(n,1);
th=zeros(n,1);
thd=zeros(n,1);
thdd=zeros(n,1);



t(1)=0;
t(2)=dt;

x(1)=x0;
xd(1)=vx;
xd(2)=vx;
x(2)=x(1)+dt*xd(2);

y(1)=y0;
y(2)=y0;
yd(1)=yd0;
yd(2)=yd0;

th(1)=th0;
th(2)=th0;
thd(1)=thd0;
thd(2)=thd0;

% footfals xi exhaustive search
for x1=x(1)-d*cos(th(1)):dx:x(1)-d*cos(th(1))+2*d
    for x3=x(1)+d*cos(th(1)):dx:x(1)+d*cos(th(1))+2*d
```

```
x2=0;
x4=0;

for j=1:3
    [c1,c2,c3,c4]=c_i(tt1,tt2,tt3,tt4,t(j),T);
    [fxj,fyj] = Force(c1,c2,c3,c4,tt1,dfi,afx,t(j),T,mi,m,g,fs);
    [tauj] = Torque(c1,c2,c3,c4,x1,x2,x3,x4,yg,x(j),y(j),fxj,fyj);
    fx(j)=fxj;
    fy(j)=fyj;
    tau(j)=tauj;
end

for j=3:tt1(2)*n/100
    t(j)=t(j-1)+dt;
    [c1,c2,c3,c4]=c_i(tt1,tt2,tt3,tt4,t(j),T);
    [fxj,fyj] = Force(c1,c2,c3,c4,tt1,dfi,afx,t(j),T,mi,m,g,fs);
    xdd(j)=1/m*(fxj);
    xd(j)=1/3*(2*dt*xdd(j)+4*xd(j-1)-xd(j-2));
    x(j)=dt^2*xdd(j)+2*x(j-1)-x(j-2);
    ydd(j)=1/m*(fyj-m*g);
    yd(j)=1/3*(2*dt*ydd(j)+4*yd(j-1)-yd(j-2));
    y(j)=dt^2*ydd(j)+2*y(j-1)-y(j-2);
    [tauj] = Torque(c1,c2,c3,c4,x1,x2,x3,x4,yg,x(j),y(j),fxj,fyj);
    thdd(j)=1/I*(tauj);
    thd(j)=1/3*(2*dt*thdd(j)+4*thd(j-1)-thd(j-2));
    th(j)=dt^2*thdd(j)+2*th(j-1)-th(j-2);
    fx(j)=fxj;
    fy(j)=fyj;
    tau(j)=tauj;
end

j12=j;

for x2=x(j12)+d*cos(th(j12)):dx:x(j12)+d*cos(th(j12))+2*d
    for x4=x(j12)-d*cos(th(j12)):dx:x(j12)-d*cos(th(j12))+2*d

        for j=j12+1:tt2(2)*n/100
            t(j)=t(j-1)+dt;
            [c1,c2,c3,c4]=c_i(tt1,tt2,tt3,tt4,t(j),T);
            [fxj,fyj] = Force(c1,c2,c3,c4,tt1,dfi,afx,t(j),T,...
                mi,m,g,fs);
            xdd(j)=1/m*(fxj);
            xd(j)=1/3*(2*dt*xdd(j)+4*xd(j-1)-xd(j-2));
            x(j)=dt^2*xdd(j)+2*x(j-1)-x(j-2);
            ydd(j)=1/m*(fyj-m*g);
            yd(j)=1/3*(2*dt*ydd(j)+4*yd(j-1)-yd(j-2));
            y(j)=dt^2*ydd(j)+2*y(j-1)-y(j-2);
            [tauj] = Torque(c1,c2,c3,c4,x1,x2,x3,x4,yg,x(j),...
                y(j),fxj,fyj);
            thdd(j)=1/I*(tauj);
            thd(j)=1/3*(2*dt*thdd(j)+4*thd(j-1)-thd(j-2));
            th(j)=dt^2*thdd(j)+2*th(j-1)-th(j-2);
            fx(j)=fxj;
            fy(j)=fyj;
            tau(j)=tauj;
        end

        i11=round(tt1(1)*n/100+1);
        i12=round(tt1(2)*n/100);
```

```
                i21=round(tt2(1)*n/100+1);
                i22=round(tt2(2)*n/100);
                i31=round(tt3(1)*n/100+1);
                i32=round(tt3(2)*n/100);
                i41=round(tt4(1)*n/100+1);
                i42=round(tt4(2)*n/100);

                I1=dt*trapz(abs(x1-x(i11:i12)+d*cos(th(i11:i12))));
                I2=dt*trapz(abs(x2-x(i21:i22)-d*cos(th(i21:i22))));
                I3=dt*trapz(abs(x3-x(i31:i32)-d*cos(th(i31:i32))));
                I4=dt*trapz(abs(x4-x(i41:i42)+d*cos(th(i41:i42))));

                % objective function
                fpth=abs(th(n)-th(1))*100+10*abs(thd(n)-thd(1))+...
                    10*(I1+I2+I3+I4);

                if fpth<minfpth
                    minfpth=fpth;
                    x1o=x1;
                    x2o=x2;
                    x3o=x3;
                    x4o=x4;
                    xo=x;
                    yo=y;
                    tho=th;
                    tauo=tau;
                    fxo=fx;
                    fyo=fy;
                end
            end
        end
    end
end
%-------------------------------------------------------------------------%
%----------------------- FOOTFALL OPTIMISATION ---------------------------%
% xff = [x1 x2 x3 x4]'
dx=0.05;
x1lb=x1o-dx;
x1ub=x1o+dx;
x2lb=x2o-dx;
x2ub=x2o+dx;
x3lb=x3o-dx;
x3ub=x3o+dx;
x4lb=x4o-dx;
x4ub=x4o+dx;

lb=[x1lb x2lb x3lb x4lb];
ub=[x1ub x2ub x3ub x4ub];

xff0=[x1o x2o x3o x4o];

opt=optimset('Display', 'off');
[xff,fval,exitflag]=fmincon(@(xff)ObjFun(xff,m,I,g,x0,vx,y0,yd0,th0,...
    thd0,yg,tt1,tt2,tt3,tt4,T,dfi,afx,mi,fs,d),xff0,[],[],[],[],lb,...
    ub,[],opt);

x1=xff(1);
x2=xff(2);
x3=xff(3);
x4=xff(4);
```

```
%-------------------------------------------------------------------------%
%--------------- CoM TRAJECTORY WITH OPTIMAL FOOTFALLS -------------------%
n=1000;
dt=T/(n-1);
t=linspace(0,T,n);
q0=[x0 vx y0 yd0 th0 thd0];
q=ode3(@(t,q)OdeFun(t,q,m,g,I,x1,x2,x3,x4,yg,tt1,tt2,tt3,tt4,T,dfi,...
    afx,mi,fs),t,q0);

x=q(:,1);
xd=q(:,2);
y=q(:,3);
yd=q(:,4);
th=q(:,5);
thd=q(:,6);
n=length(t);

fx=zeros(n,1);
fy=zeros(n,1);
tau=zeros(n,1);
xdd=zeros(n,1);
ydd=zeros(n,1);
thdd=zeros(n,1);

for j=1:n
    [c1,c2,c3,c4]=c_i(tt1,tt2,tt3,tt4,t(j),T);

    [fxj,fyj] = Force(c1,c2,c3,c4,tt1,dfi,afx,t(j),T,mi,m,g,fs);
    [tauj] = Torque(c1,c2,c3,c4,x1,x2,x3,x4,yg,x(j),y(j),fxj,fyj);

    fx(j)=fxj;
    fy(j)=fyj;
    tau(j)=tauj;

    xdd(j)=1/m*fx(j);
    ydd(j)=1/m*(fy(j)-m*g);
    thdd(j)=1/I*tau(j);
end
%-------------------------------------------------------------------------%
%------------------- MAXIMUM PERMISSIBLE PITCH ---------------------------%
[maxabsth,ithm]=max(abs(th));
if a/2*sin(maxabsth)+b/2*cos(maxabsth)<=(y(ithm)-yg)/2&&maxabsth<=15*pi/180

%---------------------- LEG JOINT TRAJECTORIES ---------------------------%
% calculate joint trajectories from Inverse Kinematics
l1h1=l1;
l20h1=l20;
kh1=k;
leg=1;
[qjh1,qjdh1,qjddh1,flag1] = InvKinH(l1h1,l20h1,kh1,t,x,y,th,thd,thdd,...
    tt1,x1,yg,fx,fy,d,leg);

l1h4=l1;
l20h4=l20;
kh4=k;
leg=4;
[qjh4,qjdh4,qjddh4,flag4] = InvKinH(l1h4,l20h4,kh4,t,x,y,th,thd,thdd,...
    tt4,x4,yg,fx,fy,d,leg);
```

```
l1f2=l1;
l20f2=l20;
kf2=k;
leg=2;
[qjf2,qjdf2,qjddf2,flag2] = InvKinF(l1f2,l20f2,kf2,t,x,y,th,thd,thdd,...
    tt2,x2,yg,fx,fy,d,leg);

l1f3=l1;
l20f3=l20;
kf3=k;
leg=3;
[qjf3,qjdf3,qjddf3,flag3] = InvKinF(l1f3,l20f3,kf3,t,x,y,th,thd,thdd,...
    tt3,x3,yg,fx,fy,d,leg);

% if leg collisions are avoided and toe remains in the workspace of the leg
% then flagi=0
if flag1==0&&flag2==0&&flag3==0&&flag4==0

% leg joint trajectories calculation
thh1=qjh1(:,1);
th1h1=qjh1(:,2);
th2h1=qjh1(:,3);
l2h1=qjh1(:,4);
thdh1=qjdh1(:,1);
th1dh1=qjdh1(:,2);
th2dh1=qjdh1(:,3);
l2dh1=qjdh1(:,4);
thddh1=qjddh1(:,1);
th1ddh1=qjddh1(:,2);
th2ddh1=qjddh1(:,3);
l2ddh1=qjddh1(:,4);

thf2=qjf2(:,1);
th1f2=qjf2(:,2);
th2f2=qjf2(:,3);
l2f2=qjf2(:,4);
thdf2=qjdf2(:,1);
th1df2=qjdf2(:,2);
th2df2=qjdf2(:,3);
l2df2=qjdf2(:,4);
thddf2=qjddf2(:,1);
th1ddf2=qjddf2(:,2);
th2ddf2=qjddf2(:,3);
l2ddf2=qjddf2(:,4);

thf3=qjf3(:,1);
th1f3=qjf3(:,2);
th2f3=qjf3(:,3);
l2f3=qjf3(:,4);
thdf3=qjdf3(:,1);
th1df3=qjdf3(:,2);
th2df3=qjdf3(:,3);
l2df3=qjdf3(:,4);
thddf3=qjddf3(:,1);
th1ddf3=qjddf3(:,2);
th2ddf3=qjddf3(:,3);
l2ddf3=qjddf3(:,4);

thh4=qjh4(:,1);
th1h4=qjh4(:,2);
```

```
th2h4=qjh4(:,3);
l2h4=qjh4(:,4);
thdh4=qjdh4(:,1);
th1dh4=qjdh4(:,2);
th2dh4=qjdh4(:,3);
l2dh4=qjdh4(:,4);
thddh4=qjddh4(:,1);
th1ddh4=qjddh4(:,2);
th2ddh4=qjddh4(:,3);
l2ddh4=qjddh4(:,4);

% leg joint torque calculation
[torque1h1,torque2h1,torque1h4,torque2h4,torque1f2,torque2f2,...
torque1f3,torque2f3,flags] = Torques(xd,xdd,yd,ydd,th,thd,thdd,th1h1,...
    th1dh1,th1ddh1,l1h1,th2h1,th2dh1,th2ddh1,l2h1,l2dh1,l2ddh1,th1h4,...
    th1dh4,th1ddh4,l1h4,th2h4,th2dh4,th2ddh4,l2h4,l2dh4,l2ddh4,th1f2,...
    th1df2,th1ddf2,l1f2,th2f2,th2df2,th2ddf2,l2f2,l2df2,l2ddf2,th1f3,...
    th1df3,th1ddf3,l1f3,th2f3,th2df3,th2ddf3,l2f3,l2df3,l2ddf3,g,d,...
    tt1,tt2,tt3,tt4,t,T,fx,fy);

trq11rms=rms(torque1h1);
trq21rms=rms(torque2h1);
trq13rms=rms(torque1f3);
trq23rms=rms(torque2f3);

trq14rms=rms(torque1h4);
trq24rms=rms(torque2h4);
trq12rms=rms(torque1f2);
trq22rms=rms(torque2f2);

% stress calculation
[din,dout,flag] = check_strength(fx,fy,th,th1h1,l1h1,th2h1,l2h1...
    ,th1f2,l1f2,th2f2,l2f2,th1f3,l1f3,th2f3,l2f3,th1h4,l1h4,th2h4,l2h4);
%--------------------------------------------------------------------------%
%------------------------------ ANIMATIONS --------------------------------%
%Animation % centroidal expression
AnimationFull % full robot
%--------------------------------------------------------------------------%
%------------------- JOINT TORQUE/ANG. VELOCITY PLOTS ---------------------%
PlotTorquesNRadPerSec
%--------------------------------------------------------------------------%
%----------------------- LEG SEGMENT STRESS PLOTS -------------------------%
PlotStress
%--------------------------------------------------------------------------%
end
end
%--------------------------------------------------------------------------%
toc
```

c_i.m

```
function [c1,c2,c3,c4] = c_i(tt1,tt2,tt3,tt4,t,T)
%C_I produces the gait graph constants c1, c2, c3 ,c4. The input vectors
%tti include the touchdown and take off times as a percentage of the stride
%cyrcle. For instance tt1=[0,25] means that the leg 1 touches the ground at
%the 0% of the cyrcle and takes off at 25% of the stride cyrcle.

%------------- LEG 1 --------------%
if length(tt1)==2
    t11=tt1(1)/100*T;
    t12=tt1(2)/100*T;
    if t>=t11&&t<=t12
        c1=1;
    else
        c1=0;
    end
end

%------------- LEG 2 --------------%
if length(tt2)==2
    t21=tt2(1)/100*T;
    t22=tt2(2)/100*T;
    if t>=t21&&t<=t22
        c2=1;
    else
        c2=0;
    end
end

%------------- LEG 3 --------------%
if length(tt3)==2
    t31=tt3(1)/100*T;
    t32=tt3(2)/100*T;
    if t>=t31&&t<=t32
        c3=1;
    else
        c3=0;
    end
end

%------------- LEG 4 --------------%
if length(tt4)==2
    t41=tt4(1)/100*T;
    t42=tt4(2)/100*T;
    if t>=t41&&t<=t42
        c4=1;
    else
        c4=0;
    end
end

end
```

Force.m

```
function [fxj,fyj] = Force(c1,c2,c3,c4,tt1,dfi,afx,tj,T,mi,m,g,fs)
%FORCE takes as input the gait parameters ci, the times of touch down and
%lift off tt1 of the first leg as a percentage of the gait period T, the
%footfall phase difference dfi, the horizontal force size factor afx,
%the current time tj, coulomb friction coeefficient mi, the mass of the
%robot m, gravitational acceleration g and horizontal force scenario fs
%(fs=1 or fs=2) and returns the forces fxj, fyj exerted to the CoM of the
%legged robot.

if length(tt1)==2
    t11=tt1(1)/100*T;
    t12=tt1(2)/100*T;
    wy=pi/(t12-t11);
    fi1y=-wy*t11;
    dfl=(t12-t11)/T;
    fymax=pi*m*g/8/dfl;
    dfiy=dfi;
    fy1=fymax*sin(wy*tj+fi1y);
    fy2=fymax*sin(wy*tj+fi1y+dfiy);
    fy3=fymax*sin(wy*tj+fi1y+2*dfiy);
    fy4=fymax*sin(wy*tj+fi1y+3*dfiy);
    fyj=c1*fy1+c2*fy2+c3*fy3+c4*fy4;

    if fs==1
        % CASE 1: fxi is a sinusoidal impulse
        fx1=afx*mi*fy1;
        fx2=-afx*mi*fy2;
        fx3=-afx*mi*fy3;
        fx4=afx*mi*fy4;
        fxj=c1*fx1+c2*fx2+c3*fx3+c4*fx4;
    end
    if fs==2
        % CASE 2: fxi is sinusoidal decelerating-accelerating
        wx=2*wy;
        fi1x=pi-wx*t11;
        dfix=2*dfiy;
        fxub=mi*fymax*sin(pi/8+3*pi/80);
        fxmax=afx*fxub;
        fx1=fxmax*sin(wx*tj+fi1x);
        fx2=fxmax*sin(wx*tj+fi1x+dfix);
        fx3=fxmax*sin(wx*tj+fi1x+2*dfix);
        fx4=fxmax*sin(wx*tj+fi1x+3*dfix);
        fxj=c1*fx1+c2*fx2+c3*fx3+c4*fx4;
    end
end

end
```

Torque.m

```
function [tauj] = Torque(c1,c2,c3,c4,x1,x2,x3,x4,yg,xj,yj,fxj,fyj)
%TORQUE takes as input the gait parameters ci, the coordinates of the
%footfalls (xi,yg),position xj, yj of the CoM, and current exerted forces
%from the ground fxj, fyj and produces the total torque exerted on the
%robot w.r.t CoM.

    tau1=(x1-xj)*fyj-(yg-yj)*fxj;
    tau2=(x2-xj)*fyj-(yg-yj)*fxj;
    tau3=(x3-xj)*fyj-(yg-yj)*fxj;
    tau4=(x4-xj)*fyj-(yg-yj)*fxj;
    tauj=c1*tau1+c2*tau2+c3*tau3+c4*tau4;

end
```

ObjFun.m

```
function fpth = ObjFun(xff,m,I,g,x0,vx,y0,yd0,th0,thd0,...
    yg,tt1,tt2,tt3,tt4,T,dfi,afx,mi,fs,d)
%OBJFUN calculates the objective function value for the optimization of the
%footfalls xi and the centroidal CoM trajectory.

x1=xff(1);
x2=xff(2);
x3=xff(3);
x4=xff(4);

tspan=[0 T];
q0=[x0 vx y0 yd0 th0 thd0];
options=odeset('AbsTol',1e-6);
[t,q]=ode45(@(t,q)OdeFun(t,q,m,g,I,x1,x2,x3,x4,yg,tt1,tt2,tt3,tt4,T,...
    dfi,afx,mi,fs),tspan,q0,options);

x=q(:,1);
y=q(:,3);
th=q(:,5);
thd=q(:,6);
n=length(t);
dt=T/(n-1);

fx=zeros(n,1);
fy=zeros(n,1);
tau=zeros(n,1);
for j=1:n
    [c1,c2,c3,c4]=c_i(tt1,tt2,tt3,tt4,t(j),T);
    [fxj,fyj] = Force(c1,c2,c3,c4,tt1,dfi,afx,t(j),T,mi,m,g,fs);
    [tauj] = Torque(c1,c2,c3,c4,x1,x2,x3,x4,yg,x(j),y(j),fxj,fyj);

    fx(j)=fxj;
    fy(j)=fyj;
    tau(j)=tauj;
end

i11=round(tt1(1)*n/100+1);
i12=round(tt1(2)*n/100);
i21=round(tt2(1)*n/100+1);
i22=round(tt2(2)*n/100);
```

```
i31=round(tt3(1)*n/100+1);
i32=round(tt3(2)*n/100);
i41=round(tt4(1)*n/100+1);
i42=round(tt4(2)*n/100);

I1=dt*trapz(abs(x1-x(i11:i12)+d*cos(th(i11:i12))));
I2=dt*trapz(abs(x2-x(i21:i22)-d*cos(th(i21:i22))));
I3=dt*trapz(abs(x3-x(i31:i32)-d*cos(th(i31:i32))));
I4=dt*trapz(abs(x4-x(i41:i42)+d*cos(th(i41:i42))));

fpth=abs(th(n)-th(1))*100+10*abs(thd(n)-thd(1))+10*(I1+I2+I3+I4);

end
```

OdeFun.m

```
function qd = OdeFun(t,q,m,g,I,x1,x2,x3,x4,yg,tt1,tt2,tt3,tt4,T,dfi,...
    afx,mi,fs)
%ODEFUN provides the rhs of the Centroidal Dynamics EoM


    [c1,c2,c3,c4]=c_i(tt1,tt2,tt3,tt4,t,T);

    [fxj,fyj] = Force(c1,c2,c3,c4,tt1,dfi,afx,t,T,mi,m,g,fs);

    x=q(1);
    y=q(3);

    [tauj] = Torque(c1,c2,c3,c4,x1,x2,x3,x4,yg,x,y,fxj,fyj);

    qd(1,1)=q(2);
    qd(2,1)=1/m*(fxj);
    qd(3,1)=q(4);
    qd(4,1)=1/m*(fyj-m*g);
    qd(5,1)=q(6);
    qd(6,1)=1/I*(tauj);

end
```

OptLegs.m

```
function [l1o,l20o,ko,flag1,flagsm,bfeps]
=OptLegs(t,x,xd,xdd,y,yd,ydd,th,...
    thd,thdd,tt1,tt2,tt3,tt4,x1,x2,x3,x4,yg,fx,fy,d,g,...
    tau1maxst,tau1maxct,th1dmax,tau2maxst,tau2maxct,th2dmax,T)
%OPTLEGS finds all permissible leg segment lengths and spring stiffnesses
%for which the actuation constraints are satisfied.

n=length(x);

xh1=x(tt1(1)*n/100+1:tt1(2)*n/100)-x1;
yh1=y(tt1(1)*n/100+1:tt1(2)*n/100)-yg;
thh1=th(tt1(1)*n/100+1:tt1(2)*n/100);
nh1=length(xh1);
xsh1=xh1(1:nh1)-d*cos(thh1(1:nh1));
```

```
ysh1=yh1(1:nh1)-d*sin(thh1(1:nh1));
lh2t_h1=sqrt(xsh1.^2+ysh1.^2);

xf2=x(tt2(1)*n/100+1:tt2(2)*n/100)-x2;
yf2=y(tt2(1)*n/100+1:tt2(2)*n/100)-yg;
thf2=th(tt2(1)*n/100+1:tt2(2)*n/100);
nf2=length(xf2);
xsf2=xf2(1:nf2)+d*cos(thf2(1:nf2));
ysf2=yf2(1:nf2)+d*sin(thf2(1:nf2));
lh2t_f2=sqrt(xsf2.^2+ysf2.^2);

xf3=x(tt3(1)*n/100+1:tt3(2)*n/100)-x3;
yf3=y(tt3(1)*n/100+1:tt3(2)*n/100)-yg;
thf3=th(tt3(1)*n/100+1:tt3(2)*n/100);
nf3=length(xf3);
xsf3=xf3(1:nf3)+d*cos(thf3(1:nf3));
ysf3=yf3(1:nf3)+d*sin(thf3(1:nf3));
lh2t_f3=sqrt(xsf3.^2+ysf3.^2);

xh4=x(tt4(1)*n/100+1:tt4(2)*n/100)-x4;
yh4=y(tt4(1)*n/100+1:tt4(2)*n/100)-yg;
thh4=th(tt4(1)*n/100+1:tt4(2)*n/100);
nh4=length(xh4);
xsh4=xh4(1:nh4)-d*cos(thh4(1:nh4));
ysh4=yh4(1:nh4)-d*sin(thh4(1:nh4));
lh2t_h4=sqrt(xsh4.^2+ysh4.^2);

l_min=max([max(lh2t_f2),max(lh2t_f3),max(lh2t_h1),max(lh2t_h4)]);

flagl=0;
ic=0;
epsub=1.1;

for k=1250:1250:5000
    for eps=1:0.025:epsub
        for l1=0.1:0.01:l_min*eps-0.1
            l20=(l_min*eps-l1)/1;

            [qjh1,qjdh1,qjddh1,flag1] =
InvKinH(l1,l20,k,t,x,y,th,thd,thdd,...
    tt1,x1,yg,fx,fy,d,1);
            [qjh4,qjdh4,qjddh4,flag4] =
InvKinH(l1,l20,k,t,x,y,th,thd,thdd,...
    tt4,x4,yg,fx,fy,d,4);
            [qjf2,qjdf2,qjddf2,flag2] =
InvKinF(l1,l20,k,t,x,y,th,thd,thdd,...
    tt2,x2,yg,fx,fy,d,2);
            [qjf3,qjdf3,qjddf3,flag3] =
InvKinF(l1,l20,k,t,x,y,th,thd,thdd,...
    tt3,x3,yg,fx,fy,d,3);

            if flag1==0&&flag2==0&&flag3==0&&flag4==0

                l1h1=l1;
                th1h1=qjh1(:,2);
                th2h1=qjh1(:,3);
                l2h1=qjh1(:,4);
                th1dh1=qjdh1(:,2);
                th2dh1=qjdh1(:,3);
```

```
l2dh1=qjdh1(:,4);
th1ddh1=qjddh1(:,2);
th2ddh1=qjddh1(:,3);
l2ddh1=qjddh1(:,4);

l1f2=l1;
th1f2=qjf2(:,2);
th2f2=qjf2(:,3);
l2f2=qjf2(:,4);
th1df2=qjdf2(:,2);
th2df2=qjdf2(:,3);
l2df2=qjdf2(:,4);
th1ddf2=qjddf2(:,2);
th2ddf2=qjddf2(:,3);
l2ddf2=qjddf2(:,4);

l1f3=l1;
th1f3=qjf3(:,2);
th2f3=qjf3(:,3);
l2f3=qjf3(:,4);
th1df3=qjdf3(:,2);
th2df3=qjdf3(:,3);
l2df3=qjdf3(:,4);
th1ddf3=qjddf3(:,2);
th2ddf3=qjddf3(:,3);
l2ddf3=qjddf3(:,4);

l1h4=l1;
th1h4=qjh4(:,2);
th2h4=qjh4(:,3);
l2h4=qjh4(:,4);
th1dh4=qjdh4(:,2);
th2dh4=qjdh4(:,3);
l2dh4=qjdh4(:,4);
th1ddh4=qjddh4(:,2);
th2ddh4=qjddh4(:,3);
l2ddh4=qjddh4(:,4);


[torque1h1,torque2h1,torque1h4,torque2h4,torque1f2,torque2f2,...
                    torque1f3,torque2f3,flags] =
Torques(xd,xdd,yd,ydd,th,thd,thdd,th1h1,...
th1dh1,th1ddh1,l1h1,th2h1,th2dh1,th2ddh1,l2h1,l2dh1,l2ddh1,th1h4,...
th1dh4,th1ddh4,l1h4,th2h4,th2dh4,th2ddh4,l2h4,l2dh4,l2ddh4,th1f2,...
th1df2,th1ddf2,l1f2,th2f2,th2df2,th2ddf2,l2f2,l2df2,l2ddf2,th1f3,...
th1df3,th1ddf3,l1f3,th2f3,th2df3,th2ddf3,l2f3,l2df3,l2ddf3,g,d,...
                    tt1,tt2,tt3,tt4,t,T,fx,fy);

ct1f2st=max(abs(torque1f2))/tau1maxst;
ct1f2ct=rms(torque1f2)/tau1maxct;
cth1df2=max(abs(th1df2))/th1dmax;
ct2f2st=max(abs(torque2f2))/tau2maxst;
ct2f2ct=rms(torque2f2)/tau2maxct;
cth2df2=max(abs(th2df2))/th2dmax;

ct1f3st=max(abs(torque1f3))/tau1maxst;
ct1f3ct=rms(torque1f3)/tau1maxct;
cth1df3=max(abs(th1df3))/th1dmax;
ct2f3st=max(abs(torque2f3))/tau2maxst;
ct2f3ct=rms(torque2f3)/tau2maxct;
```

```
                cth2df3=max(abs(th2df3))/th2dmax;

                if
ct1f2st<=1&&cth1df2<=1&&ct2f2st<=1&&cth2df2<=1&&ct1f3st<=1&&...
                        cth1df3<=1&&ct2f3st<=1&&cth2df3<=1&&ct1f2ct<=1&&...
                        ct2f2ct<=1&&ct1f3ct<=1&&ct2f3ct<=1
                    acsf=1;
                else
                    acsf=0;
                end

                ct1h1st=max(abs(torque1h1))/tau1maxst;
                ct1h1ct=rms(torque1h1)/tau1maxct;
                cth1dh1=max(abs(th1dh1))/th1dmax;
                ct2h1st=max(abs(torque2h1))/tau2maxst;
                ct2h1ct=rms(torque2h1)/tau2maxct;
                cth2dh1=max(abs(th2dh1))/th2dmax;

                ct1h4st=max(abs(torque1h4))/tau1maxst;
                ct1h4ct=rms(torque1h4)/tau1maxct;
                cth1dh4=max(abs(th1dh4))/th1dmax;
                ct2h4st=max(abs(torque2h4))/tau2maxst;
                ct2h4ct=rms(torque2h4)/tau2maxct;
                cth2dh4=max(abs(th2dh4))/th2dmax;

                if
ct1h1st<=1&&cth1dh1<=1&&ct2h1st<=1&&cth2dh1<=1&&ct1h4st<=1&&...
                        cth1dh4<=1&&ct2h4st<=1&&cth2dh4<=1&&ct1h1ct<=1&&...
                        ct2h1ct<=1&&ct1h4ct<=1&&ct2h4ct<=1
                    acsh=1;
                else
                    acsh=0;
                end

                if acsf==1&&acsh==1
                    ic=ic+1;
                    l1o(ic,1)=l1;
                    l20o(ic,1)=l20;
                    ko(ic,1)=k;
                    flagsm(ic,1)=flags;
                    flagl=1;

                    if eps==1||eps==epsub
                        bfeps(ic,1)=1;
                    else
                        bfeps(ic,1)=0;
                    end

                end
            end
        end
    end
end

if flagl==0
    l1o=0;
    l20o=0;
    ko=0;
    bfeps=0;
    flagsm=5;
```

```
end

end
```

InvKinH.m

```
function [qjh,qjdh,qjddh,flag] = InvKinH(l1h,l20h,kh,t,x,y,th,thd,thdd,...
    tti,xi,yg,fx,fy,d,leg)
%InvKinH Calculates the joint trajectories of a hind leg given the link
%lengths l1h, l20h and distal limb stiffness kh and the body trajectory x,
%y, th, the ground forces fx, fy, the half hip separation distance d and
%the leg {1,4}.

n=length(x);
T=t(n);
dt=t(2)-t(1);
xh=x-xi;
yh=y-yg;
thh=th;
thdh=thd;
thddh=thdd;
fxh=fx/2;
fyh=fy/2;

xsh=xh-d*cos(thh);
ysh=yh-d*sin(thh);
th1h=zeros(n,1);
th2h=zeros(n,1);
l2h=zeros(n,1);

xhh=x-d*cos(th);
yhh=y-d*sin(th);

sng=0; % if sng=1 toe is out of leg's workspace
clsn=0;
flag=0;

l2h(tti(1)*n/100+1)=l20h;
%------------------- ELLIPTICAL TRAJECTORY PROPERTIES --------------------%
if leg==1
    atr1=(max(x)-min(x))/2;
    btr1=0.05; %change clearance value here
    xc1=xi+atr1;
    yc1=yg;
    wtr1=-pi/(T-tti(2)/100*T);
    fitr1=pi*T/(T-tti(2)/100*T);
    xtr1=transpose(xc1+atr1*cos(wtr1.*t+fitr1));
    ytr1=transpose(yc1+btr1*sin(wtr1.*t+fitr1));
    xssph=xhh-xtr1;
    yssph=yhh-ytr1;
end
if leg==4
    atr4=(max(x)-min(x))/2;
    btr4=0.05; %change clearance value here
    xc4=xi-atr4;
    yc4=yg;
    wtr4=-pi/(tti(1)/100*T-0);
```

```
    fitr4=pi;
    xtr4=transpose(xc4+atr4*cos(wtr4.*t+fitr4));
    ytr4=transpose(yc4+btr4*sin(wtr4.*t+fitr4));
    xssph=xhh-xtr4;
    yssph=yhh-ytr4;
end
%-------------------------------------------------------------------------%
for j=1:n
%--------------- STANCE PHASE INVERSE KINEMATICS (ELLIPSE) ---------------%
    if j>tti(1)*n/100&&j<=tti(2)*n/100
        ah=(l1h^2-xsh(j)^2-ysh(j)^2-l2h(j)^2)/2/l2h(j);

        if xsh(j)^2+ysh(j)^2-ah^2>0
            th2h(j)=2*atan2(-xsh(j)-sqrt(xsh(j)^2+ysh(j)^2-ah^2),ysh(j)-...
                ah)-thh(j); %change sign before sqrt to change configurati-
            %on: '-' corresponds to KF, '+' corresponds to KB

            th2h(j)=mod(th2h(j),2*pi);
            th2h(j)=th2h(j)-(th2h(j)>3*pi/2)*2*pi;

            th1h(j)=atan2(-xsh(j)-l2h(j)*sin(thh(j)+th2h(j)),ysh(j)-...
                l2h(j)*cos(thh(j)+th2h(j)))-thh(j);

            if j<tti(2)*n/100
                l2h(j+1)=l20h+(fxh(j)*sin(thh(j)+th2h(j))-fyh(j)*...
                    cos(thh(j)+th2h(j)))/kh;
            end
        else
            sng=1;
        end
%-------------------------------------------------------------------------%
    else
%-------------------- FLIGHT PHASE INVERSE KINEMATICS --------------------%
        l2h(j)=l20h;
        ar=(l1h^2-xssph(j)^2-yssph(j)^2-l2h(j)^2)/2/l2h(j);
        if xssph(j)^2+yssph(j)^2-ar^2>0
            th2h(j)=2*atan2(-xssph(j)-sqrt(xssph(j)^2+yssph(j)^2-ar^2),...
                yssph(j)-ar)-th(j); %change sign before sqrt to change
            %configuration: '-' corresponds to KF, '+' corresponds to KB
            th2h(j)=mod(th2h(j),2*pi);
            th2h(j)=th2h(j)-(th2h(j)>3*pi/2)*2*pi;

            th1h(j)=atan2(-xssph(j)-l2h(j)*sin(th(j)+th2h(j)),yssph(j)-...
                l2h(j)*cos(th(j)+th2h(j)))-th(j);
        else
            sng=1;
        end
%-------------------------------------------------------------------------%
    end
end


%------------ CHECK COLISION OF THE KNEE JOINT WITH THE GROUND -----------%
if min(th2h+thh+pi/2)<5*pi/180
    clsn=1;
end
%-------------------------------------------------------------------------%
if sng==0&&clsn==0
%--------------------- ACCUMULATE JOINT TRAJECTORIES ---------------------%
    th2dh=[gradient(th2h(1:500),dt);gradient(th2h(501:1000),dt)];
```

```
    th1dh=[gradient(th1h(1:500),dt);gradient(th1h(501:1000),dt)];
    l2dh=[gradient(l2h(1:500),dt);gradient(l2h(501:1000),dt)];
    th1dh(1:3)=interp1([4 5 6 7 8],th1dh(4:8),[1 2 3],'pchip');
    th2dh(1:3)=interp1([4 5 6 7 8],th2dh(4:8),[1 2 3],'pchip');
    l2dh(1:3)=interp1([4 5 6 7 8],l2dh(4:8),[1 2 3],'pchip');
    th1dh(501:503)=interp1(504:508,th1dh(504:508),501:503,'pchip');
    th2dh(501:503)=interp1(504:508,th2dh(504:508),501:503,'pchip');
    l2dh(501:503)=interp1(504:508,l2dh(504:508),501:503,'pchip');
    l2ddh=4*[del2(l2h(1:500),t(1:500));del2(l2h(501:1000),t(501:1000))];
    l2ddh(1:3)=interp1([4 5 6 7 8],l2ddh(4:8),[1 2 3],'pchip');
    l2ddh(501:503)=interp1(504:508,l2ddh(504:508),501:503,'pchip');
%-------------------------------------------------------------------------%
%---------------------- CUBIC POLYNOMIAL CORRECTION ----------------------%
% after cubic correction the toe trajectory is hybrid cubic/elliptic.
    [th1c,th2c,th1dc,th2dc,th1ddc,th2ddc] = cubic_correction(t,th1h,...
    th2h,th1dh,th2dh,leg);

    qjh=[thh,th1c,th2c,l2h];
    qjdh=[thdh,th1dc,th2dc,l2dh];
    qjddh=[thddh,th1ddc,th2ddc,l2ddh];
%-------------------------------------------------------------------------%
else
    flag=1;
    th1dh=zeros(n,1);
    th2dh=zeros(n,1);
    l2dh=zeros(n,1);
    th1ddh=zeros(n,1);
    th2ddh=zeros(n,1);
    l2ddh=zeros(n,1);
    qjh=[thh,th1h,th2h,l2h];
    qjdh=[thdh,th1dh,th2dh,l2dh];
    qjddh=[thddh,th1ddh,th2ddh,l2ddh];
end

end
```

InvKinF.m

```
function [qjf,qjdf,qjddf,flag] = InvKinF(l1f,l20f,kf,t,x,y,th,thd,thdd,...
    tti,xi,yg,fx,fy,d,leg)
%INVKINF Calculates the joint trajectory of the front leg given the link
%lengths l1f, l20f and distal limb stiffness kf and the body trajectory x,
%y, th, the ground forces fx, fy, the half hip separation distance d and
%the leg {2,3}.

n=length(x);
T=t(n);
dt=t(2)-t(1);
xf=x-xi;
yf=y-yg;
thf=th;
thdf=thd;
thddf=thdd;
fxf=fx/2;
fyf=fy/2;

xsf=xf+d*cos(thf);
ysf=yf+d*sin(thf);
```

```
th1f=zeros(n,1);
th2f=zeros(n,1);
l2f=zeros(n,1);

xhf=x+d*cos(th);
yhf=y+d*sin(th);

sng=0; % if sng=1 toe is out of leg's workspace
clsn=0;
flag=0;

l2f(tti(1)*n/100+1)=l20f;
%------------------- ELLIPTICAL TRAJECTORY PROPERTIES --------------------%
if leg==2
    atr2=(max(x)-min(x))/2;
    btr2=0.05; %change clearance value here
    xc2=xi-atr2;
    yc2=yg;
    wtr2=-pi/(tti(1)/100*T-0);
    fitr2=pi;
    xtr2=transpose(xc2+atr2*cos(wtr2.*t+fitr2));
    ytr2=transpose(yc2+btr2*sin(wtr2.*t+fitr2));
    xsspf=xhf-xtr2;
    ysspf=yhf-ytr2;
end
if leg==3
    atr3=(max(x)-min(x))/2;
    btr3=0.05; %change clearance value here
    xc3=xi+atr3;
    yc3=yg;
    wtr3=-pi/(T-tti(2)/100*T);
    fitr3=pi*T/(T-tti(2)/100*T);
    xtr3=transpose(xc3+atr3*cos(wtr3.*t+fitr3));
    ytr3=transpose(yc3+btr3*sin(wtr3.*t+fitr3));
    xsspf=xhf-xtr3;
    ysspf=yhf-ytr3;
end
%-------------------------------------------------------------------------%
for j=1:n
%-------------------- STANCE PHASE INVERSE KINEMATICS --------------------%
    if j>tti(1)*n/100&&j<=tti(2)*n/100

        af=(l1f^2-xsf(j)^2-ysf(j)^2-l2f(j)^2)/2/l2f(j);

        if xsf(j)^2+ysf(j)^2-af^2>0
            th2f(j)=2*atan2(-xsf(j)-sqrt(xsf(j)^2+ysf(j)^2-af^2),...
                ysf(j)-af)-thf(j); %change sign before sqrt to change
configurati-
            %on: '-' corresponds to KF, '+' corresponds to KB
            th2f(j)=mod(th2f(j),2*pi);
            th2f(j)=th2f(j)-(th2f(j)>3*pi/2)*2*pi;
            th1f(j)=atan2(-xsf(j)-l2f(j)*sin(thf(j)+th2f(j)),ysf(j)-...
                l2f(j)*cos(thf(j)+th2f(j)))-thf(j);

            if j<tti(2)*n/100
                l2f(j+1)=l20f+(fxf(j)*sin(thf(j)+th2f(j))-fyf(j)*...
                    cos(thf(j)+th2f(j)))/kf;
            end
        else
            sng=1;
```

```
        end
%-------------------------------------------------------------------------%
    else
%--------------- FLIGHT PHASE INVERSE KINEMATICS (ELLIPSE) ---------------%
        l2f(j)=l20f;
        ar=(l1f^2-xsspf(j)^2-ysspf(j)^2-l2f(j)^2)/2/l2f(j);
        if xsspf(j)^2+ysspf(j)^2-ar^2>0
            th2f(j)=2*atan2(-xsspf(j)-sqrt(xsspf(j)^2+ysspf(j)^2-ar^2),...
                ysspf(j)-ar)-th(j); %change sign before sqrt to change
            %configuration: '-' corresponds to KF, '+' corresponds to KB
            th2f(j)=mod(th2f(j),2*pi);
            th2f(j)=th2f(j)-(th2f(j)>3*pi/2)*2*pi;

            th1f(j)=atan2(-xsspf(j)-l2f(j)*sin(thf(j)+th2f(j)),ysspf(j)-...
                l2f(j)*cos(thf(j)+th2f(j)))-thf(j);
        else
            sng=1;
        end
%-------------------------------------------------------------------------%
    end
end
%------------ CHECK COLISION OF THE KNEE JOINT WITH THE GROUND -----------%
if min(th2f+thf+pi/2)<5*pi/180
    clsn=1;
end
%-------------------------------------------------------------------------%
if sng==0&&clsn==0
%--------------------- ACCUMULATE JOINT TRAJECTORIES ---------------------%
    th2df=[gradient(th2f(1:500),dt);gradient(th2f(501:1000),dt)];
    th1df=[gradient(th1f(1:500),dt);gradient(th1f(501:1000),dt)];
    l2df=[gradient(l2f(1:500),dt);gradient(l2f(501:1000),dt)];
    th1df(1:3)=interp1([4 5 6 7 8],th1df(4:8),[1 2 3],'pchip');
    th2df(1:3)=interp1([4 5 6 7 8],th2df(4:8),[1 2 3],'pchip');
    l2df(1:3)=interp1([4 5 6 7 8],l2df(4:8),[1 2 3],'pchip');
    th1df(501:503)=interp1(504:508,th1df(504:508),501:503,'pchip');
    th2df(501:503)=interp1(504:508,th2df(504:508),501:503,'pchip');
    l2df(501:503)=interp1(504:508,l2df(504:508),501:503,'pchip');
    l2ddf=4*[del2(l2f(1:500),t(1:500));del2(l2f(501:1000),t(501:1000))];
    l2ddf(1:3)=interp1([4 5 6 7 8],l2ddf(4:8),[1 2 3],'pchip');
    l2ddf(501:503)=interp1(504:508,l2ddf(504:508),501:503,'pchip');
%-------------------------------------------------------------------------%
%---------------------- CUBIC POLYNOMIAL CORRECTION ----------------------%
% after cubic correction the toe trajectory is hybrid cubic/elliptic.
    [th1c,th2c,th1dc,th2dc,th1ddc,th2ddc] = cubic_correction(t,th1f,...
        th2f,th1df,th2df,leg);

    qjf=[thf,th1c,th2c,l2f];
    qjdf=[thdf,th1dc,th2dc,l2df];
    qjddf=[thddf,th1ddc,th2ddc,l2ddf];
%-------------------------------------------------------------------------%
else
    flag=1;
    th1df=zeros(n,1);
    th2df=zeros(n,1);
    l2df=zeros(n,1);
    th1ddf=zeros(n,1);
    th2ddf=zeros(n,1);
    l2ddf=zeros(n,1);
    qjf=[thf,th1f,th2f,l2f];
    qjdf=[thdf,th1df,th2df,l2df];
    qjddf=[thddf,th1ddf,th2ddf,l2ddf];
```

```
end

end
```

cubic_correction.m

```
function [th1c,th2c,th1dc,th2dc,th1ddc,th2ddc] = cubic_correction(t,th1,...
    th2,th1d,th2d,leg)
%CUBIC_CORRECTION takes as input the joint trajectories of the elliptical
%flight phase and returns cubic polynomial, continuous leg joint
%trajectories.

if leg==1||leg==3
    j0=500;
    jm=750;
    jf=1;
    t0=t(j0); % time of take off
    tm=t(jm); % time of mid flight
    tf=t(jf+999); % time of touch down ending flight phase

    % The boundary conditions are expressed in the form
    % Ap1*cff1=[th1(j0);th1d(j0);th1(jm);th1d(jm)];
    % where Ap1 contains the terms of the polynomials t^p and cff1 contains
    % the coefficients of the polynomials a1, b1, c1, d1.
    % Then cff1=InvAp1*[th1(j0);th1d(j0);th1(jm);th1d(jm)];

    InvAp1=[((-2).*t0+2.*tm).*(t0.^4+(-4).*t0.^3.*tm+6.*t0.^2.*tm.^2+( ...
  -4).*t0.*tm.^3+tm.^4).^(-1),(t0.^2+(-2).*t0.*tm+tm.^2).*( ...
  t0.^4+(-4).*t0.^3.*tm+6.*t0.^2.*tm.^2+(-4).*t0.*tm.^3+tm.^4) ...
  .^(-1),(2.*t0+(-2).*tm).*(t0.^4+(-4).*t0.^3.*tm+6.*t0.^2.* ...
  tm.^2+(-4).*t0.*tm.^3+tm.^4).^(-1),(t0.^2+(-2).*t0.*tm+ ...
  tm.^2).*(t0.^4+(-4).*t0.^3.*tm+6.*t0.^2.*tm.^2+(-4).*t0.* ...
  tm.^3+tm.^4).^(-1);(3.*t0.^2+(-3).*tm.^2).*(t0.^4+(-4).* ...
  t0.^3.*tm+6.*t0.^2.*tm.^2+(-4).*t0.*tm.^3+tm.^4).^(-1),((-1) ...
  .*t0.^3+3.*t0.*tm.^2+(-2).*tm.^3).*(t0.^4+(-4).*t0.^3.*tm+ ...
  6.*t0.^2.*tm.^2+(-4).*t0.*tm.^3+tm.^4).^(-1),((-3).*t0.^2+ ...
  3.*tm.^2).*(t0.^4+(-4).*t0.^3.*tm+6.*t0.^2.*tm.^2+(-4).*t0.* ...
  tm.^3+tm.^4).^(-1),((-2).*t0.^3+3.*t0.^2.*tm+(-1).*tm.^3).*( ...
  t0.^4+(-4).*t0.^3.*tm+6.*t0.^2.*tm.^2+(-4).*t0.*tm.^3+tm.^4) ...
  .^(-1);((-6).*t0.^2.*tm+6.*t0.*tm.^2).*(t0.^4+(-4).*t0.^3.* ...
  tm+6.*t0.^2.*tm.^2+(-4).*t0.*tm.^3+tm.^4).^(-1),(2.*t0.^3.* ...
  tm+(-3).*t0.^2.*tm.^2+tm.^4).*(t0.^4+(-4).*t0.^3.*tm+6.* ...
  t0.^2.*tm.^2+(-4).*t0.*tm.^3+tm.^4).^(-1),(6.*t0.^2.*tm+(-6) ...
  .*t0.*tm.^2).*(t0.^4+(-4).*t0.^3.*tm+6.*t0.^2.*tm.^2+(-4).* ...
  t0.*tm.^3+tm.^4).^(-1),(t0.^4+(-3).*t0.^2.*tm.^2+2.*t0.* ...
  tm.^3).*(t0.^4+(-4).*t0.^3.*tm+6.*t0.^2.*tm.^2+(-4).*t0.* ...
  tm.^3+tm.^4).^(-1);(3.*t0.^2.*tm.^2+(-4).*t0.*tm.^3+tm.^4).* ...
  (t0.^4+(-4).*t0.^3.*tm+6.*t0.^2.*tm.^2+(-4).*t0.*tm.^3+ ...
  tm.^4).^(-1),(t0.^4+(-4).*t0.^3.*tm+6.*t0.^2.*tm.^2+(-4).* ...
  t0.*tm.^3+tm.^4).^(-1).*((-1).*t0.^3.*tm.^2+2.*t0.^2.*tm.^3+ ...
  (-1).*t0.*tm.^4),(t0.^4+(-4).*t0.^3.*tm+3.*t0.^2.*tm.^2).*( ...
  t0.^4+(-4).*t0.^3.*tm+6.*t0.^2.*tm.^2+(-4).*t0.*tm.^3+tm.^4) ...
  .^(-1),((-1).*t0.^4.*tm+2.*t0.^3.*tm.^2+(-1).*t0.^2.*tm.^3) ...
  .*(t0.^4+(-4).*t0.^3.*tm+6.*t0.^2.*tm.^2+(-4).*t0.*tm.^3+ ...
  tm.^4).^(-1)];

InvAp2=[(2.*tf+(-2).*tm).*(tf.^4+(-4).*tf.^3.*tm+6.*tf.^2.*tm.^2+( ...
  -4).*tf.*tm.^3+tm.^4).^(-1),(tf.^2+(-2).*tf.*tm+tm.^2).*( ...
```

```
tf.^4+(-4).*tf.^3.*tm+6.*tf.^2.*tm.^2+(-4).*tf.*tm.^3+tm.^4) ...
.^(-1),((-2).*tf+2.*tm).*(tf.^4+(-4).*tf.^3.*tm+6.*tf.^2.* ...
tm.^2+(-4).*tf.*tm.^3+tm.^4).^(-1),(tf.^2+(-2).*tf.*tm+ ...
tm.^2).*(tf.^4+(-4).*tf.^3.*tm+6.*tf.^2.*tm.^2+(-4).*tf.* ...
tm.^3+tm.^4).^(-1);((-3).*tf.^2+3.*tm.^2).*(tf.^4+(-4).* ...
tf.^3.*tm+6.*tf.^2.*tm.^2+(-4).*tf.*tm.^3+tm.^4).^(-1),((-2) ...
.*tf.^3+3.*tf.^2.*tm+(-1).*tm.^3).*(tf.^4+(-4).*tf.^3.*tm+ ...
6.*tf.^2.*tm.^2+(-4).*tf.*tm.^3+tm.^4).^(-1),(3.*tf.^2+(-3) ...
.*tm.^2).*(tf.^4+(-4).*tf.^3.*tm+6.*tf.^2.*tm.^2+(-4).*tf.* ...
tm.^3+tm.^4).^(-1),((-1).*tf.^3+3.*tf.*tm.^2+(-2).*tm.^3).*( ...
tf.^4+(-4).*tf.^3.*tm+6.*tf.^2.*tm.^2+(-4).*tf.*tm.^3+tm.^4) ...
.^(-1);(6.*tf.^2.*tm+(-6).*tf.*tm.^2).*(tf.^4+(-4).*tf.^3.* ...
tm+6.*tf.^2.*tm.^2+(-4).*tf.*tm.^3+tm.^4).^(-1),(tf.^4+(-3) ...
.*tf.^2.*tm.^2+2.*tf.*tm.^3).*(tf.^4+(-4).*tf.^3.*tm+6.* ...
tf.^2.*tm.^2+(-4).*tf.*tm.^3+tm.^4).^(-1),((-6).*tf.^2.*tm+ ...
6.*tf.*tm.^2).*(tf.^4+(-4).*tf.^3.*tm+6.*tf.^2.*tm.^2+(-4).* ...
tf.*tm.^3+tm.^4).^(-1),(2.*tf.^3.*tm+(-3).*tf.^2.*tm.^2+ ...
tm.^4).*(tf.^4+(-4).*tf.^3.*tm+6.*tf.^2.*tm.^2+(-4).*tf.* ...
tm.^3+tm.^4).^(-1);(tf.^4+(-4).*tf.^3.*tm+3.*tf.^2.*tm.^2).* ...
(tf.^4+(-4).*tf.^3.*tm+6.*tf.^2.*tm.^2+(-4).*tf.*tm.^3+ ...
tm.^4).^(-1),((-1).*tf.^4.*tm+2.*tf.^3.*tm.^2+(-1).*tf.^2.* ...
tm.^3).*(tf.^4+(-4).*tf.^3.*tm+6.*tf.^2.*tm.^2+(-4).*tf.* ...
tm.^3+tm.^4).^(-1),(3.*tf.^2.*tm.^2+(-4).*tf.*tm.^3+tm.^4).* ...
(tf.^4+(-4).*tf.^3.*tm+6.*tf.^2.*tm.^2+(-4).*tf.*tm.^3+ ...
tm.^4).^(-1),(tf.^4+(-4).*tf.^3.*tm+6.*tf.^2.*tm.^2+(-4).* ...
tf.*tm.^3+tm.^4).^(-1).*((-1).*tf.^3.*tm.^2+2.*tf.^2.*tm.^3+ ...
(-1).*tf.*tm.^4)];

cff1=InvAp1*[th1(j0);th1d(j0);th1(jm);th1d(jm)];
cff2=InvAp1*[th2(j0);th2d(j0);th2(jm);th2d(jm)];

cff3=InvAp2*[th1(jm);th1d(jm);th1(jf);th1d(jf)];
cff4=InvAp2*[th2(jm);th2d(jm);th2(jf);th2d(jf)];

a1=cff1(1);
b1=cff1(2);
c1=cff1(3);
d1=cff1(4);

a2=cff2(1);
b2=cff2(2);
c2=cff2(3);
d2=cff2(4);

a3=cff3(1);
b3=cff3(2);
c3=cff3(3);
d3=cff3(4);

a4=cff4(1);
b4=cff4(2);
c4=cff4(3);
d4=cff4(4);

th1c=th1;
th2c=th2;

for j=j0+1:jm
    th1c(j)=a1*t(j)^3+b1*t(j)^2+c1*t(j)+d1;
    th2c(j)=a2*t(j)^3+b2*t(j)^2+c2*t(j)+d2;
```

```
  end
  for j=jm+1:jf+999
      th1c(j)=a3*t(j)^3+b3*t(j)^2+c3*t(j)+d3;
      th2c(j)=a4*t(j)^3+b4*t(j)^2+c4*t(j)+d4;
  end

  dt=t(2)-t(1);
  th2dc=[gradient(th2c(1:500),dt);gradient(th2c(501:1000),dt)];
  th2dc(1:3)=interp1([4 5 6 7 8],th2dc(4:8),[1 2 3],'pchip');
  th2dc(501:503)=interp1(504:508,th2dc(504:508),501:503,'pchip');

  th1dc=[gradient(th1c(1:500),dt);gradient(th1c(501:1000),dt)];
  th1dc(1:3)=interp1([4 5 6 7 8],th1dc(4:8),[1 2 3],'pchip');
  th1dc(501:503)=interp1(504:508,th1dc(504:508),501:503,'pchip');

  th2ddc=4*[del2(th2c(1:500),t(1:500));del2(th2c(501:1000),t(501:1000))];
  th2ddc(1:3)=interp1([4 5 6 7 8],th2ddc(4:8),[1 2 3],'pchip');
  th2ddc(501:503)=interp1(504:508,th2ddc(504:508),501:503,'pchip');

  th1ddc=4*[del2(th1c(1:500),t(1:500));del2(th1c(501:1000),t(501:1000))];
  th1ddc(1:3)=interp1([4 5 6 7 8],th1ddc(4:8),[1 2 3],'pchip');
  th1ddc(501:503)=interp1(504:508,th1ddc(504:508),501:503,'pchip');

end

if leg==2||leg==4
    j0=1000;
    jm=250;
    jf=501;
    t0=t(j0-999); % time of take off
    tm=t(jm);     % time of mid flight
    tf=t(jf);     % time of touch down ending flight phase

    InvAp1=[((-2).*t0+2.*tm).*(t0.^4+(-4).*t0.^3.*tm+6.*t0.^2.*tm.^2+( ...
  -4).*t0.*tm.^3+tm.^4).^(-1),(t0.^2+(-2).*t0.*tm+tm.^2).*( ...
  t0.^4+(-4).*t0.^3.*tm+6.*t0.^2.*tm.^2+(-4).*t0.*tm.^3+tm.^4) ...
  .^(-1),(2.*t0+(-2).*tm).*(t0.^4+(-4).*t0.^3.*tm+6.*t0.^2.* ...
  tm.^2+(-4).*t0.*tm.^3+tm.^4).^(-1),(t0.^2+(-2).*t0.*tm+ ...
  tm.^2).*(t0.^4+(-4).*t0.^3.*tm+6.*t0.^2.*tm.^2+(-4).*t0.* ...
  tm.^3+tm.^4).^(-1);(3.*t0.^2+(-3).*tm.^2).*(t0.^4+(-4).* ...
  t0.^3.*tm+6.*t0.^2.*tm.^2+(-4).*t0.*tm.^3+tm.^4).^(-1),((-1) ...
  .*t0.^3+3.*t0.*tm.^2+(-2).*tm.^3).*(t0.^4+(-4).*t0.^3.*tm+ ...
  6.*t0.^2.*tm.^2+(-4).*t0.*tm.^3+tm.^4).^(-1),((-3).*t0.^2+ ...
  3.*tm.^2).*(t0.^4+(-4).*t0.^3.*tm+6.*t0.^2.*tm.^2+(-4).*t0.* ...
  tm.^3+tm.^4).^(-1),((-2).*t0.^3+3.*t0.^2.*tm+(-1).*tm.^3).*( ...
  t0.^4+(-4).*t0.^3.*tm+6.*t0.^2.*tm.^2+(-4).*t0.*tm.^3+tm.^4) ...
  .^(-1);((-6).*t0.^2.*tm+6.*t0.*tm.^2).*(t0.^4+(-4).*t0.^3.* ...
  tm+6.*t0.^2.*tm.^2+(-4).*t0.*tm.^3+tm.^4).^(-1),(2.*t0.^3.* ...
  tm+(-3).*t0.^2.*tm.^2+tm.^4).*(t0.^4+(-4).*t0.^3.*tm+6.* ...
  t0.^2.*tm.^2+(-4).*t0.*tm.^3+tm.^4).^(-1),(6.*t0.^2.*tm+(-6) ...
  .*t0.*tm.^2).*(t0.^4+(-4).*t0.^3.*tm+6.*t0.^2.*tm.^2+(-4).* ...
  t0.*tm.^3+tm.^4).^(-1),(t0.^4+(-3).*t0.^2.*tm.^2+2.*t0.* ...
  tm.^3).*(t0.^4+(-4).*t0.^3.*tm+6.*t0.^2.*tm.^2+(-4).*t0.* ...
  tm.^3+tm.^4).^(-1);(3.*t0.^2.*tm.^2+(-4).*t0.*tm.^3+tm.^4).* ...
  (t0.^4+(-4).*t0.^3.*tm+6.*t0.^2.*tm.^2+(-4).*t0.*tm.^3+ ...
  tm.^4).^(-1),(t0.^4+(-4).*t0.^3.*tm+6.*t0.^2.*tm.^2+(-4).* ...
  t0.*tm.^3+tm.^4).^(-1).*((-1).*t0.^3.*tm.^2+2.*t0.^2.*tm.^3+ ...
  (-1).*t0.*tm.^4),(t0.^4+(-4).*t0.^3.*tm+3.*t0.^2.*tm.^2).*( ...
  t0.^4+(-4).*t0.^3.*tm+6.*t0.^2.*tm.^2+(-4).*t0.*tm.^3+tm.^4) ...
  .^(-1),((-1).*t0.^4.*tm+2.*t0.^3.*tm.^2+(-1).*t0.^2.*tm.^3) ...
```

```
  .*(t0.^4+(-4).*t0.^3.*tm+6.*t0.^2.*tm.^2+(-4).*t0.*tm.^3+ ...
  tm.^4).^(-1)];

InvAp2=[(2.*tf+(-2).*tm).*(tf.^4+(-4).*tf.^3.*tm+6.*tf.^2.*tm.^2+( ...
  -4).*tf.*tm.^3+tm.^4).^(-1),(tf.^2+(-2).*tf.*tm+tm.^2).*( ...
  tf.^4+(-4).*tf.^3.*tm+6.*tf.^2.*tm.^2+(-4).*tf.*tm.^3+tm.^4) ...
  .^(-1),((-2).*tf+2.*tm).*(tf.^4+(-4).*tf.^3.*tm+6.*tf.^2.* ...
  tm.^2+(-4).*tf.*tm.^3+tm.^4).^(-1),(tf.^2+(-2).*tf.*tm+ ...
  tm.^2).*(tf.^4+(-4).*tf.^3.*tm+6.*tf.^2.*tm.^2+(-4).*tf.* ...
  tm.^3+tm.^4).^(-1);((-3).*tf.^2+3.*tm.^2).*(tf.^4+(-4).* ...
  tf.^3.*tm+6.*tf.^2.*tm.^2+(-4).*tf.*tm.^3+tm.^4).^(-1),((-2) ...
  .*tf.^3+3.*tf.^2.*tm+(-1).*tm.^3).*(tf.^4+(-4).*tf.^3.*tm+ ...
  6.*tf.^2.*tm.^2+(-4).*tf.*tm.^3+tm.^4).^(-1),(3.*tf.^2+(-3) ...
  .*tm.^2).*(tf.^4+(-4).*tf.^3.*tm+6.*tf.^2.*tm.^2+(-4).*tf.* ...
  tm.^3+tm.^4).^(-1),((-1).*tf.^3+3.*tf.*tm.^2+(-2).*tm.^3).*( ...
  tf.^4+(-4).*tf.^3.*tm+6.*tf.^2.*tm.^2+(-4).*tf.*tm.^3+tm.^4) ...
  .^(-1);(6.*tf.^2.*tm+(-6).*tf.*tm.^2).*(tf.^4+(-4).*tf.^3.* ...
  tm+6.*tf.^2.*tm.^2+(-4).*tf.*tm.^3+tm.^4).^(-1),(tf.^4+(-3) ...
  .*tf.^2.*tm.^2+2.*tf.*tm.^3).*(tf.^4+(-4).*tf.^3.*tm+6.* ...
  tf.^2.*tm.^2+(-4).*tf.*tm.^3+tm.^4).^(-1),((-6).*tf.^2.*tm+ ...
  6.*tf.*tm.^2).*(tf.^4+(-4).*tf.^3.*tm+6.*tf.^2.*tm.^2+(-4).* ...
  tf.*tm.^3+tm.^4).^(-1),(2.*tf.^3.*tm+(-3).*tf.^2.*tm.^2+ ...
  tm.^4).*(tf.^4+(-4).*tf.^3.*tm+6.*tf.^2.*tm.^2+(-4).*tf.* ...
  tm.^3+tm.^4).^(-1);(tf.^4+(-4).*tf.^3.*tm+3.*tf.^2.*tm.^2).* ...
  (tf.^4+(-4).*tf.^3.*tm+6.*tf.^2.*tm.^2+(-4).*tf.*tm.^3+ ...
  tm.^4).^(-1),((-1).*tf.^4.*tm+2.*tf.^3.*tm.^2+(-1).*tf.^2.* ...
  tm.^3).*(tf.^4+(-4).*tf.^3.*tm+6.*tf.^2.*tm.^2+(-4).*tf.* ...
  tm.^3+tm.^4).^(-1),(3.*tf.^2.*tm.^2+(-4).*tf.*tm.^3+tm.^4).* ...
  (tf.^4+(-4).*tf.^3.*tm+6.*tf.^2.*tm.^2+(-4).*tf.*tm.^3+ ...
  tm.^4).^(-1),(tf.^4+(-4).*tf.^3.*tm+6.*tf.^2.*tm.^2+(-4).* ...
  tf.*tm.^3+tm.^4).^(-1).*((-1).*tf.^3.*tm.^2+2.*tf.^2.*tm.^3+ ...
  (-1).*tf.*tm.^4)];

  cff1=InvAp1*[th1(j0);th1d(j0);th1(jm);th1d(jm)];
  cff2=InvAp1*[th2(j0);th2d(j0);th2(jm);th2d(jm)];

  cff3=InvAp2*[th1(jm);th1d(jm);th1(jf);th1d(jf)];
  cff4=InvAp2*[th2(jm);th2d(jm);th2(jf);th2d(jf)];

  a1=cff1(1);
  b1=cff1(2);
  c1=cff1(3);
  d1=cff1(4);

  a2=cff2(1);
  b2=cff2(2);
  c2=cff2(3);
  d2=cff2(4);

  a3=cff3(1);
  b3=cff3(2);
  c3=cff3(3);
  d3=cff3(4);

  a4=cff4(1);
  b4=cff4(2);
  c4=cff4(3);
  d4=cff4(4);

  th1c=th1;
```

```
  th2c=th2;

  for j=j0-999:jm
      th1c(j)=a1*t(j)^3+b1*t(j)^2+c1*t(j)+d1;
      th2c(j)=a2*t(j)^3+b2*t(j)^2+c2*t(j)+d2;
  end
  for j=jm+1:jf-1
      th1c(j)=a3*t(j)^3+b3*t(j)^2+c3*t(j)+d3;
      th2c(j)=a4*t(j)^3+b4*t(j)^2+c4*t(j)+d4;
  end

  dt=t(2)-t(1);
  th2dc=[gradient(th2c(1:500),dt);gradient(th2c(501:1000),dt)];
  th2dc(1:3)=interp1([4 5 6 7 8],th2dc(4:8),[1 2 3],'pchip');
  th2dc(501:503)=interp1(504:508,th2dc(504:508),501:503,'pchip');

  th1dc=[gradient(th1c(1:500),dt);gradient(th1c(501:1000),dt)];
  th1dc(1:3)=interp1([4 5 6 7 8],th1dc(4:8),[1 2 3],'pchip');
  th1dc(501:503)=interp1(504:508,th1dc(504:508),501:503,'pchip');

  th2ddc=4*[del2(th2c(1:500),t(1:500));del2(th2c(501:1000),t(501:1000))];
  th2ddc(1:3)=interp1([4 5 6 7 8],th2ddc(4:8),[1 2 3],'pchip');
  th2ddc(501:503)=interp1(504:508,th2ddc(504:508),501:503,'pchip');

  th1ddc=4*[del2(th1c(1:500),t(1:500));del2(th1c(501:1000),t(501:1000))];
  th1ddc(1:3)=interp1([4 5 6 7 8],th1ddc(4:8),[1 2 3],'pchip');
  th1ddc(501:503)=interp1(504:508,th1ddc(504:508),501:503,'pchip');

end


end
```

Torques.m

```
function [torque1h1,torque2h1,torque1h4,torque2h4,torque1f2,torque2f2,...
torque1f3,torque2f3,flag] = Torques(xd,xdd,yd,ydd,th,thd,thdd,th1h1,...
    th1dh1,th1ddh1,l1h1,th2h1,th2dh1,th2ddh1,l2h1,l2dh1,l2ddh1,th1h4,...
    th1dh4,th1ddh4,l1h4,th2h4,th2dh4,th2ddh4,l2h4,l2dh4,l2ddh4,th1f2,...
    th1df2,th1ddf2,l1f2,th2f2,th2df2,th2ddf2,l2f2,l2df2,l2ddf2,th1f3,...
    th1df3,th1ddf3,l1f3,th2f3,th2df3,th2ddf3,l2f3,l2df3,l2ddf3,g,d,...
    tt1,tt2,tt3,tt4,t,T,fx,fy)

%TORQUES calculates the required joint torques, taking in account the full
%quadruped dynamics and the joint trajectories.

%------------------------- ACTUATOR PROPERTIES -------------------------%
% rotor inertia before reduction
Ir1=542*10^(-7);
Ir2=542*10^(-7);
% reduction ratios
n1=53;
n2=53;
%-----------------------------------------------------------------------%
%------------------------ CARBON TUBE PROPERTIES ------------------------%
dens=1466;
```

```
[din,dout,flag] = check_strength(fx,fy,th,th1h1,l1h1,th2h1,l2h1...
    ,th1f2,l1f2,th2f2,l2f2,th1f3,l1f3,th2f3,l2f3,th1h4,l1h4,th2h4,l2h4);
A=pi*(dout^2-din^2)/4;
%-------------------------------------------------------------------------%
%---------------------- LEG LUMPED MASS PROPERTIES -----------------------%
mj1=0.4; % hip joint mass
mj2=0.4; % knee joint mass
mt=0.1;  % toe mass
ms=0.17; % spring mass
%-------------------------------------------------------------------------%
%------------------ LEG INERTIAL PROPERTIES CALCULATION ------------------%
% LEG 1 PROPERTIES
mr1h1=dens*A*l1h1;
l20h1=l2h1(1);
mr2h1=dens*A*l20h1;
m1h1=mj1/2+mr1h1+mj2/2;
m2h1=mj2/2+ms+mr2h1+mt;
lcm1h1=(mr1h1*l1h1/2+mj2/2*l1h1)/m1h1;
I1h1=1/12*m1h1*l1h1^2+Ir1*n1^2;
I2h1=1/12*m2h1*l2h1.^2+Ir2*n2^2;

% LEG 4 PROPERTIES
mr1h4=dens*A*l1h4;
l20h4=l2h4(1);
mr2h4=dens*A*l20h4;
m1h4=mj1/2+mr1h4+mj2/2;
m2h4=mj2/2+ms+mr2h4+mt;
lcm1h4=(mr1h4*l1h4/2+mj2/2*l1h4)/m1h4;
I1h4=1/12*m1h4*l1h4^2+Ir1*n1^2;
I2h4=1/12*m2h4*l2h4.^2+Ir2*n2^2;

% LEG 2 PROPERTIES
mr1f2=dens*A*l1f2;
l20f2=l2f2(1);
mr2f2=dens*A*l20f2;
m1f2=mj1/2+mr1f2+mj2/2;
m2f2=mj2/2+ms+mr2f2+mt;
lcm1f2=(mr1f2*l1f2/2+mj2/2*l1f2)/m1f2;
I1f2=1/12*m1f2*l1f2^2+Ir1*n1^2;
I2f2=1/12*m2f2*l2f2.^2+Ir2*n2^2;

% LEG 3 PROPERTIES
mr1f3=dens*A*l1f3;
l20f3=l2f3(1);
mr2f3=dens*A*l20f3;
m1f3=mj1/2+mr1f3+mj2/2;
m2f3=mj2/2+ms+mr2f3+mt;
lcm1f3=(mr1f3*l1f3/2+mj2/2*l1f3)/m1f3;
I1f3=1/12*m1f3*l1f3^2+Ir1*n1^2;
I2f3=1/12*m2f3*l2f3.^2+Ir2*n2^2;
%-------------------------------------------------------------------------%
%------------------------ FORCES FROM THE GROUND -------------------------%
% total forces on the CoM are shared from the two legs in phase.
fx1=fx/2;
fy1=fy/2;
fx2=fx/2;
fy2=fy/2;
fx3=fx/2;
fy3=fy/2;
fx4=fx/2;
fy4=fy/2;
```

```
%--------------------------------------------------------------------------%
%--------------------------- LEG JOINT TORQUES ----------------------------%
% The leg joint torques as calculated from the full robot dynamics
n=length(th);
torque1h1=zeros(n,1);
torque2h1=zeros(n,1);
torque1h4=zeros(n,1);
torque2h4=zeros(n,1);
torque1f2=zeros(n,1);
torque2f2=zeros(n,1);
torque1f3=zeros(n,1);
torque2f3=zeros(n,1);

for j=1:n

    [c1,c2,c3,c4]=c_i(tt1,tt2,tt3,tt4,t(j),T);

torque1h1(j)=(-0.1E1).*c1.*l1h1.*cos(th(j)+th1h1(j)).*fx1(j)+g.*...
    (0.1E1.*lcm1h1.*m1h1+0.1E1.*l1h1.*m2h1).*sin(th(j)+th1h1(j))+( ...
  -0.1E1).*c1.*l1h1.*fy1(j).*sin(th(j)+th1h1(j))+(-0.5E0).* ...
  l1h1.*mr2h1.*l2ddh1(j).*sin(th1h1(j)+(-1).*th2h1(j))+( ...
  -0.25E0).*l1h1.*ms.*l2ddh1(j).*sin(th1h1(j)+(-1).*th2h1(j))+ ...
  (-0.1E1).*l1h1.*mt.*l2ddh1(j).*sin(th1h1(j)+(-1).*th2h1(j))+ ...
  0.1E1.*I1h1.*th1ddh1(j)+0.1E1.*lcm1h1.^2.*m1h1.*th1ddh1(j)+ ...
  0.1E1.*l1h1.^2.*m2h1.*th1ddh1(j)+0.5E0.*l1h1.*mr2h1.*cos( ...
  th1h1(j)+(-1).*th2h1(j)).*l2h1(j).*th2ddh1(j)+0.25E0.*l1h1.* ...
  ms.*cos(th1h1(j)+(-1).*th2h1(j)).*l2h1(j).*th2ddh1(j)+ ...
  0.1E1.*l1h1.*mt.*cos(th1h1(j)+(-1).*th2h1(j)).*l2h1(j).* ...
  th2ddh1(j)+0.1E1.*l1h1.*mr2h1.*cos(th1h1(j)+(-1).*th2h1(j)) ...
  .*l2dh1(j).*th2dh1(j)+0.5E0.*l1h1.*ms.*cos(th1h1(j)+(-1).* ...
  th2h1(j)).*l2dh1(j).*th2dh1(j)+0.2E1.*l1h1.*mt.*cos(th1h1(j) ...
  +(-1).*th2h1(j)).*l2dh1(j).*th2dh1(j)+0.5E0.*l1h1.*mr2h1.* ...
  l2h1(j).*sin(th1h1(j)+(-1).*th2h1(j)).*th2dh1(j).^2+0.25E0.* ...
  l1h1.*ms.*l2h1(j).*sin(th1h1(j)+(-1).*th2h1(j)).*th2dh1(j) ...
  .^2+0.1E1.*l1h1.*mt.*l2h1(j).*sin(th1h1(j)+(-1).*th2h1(j)).* ...
  th2dh1(j).^2+0.1E1.*l1h1.*mr2h1.*cos(th1h1(j)+(-1).*th2h1(j) ...
  ).*l2dh1(j).*thd(j)+0.5E0.*l1h1.*ms.*cos(th1h1(j)+(-1).* ...
  th2h1(j)).*l2dh1(j).*thd(j)+0.2E1.*l1h1.*mt.*cos(th1h1(j)+( ...
  -1).*th2h1(j)).*l2dh1(j).*thd(j)+0.1E1.*l1h1.*mr2h1.*l2h1(j) ...
  .*sin(th1h1(j)+(-1).*th2h1(j)).*th2dh1(j).*thd(j)+0.5E0.* ...
  l1h1.*ms.*l2h1(j).*sin(th1h1(j)+(-1).*th2h1(j)).*th2dh1(j).* ...
  thd(j)+0.2E1.*l1h1.*mt.*l2h1(j).*sin(th1h1(j)+(-1).*th2h1(j) ...
  ).*th2dh1(j).*thd(j)+0.1E1.*d.*lcm1h1.*m1h1.*cos(th1h1(j)).* ...
  thd(j).^2+0.1E1.*d.*l1h1.*m2h1.*cos(th1h1(j)).*thd(j).^2+ ...
  0.5E0.*l1h1.*mr2h1.*l2h1(j).*sin(th1h1(j)+(-1).*th2h1(j)).* ...
  thd(j).^2+0.25E0.*l1h1.*ms.*l2h1(j).*sin(th1h1(j)+(-1).* ...
  th2h1(j)).*thd(j).^2+0.1E1.*l1h1.*mt.*l2h1(j).*sin(th1h1(j)+ ...
  (-1).*th2h1(j)).*thd(j).^2+0.1E1.*lcm1h1.^2.*m1h1.*thdd(j)+ ...
  0.1E1.*l1h1.^2.*m2h1.*thdd(j)+0.5E0.*l1h1.*mr2h1.*cos(th1h1( ...
  j)+(-1).*th2h1(j)).*l2h1(j).*thdd(j)+0.25E0.*l1h1.*ms.*cos( ...
  th1h1(j)+(-1).*th2h1(j)).*l2h1(j).*thdd(j)+0.1E1.*l1h1.*mt.* ...
  cos(th1h1(j)+(-1).*th2h1(j)).*l2h1(j).*thdd(j)+(-0.1E1).*d.* ...
  lcm1h1.*m1h1.*sin(th1h1(j)).*thdd(j)+(-0.1E1).*d.*l1h1.* ...
  m2h1.*sin(th1h1(j)).*thdd(j)+0.1E1.*lcm1h1.*m1h1.*cos(th(j)+ ...
  th1h1(j)).*xdd(j)+0.1E1.*l1h1.*m2h1.*cos(th(j)+th1h1(j)).* ...
  xdd(j)+(0.1E1.*lcm1h1.*m1h1+0.1E1.*l1h1.*m2h1).*sin(th(j)+ ...
  th1h1(j)).*ydd(j);


torque2h1(j)=m2h1.^(-1).*((-0.1E1).*c1.*m2h1.*cos(th(j)+th2h1(j)).*...
```

```
    fx1(j).*l2h1(j)+0.1E1.*m2h1.*I2h1(j).*th2ddh1(j)+(0.25E0.* ...
  mr2h1.^2+0.25E0.*mr2h1.*ms+0.625E-1.*ms.^2+0.1E1.*mr2h1.*mt+ ...
  0.5E0.*ms.*mt+0.1E1.*mt.^2).*l2h1(j).^2.*(th2ddh1(j)+thdd(j) ...
  )+l2h1(j).*((-0.1E1).*c1.*m2h1.*fy1(j).*sin(th(j)+th2h1(j))+ ...
  (0.5E0.*mr2h1.^2+0.5E0.*mr2h1.*ms+0.125E0.*ms.^2+0.2E1.* ...
  mr2h1.*mt+0.1E1.*ms.*mt+0.2E1.*mt.^2).*l2dh1(j).*(th2dh1(j)+ ...
  thd(j))+m2h1.*(g.*(0.5E0.*mr2h1+0.25E0.*ms+0.1E1.*mt).*sin( ...
  th(j)+th2h1(j))+0.5E0.*l1h1.*mr2h1.*cos(th1h1(j)+(-1).* ...
  th2h1(j)).*th1ddh1(j)+0.25E0.*l1h1.*ms.*cos(th1h1(j)+(-1).* ...
  th2h1(j)).*th1ddh1(j)+0.1E1.*l1h1.*mt.*cos(th1h1(j)+(-1).* ...
  th2h1(j)).*th1ddh1(j)+l1h1.*((-0.5E0).*mr2h1+(-0.25E0).*ms+( ...
  -0.1E1).*mt).*sin(th1h1(j)+(-1).*th2h1(j)).*th1dh1(j).^2+ ...
  l1h1.*((-0.1E1).*mr2h1+(-0.5E0).*ms+(-0.2E1).*mt).*sin( ...
  th1h1(j)+(-1).*th2h1(j)).*th1dh1(j).*thd(j)+(d.*(0.5E0.* ...
  mr2h1+0.25E0.*ms+0.1E1.*mt).*cos(th2h1(j))+l1h1.*((-0.5E0).* ...
  mr2h1+(-0.25E0).*ms+(-0.1E1).*mt).*sin(th1h1(j)+(-1).*th2h1( ...
  j))).*thd(j).^2+0.5E0.*l1h1.*mr2h1.*cos(th1h1(j)+(-1).* ...
  th2h1(j)).*thdd(j)+0.25E0.*l1h1.*ms.*cos(th1h1(j)+(-1).* ...
  th2h1(j)).*thdd(j)+0.1E1.*l1h1.*mt.*cos(th1h1(j)+(-1).* ...
  th2h1(j)).*thdd(j)+(-0.5E0).*d.*mr2h1.*sin(th2h1(j)).*thdd( ...
  j)+(-0.25E0).*d.*ms.*sin(th2h1(j)).*thdd(j)+(-0.1E1).*d.* ...
  mt.*sin(th2h1(j)).*thdd(j)+0.5E0.*mr2h1.*cos(th(j)+th2h1(j)) ...
  .*xdd(j)+0.25E0.*ms.*cos(th(j)+th2h1(j)).*xdd(j)+0.1E1.*mt.* ...
  cos(th(j)+th2h1(j)).*xdd(j)+0.5E0.*mr2h1.*sin(th(j)+th2h1(j) ...
  ).*ydd(j)+0.25E0.*ms.*sin(th(j)+th2h1(j)).*ydd(j)+0.1E1.* ...
  mt.*sin(th(j)+th2h1(j)).*ydd(j))));


torque1h4(j)=(-0.1E1).*c4.*l1h4.*cos(th(j)+th1h4(j)).*fx4(j)+g.*...
    (0.1E1.*lcm1h4.*m1h4+0.1E1.*l1h4.*m2h4).*sin(th(j)+th1h4(j))+( ...
  -0.1E1).*c4.*l1h4.*fy4(j).*sin(th(j)+th1h4(j))+(-0.5E0).* ...
  l1h4.*mr2h4.*l2ddh4(j).*sin(th1h4(j)+(-1).*th2h4(j))+( ...
  -0.25E0).*l1h4.*ms.*l2ddh4(j).*sin(th1h4(j)+(-1).*th2h4(j))+ ...
  (-0.1E1).*l1h4.*mt.*l2ddh4(j).*sin(th1h4(j)+(-1).*th2h4(j))+ ...
  0.1E1.*I1h4.*th1ddh4(j)+0.1E1.*lcm1h4.^2.*m1h4.*th1ddh4(j)+ ...
  0.1E1.*l1h4.^2.*m2h4.*th1ddh4(j)+0.5E0.*l1h4.*mr2h4.*cos( ...
  th1h4(j)+(-1).*th2h4(j)).*l2h4(j).*th2ddh4(j)+0.25E0.*l1h4.* ...
  ms.*cos(th1h4(j)+(-1).*th2h4(j)).*l2h4(j).*th2ddh4(j)+ ...
  0.1E1.*l1h4.*mt.*cos(th1h4(j)+(-1).*th2h4(j)).*l2h4(j).* ...
  th2ddh4(j)+0.1E1.*l1h4.*mr2h4.*cos(th1h4(j)+(-1).*th2h4(j)) ...
  .*l2dh4(j).*th2dh4(j)+0.5E0.*l1h4.*ms.*cos(th1h4(j)+(-1).* ...
  th2h4(j)).*l2dh4(j).*th2dh4(j)+0.2E1.*l1h4.*mt.*cos(th1h4(j) ...
  +(-1).*th2h4(j)).*l2dh4(j).*th2dh4(j)+0.5E0.*l1h4.*mr2h4.* ...
  l2h4(j).*sin(th1h4(j)+(-1).*th2h4(j)).*th2dh4(j).^2+0.25E0.* ...
  l1h4.*ms.*l2h4(j).*sin(th1h4(j)+(-1).*th2h4(j)).*th2dh4(j) ...
  .^2+0.1E1.*l1h4.*mt.*l2h4(j).*sin(th1h4(j)+(-1).*th2h4(j)).* ...
  th2dh4(j).^2+0.1E1.*l1h4.*mr2h4.*cos(th1h4(j)+(-1).*th2h4(j) ...
  ).*l2dh4(j).*thd(j)+0.5E0.*l1h4.*ms.*cos(th1h4(j)+(-1).* ...
  th2h4(j)).*l2dh4(j).*thd(j)+0.2E1.*l1h4.*mt.*cos(th1h4(j)+( ...
  -1).*th2h4(j)).*l2dh4(j).*thd(j)+0.1E1.*l1h4.*mr2h4.*l2h4(j) ...
  .*sin(th1h4(j)+(-1).*th2h4(j)).*th2dh4(j).*thd(j)+0.5E0.* ...
  l1h4.*ms.*l2h4(j).*sin(th1h4(j)+(-1).*th2h4(j)).*th2dh4(j).* ...
  thd(j)+0.2E1.*l1h4.*mt.*l2h4(j).*sin(th1h4(j)+(-1).*th2h4(j) ...
  ).*th2dh4(j).*thd(j)+0.1E1.*d.*lcm1h4.*m1h4.*cos(th1h4(j)).* ...
  thd(j).^2+0.1E1.*d.*l1h4.*m2h4.*cos(th1h4(j)).*thd(j).^2+ ...
  0.5E0.*l1h4.*mr2h4.*l2h4(j).*sin(th1h4(j)+(-1).*th2h4(j)).* ...
  thd(j).^2+0.25E0.*l1h4.*ms.*l2h4(j).*sin(th1h4(j)+(-1).* ...
  th2h4(j)).*thd(j).^2+0.1E1.*l1h4.*mt.*l2h4(j).*sin(th1h4(j)+ ...
  (-1).*th2h4(j)).*thd(j).^2+0.1E1.*lcm1h4.^2.*m1h4.*thdd(j)+ ...
  0.1E1.*l1h4.^2.*m2h4.*thdd(j)+0.5E0.*l1h4.*mr2h4.*cos(th1h4( ...
  j)+(-1).*th2h4(j)).*l2h4(j).*thdd(j)+0.25E0.*l1h4.*ms.*cos( ...
```

```
  th1h4(j)+(-1).*th2h4(j)).*l2h4(j).*thdd(j)+0.1E1.*l1h4.*mt.* ...
  cos(th1h4(j)+(-1).*th2h4(j)).*l2h4(j).*thdd(j)+(-0.1E1).*d.* ...
  lcm1h4.*m1h4.*sin(th1h4(j)).*thdd(j)+(-0.1E1).*d.*l1h4.* ...
  m2h4.*sin(th1h4(j)).*thdd(j)+0.1E1.*lcm1h4.*m1h4.*cos(th(j)+ ...
  th1h4(j)).*xdd(j)+0.1E1.*l1h4.*m2h4.*cos(th(j)+th1h4(j)).* ...
  xdd(j)+(0.1E1.*lcm1h4.*m1h4+0.1E1.*l1h4.*m2h4).*sin(th(j)+ ...
  th1h4(j)).*ydd(j);


torque2h4(j)=m2h4.^(-1).*((-0.1E1).*c4.*m2h4.*cos(th(j)+th2h4(j)).*...
    fx4(j).*l2h4(j)+0.1E1.*m2h4.*I2h4(j).*th2ddh4(j)+(0.25E0.* ...
  mr2h4.^2+0.25E0.*mr2h4.*ms+0.625E-1.*ms.^2+0.1E1.*mr2h4.*mt+ ...
  0.5E0.*ms.*mt+0.1E1.*mt.^2).*l2h4(j).^2.*(th2ddh4(j)+thdd(j) ...
  )+l2h4(j).*((-0.1E1).*c4.*m2h4.*fy4(j).*sin(th(j)+th2h4(j))+ ...
  (0.5E0.*mr2h4.^2+0.5E0.*mr2h4.*ms+0.125E0.*ms.^2+0.2E1.* ...
  mr2h4.*mt+0.1E1.*ms.*mt+0.2E1.*mt.^2).*l2dh4(j).*(th2dh4(j)+ ...
  thd(j))+m2h4.*(g.*(0.5E0.*mr2h4+0.25E0.*ms+0.1E1.*mt).*sin( ...
  th(j)+th2h4(j))+0.5E0.*l1h4.*mr2h4.*cos(th1h4(j)+(-1).* ...
  th2h4(j)).*th1ddh4(j)+0.25E0.*l1h4.*ms.*cos(th1h4(j)+(-1).* ...
  th2h4(j)).*th1ddh4(j)+0.1E1.*l1h4.*mt.*cos(th1h4(j)+(-1).* ...
  th2h4(j)).*th1ddh4(j)+l1h4.*((-0.5E0).*mr2h4+(-0.25E0).*ms+( ...
  -0.1E1).*mt).*sin(th1h4(j)+(-1).*th2h4(j)).*th1dh4(j).^2+ ...
  l1h4.*((-0.1E1).*mr2h4+(-0.5E0).*ms+(-0.2E1).*mt).*sin( ...
  th1h4(j)+(-1).*th2h4(j)).*th1dh4(j).*thd(j)+(d.*(0.5E0.* ...
  mr2h4+0.25E0.*ms+0.1E1.*mt).*cos(th2h4(j))+l1h4.*((-0.5E0).* ...
  mr2h4+(-0.25E0).*ms+(-0.1E1).*mt).*sin(th1h4(j)+(-1).*th2h4( ...
  j))).*thd(j).^2+0.5E0.*l1h4.*mr2h4.*cos(th1h4(j)+(-1).* ...
  th2h4(j)).*thdd(j)+0.25E0.*l1h4.*ms.*cos(th1h4(j)+(-1).* ...
  th2h4(j)).*thdd(j)+0.1E1.*l1h4.*mt.*cos(th1h4(j)+(-1).* ...
  th2h4(j)).*thdd(j)+(-0.5E0).*d.*mr2h4.*sin(th2h4(j)).*thdd( ...
  j)+(-0.25E0).*d.*ms.*sin(th2h4(j)).*thdd(j)+(-0.1E1).*d.* ...
  mt.*sin(th2h4(j)).*thdd(j)+0.5E0.*mr2h4.*cos(th(j)+th2h4(j)) ...
  .*xdd(j)+0.25E0.*ms.*cos(th(j)+th2h4(j)).*xdd(j)+0.1E1.*mt.* ...
  cos(th(j)+th2h4(j)).*xdd(j)+0.5E0.*mr2h4.*sin(th(j)+th2h4(j) ...
  ).*ydd(j)+0.25E0.*ms.*sin(th(j)+th2h4(j)).*ydd(j)+0.1E1.* ...
  mt.*sin(th(j)+th2h4(j)).*ydd(j))));


torque1f2(j)=(-0.1E1).*c2.*l1f2.*cos(th(j)+th1f2(j)).*fx2(j)+g.*...
    (0.1E1.*lcm1f2.*m1f2+0.1E1.*l1f2.*m2f2).*sin(th(j)+th1f2(j))+( ...
  -0.1E1).*c2.*l1f2.*fy2(j).*sin(th(j)+th1f2(j))+(-0.5E0).* ...
  l1f2.*mr2f2.*l2ddf2(j).*sin(th1f2(j)+(-1).*th2f2(j))+( ...
  -0.25E0).*l1f2.*ms.*l2ddf2(j).*sin(th1f2(j)+(-1).*th2f2(j))+ ...
  (-0.1E1).*l1f2.*mt.*l2ddf2(j).*sin(th1f2(j)+(-1).*th2f2(j))+ ...
  0.1E1.*I1f2.*th1ddf2(j)+0.1E1.*lcm1f2.^2.*m1f2.*th1ddf2(j)+ ...
  0.1E1.*l1f2.^2.*m2f2.*th1ddf2(j)+0.5E0.*l1f2.*mr2f2.*cos( ...
  th1f2(j)+(-1).*th2f2(j)).*l2f2(j).*th2ddf2(j)+0.25E0.*l1f2.* ...
  ms.*cos(th1f2(j)+(-1).*th2f2(j)).*l2f2(j).*th2ddf2(j)+ ...
  0.1E1.*l1f2.*mt.*cos(th1f2(j)+(-1).*th2f2(j)).*l2f2(j).* ...
  th2ddf2(j)+0.1E1.*l1f2.*mr2f2.*cos(th1f2(j)+(-1).*th2f2(j)) ...
  .*l2df2(j).*th2df2(j)+0.5E0.*l1f2.*ms.*cos(th1f2(j)+(-1).* ...
  th2f2(j)).*l2df2(j).*th2df2(j)+0.2E1.*l1f2.*mt.*cos(th1f2(j) ...
  +(-1).*th2f2(j)).*l2df2(j).*th2df2(j)+0.5E0.*l1f2.*mr2f2.* ...
  l2f2(j).*sin(th1f2(j)+(-1).*th2f2(j)).*th2df2(j).^2+0.25E0.* ...
  l1f2.*ms.*l2f2(j).*sin(th1f2(j)+(-1).*th2f2(j)).*th2df2(j) ...
  .^2+0.1E1.*l1f2.*mt.*l2f2(j).*sin(th1f2(j)+(-1).*th2f2(j)).* ...
  th2df2(j).^2+0.1E1.*l1f2.*mr2f2.*cos(th1f2(j)+(-1).*th2f2(j) ...
  ).*l2df2(j).*thd(j)+0.5E0.*l1f2.*ms.*cos(th1f2(j)+(-1).* ...
  th2f2(j)).*l2df2(j).*thd(j)+0.2E1.*l1f2.*mt.*cos(th1f2(j)+( ...
  -1).*th2f2(j)).*l2df2(j).*thd(j)+0.1E1.*l1f2.*mr2f2.*l2f2(j) ...
```

```
  .*sin(th1f2(j)+(-1).*th2f2(j)).*th2df2(j).*thd(j)+0.5E0.* ...
  l1f2.*ms.*l2f2(j).*sin(th1f2(j)+(-1).*th2f2(j)).*th2df2(j).* ...
  thd(j)+0.2E1.*l1f2.*mt.*l2f2(j).*sin(th1f2(j)+(-1).*th2f2(j) ...
  ).*th2df2(j).*thd(j)+(-0.1E1).*d.*lcm1f2.*m1f2.*cos(th1f2(j) ...
  ).*thd(j).^2+(-0.1E1).*d.*l1f2.*m2f2.*cos(th1f2(j)).*thd(j) ...
  .^2+0.5E0.*l1f2.*mr2f2.*l2f2(j).*sin(th1f2(j)+(-1).*th2f2(j) ...
  ).*thd(j).^2+0.25E0.*l1f2.*ms.*l2f2(j).*sin(th1f2(j)+(-1).* ...
  th2f2(j)).*thd(j).^2+0.1E1.*l1f2.*mt.*l2f2(j).*sin(th1f2(j)+ ...
  (-1).*th2f2(j)).*thd(j).^2+0.1E1.*lcm1f2.^2.*m1f2.*thdd(j)+ ...
  0.1E1.*l1f2.^2.*m2f2.*thdd(j)+0.5E0.*l1f2.*mr2f2.*cos(th1f2( ...
  j)+(-1).*th2f2(j)).*l2f2(j).*thdd(j)+0.25E0.*l1f2.*ms.*cos( ...
  th1f2(j)+(-1).*th2f2(j)).*l2f2(j).*thdd(j)+0.1E1.*l1f2.*mt.* ...
  cos(th1f2(j)+(-1).*th2f2(j)).*l2f2(j).*thdd(j)+0.1E1.*d.* ...
  lcm1f2.*m1f2.*sin(th1f2(j)).*thdd(j)+0.1E1.*d.*l1f2.*m2f2.* ...
  sin(th1f2(j)).*thdd(j)+0.1E1.*lcm1f2.*m1f2.*cos(th(j)+th1f2( ...
  j)).*xdd(j)+0.1E1.*l1f2.*m2f2.*cos(th(j)+th1f2(j)).*xdd(j)+( ...
  0.1E1.*lcm1f2.*m1f2+0.1E1.*l1f2.*m2f2).*sin(th(j)+th1f2(j)) ...
  .*ydd(j);


torque2f2(j)=m2f2.^(-1).*((-0.1E1).*c2.*m2f2.*cos(th(j)+th2f2(j)).*...
    fx2(j).*l2f2(j)+0.1E1.*m2f2.*I2f2(j).*th2ddf2(j)+(0.25E0.* ...
  mr2f2.^2+0.25E0.*mr2f2.*ms+0.625E-1.*ms.^2+0.1E1.*mr2f2.*mt+ ...
  0.5E0.*ms.*mt+0.1E1.*mt.^2).*l2f2(j).^2.*(th2ddf2(j)+thdd(j) ...
  )+l2f2(j).*((-0.1E1).*c2.*m2f2.*fy2(j).*sin(th(j)+th2f2(j))+ ...
  (0.5E0.*mr2f2.^2+0.5E0.*mr2f2.*ms+0.125E0.*ms.^2+0.2E1.* ...
  mr2f2.*mt+0.1E1.*ms.*mt+0.2E1.*mt.^2).*l2df2(j).*(th2df2(j)+ ...
  thd(j))+m2f2.*(g.*(0.5E0.*mr2f2+0.25E0.*ms+0.1E1.*mt).*sin( ...
  th(j)+th2f2(j))+0.5E0.*l1f2.*mr2f2.*cos(th1f2(j)+(-1).* ...
  th2f2(j)).*th1ddf2(j)+0.25E0.*l1f2.*ms.*cos(th1f2(j)+(-1).* ...
  th2f2(j)).*th1ddf2(j)+0.1E1.*l1f2.*mt.*cos(th1f2(j)+(-1).* ...
  th2f2(j)).*th1ddf2(j)+l1f2.*((-0.5E0).*mr2f2+(-0.25E0).*ms+( ...
  -0.1E1).*mt).*sin(th1f2(j)+(-1).*th2f2(j)).*th1df2(j).^2+ ...
  l1f2.*((-0.1E1).*mr2f2+(-0.5E0).*ms+(-0.2E1).*mt).*sin( ...
  th1f2(j)+(-1).*th2f2(j)).*th1df2(j).*thd(j)+((-0.5E0).* ...
  mr2f2+(-0.25E0).*ms+(-0.1E1).*mt).*(d.*cos(th2f2(j))+l1f2.* ...
  sin(th1f2(j)+(-1).*th2f2(j))).*thd(j).^2+0.5E0.*l1f2.* ...
  mr2f2.*cos(th1f2(j)+(-1).*th2f2(j)).*thdd(j)+0.25E0.*l1f2.* ...
  ms.*cos(th1f2(j)+(-1).*th2f2(j)).*thdd(j)+0.1E1.*l1f2.*mt.* ...
  cos(th1f2(j)+(-1).*th2f2(j)).*thdd(j)+0.5E0.*d.*mr2f2.*sin( ...
  th2f2(j)).*thdd(j)+0.25E0.*d.*ms.*sin(th2f2(j)).*thdd(j)+ ...
  0.1E1.*d.*mt.*sin(th2f2(j)).*thdd(j)+0.5E0.*mr2f2.*cos(th(j) ...
  +th2f2(j)).*xdd(j)+0.25E0.*ms.*cos(th(j)+th2f2(j)).*xdd(j)+ ...
  0.1E1.*mt.*cos(th(j)+th2f2(j)).*xdd(j)+0.5E0.*mr2f2.*sin(th( ...
  j)+th2f2(j)).*ydd(j)+0.25E0.*ms.*sin(th(j)+th2f2(j)).*ydd(j) ...
  +0.1E1.*mt.*sin(th(j)+th2f2(j)).*ydd(j))));


torque1f3(j)=(-0.1E1).*c3.*l1f3.*cos(th(j)+th1f3(j)).*fx3(j)+g.*...
    (0.1E1.*lcm1f3.*m1f3+0.1E1.*l1f3.*m2f3).*sin(th(j)+th1f3(j))+( ...
  -0.1E1).*c3.*l1f3.*fy3(j).*sin(th(j)+th1f3(j))+(-0.5E0).* ...
  l1f3.*mr2f3.*l2ddf3(j).*sin(th1f3(j)+(-1).*th2f3(j))+( ...
  -0.25E0).*l1f3.*ms.*l2ddf3(j).*sin(th1f3(j)+(-1).*th2f3(j))+ ...
  (-0.1E1).*l1f3.*mt.*l2ddf3(j).*sin(th1f3(j)+(-1).*th2f3(j))+ ...
  0.1E1.*I1f3.*th1ddf3(j)+0.1E1.*lcm1f3.^2.*m1f3.*th1ddf3(j)+ ...
  0.1E1.*l1f3.^2.*m2f3.*th1ddf3(j)+0.5E0.*l1f3.*mr2f3.*cos( ...
  th1f3(j)+(-1).*th2f3(j)).*l2f3(j).*th2ddf3(j)+0.25E0.*l1f3.* ...
  ms.*cos(th1f3(j)+(-1).*th2f3(j)).*l2f3(j).*th2ddf3(j)+ ...
  0.1E1.*l1f3.*mt.*cos(th1f3(j)+(-1).*th2f3(j)).*l2f3(j).* ...
  th2ddf3(j)+0.1E1.*l1f3.*mr2f3.*cos(th1f3(j)+(-1).*th2f3(j)) ...
```

```
  .*l2df3(j).*th2df3(j)+0.5E0.*l1f3.*ms.*cos(th1f3(j)+(-1).* ...
  th2f3(j)).*l2df3(j).*th2df3(j)+0.2E1.*l1f3.*mt.*cos(th1f3(j) ...
  +(-1).*th2f3(j)).*l2df3(j).*th2df3(j)+0.5E0.*l1f3.*mr2f3.* ...
  l2f3(j).*sin(th1f3(j)+(-1).*th2f3(j)).*th2df3(j).^2+0.25E0.* ...
  l1f3.*ms.*l2f3(j).*sin(th1f3(j)+(-1).*th2f3(j)).*th2df3(j) ...
  .^2+0.1E1.*l1f3.*mt.*l2f3(j).*sin(th1f3(j)+(-1).*th2f3(j)).* ...
  th2df3(j).^2+0.1E1.*l1f3.*mr2f3.*cos(th1f3(j)+(-1).*th2f3(j) ...
  ).*l2df3(j).*thd(j)+0.5E0.*l1f3.*ms.*cos(th1f3(j)+(-1).* ...
  th2f3(j)).*l2df3(j).*thd(j)+0.2E1.*l1f3.*mt.*cos(th1f3(j)+( ...
  -1).*th2f3(j)).*l2df3(j).*thd(j)+0.1E1.*l1f3.*mr2f3.*l2f3(j) ...
  .*sin(th1f3(j)+(-1).*th2f3(j)).*th2df3(j).*thd(j)+0.5E0.* ...
  l1f3.*ms.*l2f3(j).*sin(th1f3(j)+(-1).*th2f3(j)).*th2df3(j).* ...
  thd(j)+0.2E1.*l1f3.*mt.*l2f3(j).*sin(th1f3(j)+(-1).*th2f3(j) ...
  ).*th2df3(j).*thd(j)+(-0.1E1).*d.*lcm1f3.*m1f3.*cos(th1f3(j) ...
  ).*thd(j).^2+(-0.1E1).*d.*l1f3.*m2f3.*cos(th1f3(j)).*thd(j) ...
  .^2+0.5E0.*l1f3.*mr2f3.*l2f3(j).*sin(th1f3(j)+(-1).*th2f3(j) ...
  ).*thd(j).^2+0.25E0.*l1f3.*ms.*l2f3(j).*sin(th1f3(j)+(-1).* ...
  th2f3(j)).*thd(j).^2+0.1E1.*l1f3.*mt.*l2f3(j).*sin(th1f3(j)+ ...
  (-1).*th2f3(j)).*thd(j).^2+0.1E1.*lcm1f3.^2.*m1f3.*thdd(j)+ ...
  0.1E1.*l1f3.^2.*m2f3.*thdd(j)+0.5E0.*l1f3.*mr2f3.*cos(th1f3( ...
  j)+(-1).*th2f3(j)).*l2f3(j).*thdd(j)+0.25E0.*l1f3.*ms.*cos( ...
  th1f3(j)+(-1).*th2f3(j)).*l2f3(j).*thdd(j)+0.1E1.*l1f3.*mt.* ...
  cos(th1f3(j)+(-1).*th2f3(j)).*l2f3(j).*thdd(j)+0.1E1.*d.* ...
  lcm1f3.*m1f3.*sin(th1f3(j)).*thdd(j)+0.1E1.*d.*l1f3.*m2f3.* ...
  sin(th1f3(j)).*thdd(j)+0.1E1.*lcm1f3.*m1f3.*cos(th(j)+th1f3( ...
  j)).*xdd(j)+0.1E1.*l1f3.*m2f3.*cos(th(j)+th1f3(j)).*xdd(j)+( ...
  0.1E1.*lcm1f3.*m1f3+0.1E1.*l1f3.*m2f3).*sin(th(j)+th1f3(j)) ...
  .*ydd(j);

torque2f3(j)=m2f3.^(-1).*((-0.1E1).*c3.*m2f3.*cos(th(j)+th2f3(j)).*...
    fx3(j).*l2f3(j)+0.1E1.*m2f3.*I2f3(j).*th2ddf3(j)+(0.25E0.* ...
  mr2f3.^2+0.25E0.*mr2f3.*ms+0.625E-1.*ms.^2+0.1E1.*mr2f3.*mt+ ...
  0.5E0.*ms.*mt+0.1E1.*mt.^2).*l2f3(j).^2.*(th2ddf3(j)+thdd(j) ...
  )+l2f3(j).*((-0.1E1).*c3.*m2f3.*fy3(j).*sin(th(j)+th2f3(j))+ ...
  (0.5E0.*mr2f3.^2+0.5E0.*mr2f3.*ms+0.125E0.*ms.^2+0.2E1.* ...
  mr2f3.*mt+0.1E1.*ms.*mt+0.2E1.*mt.^2).*l2df3(j).*(th2df3(j)+ ...
  thd(j))+m2f3.*(g.*(0.5E0.*mr2f3+0.25E0.*ms+0.1E1.*mt).*sin( ...
  th(j)+th2f3(j))+0.5E0.*l1f3.*mr2f3.*cos(th1f3(j)+(-1).* ...
  th2f3(j)).*th1ddf3(j)+0.25E0.*l1f3.*ms.*cos(th1f3(j)+(-1).* ...
  th2f3(j)).*th1ddf3(j)+0.1E1.*l1f3.*mt.*cos(th1f3(j)+(-1).* ...
  th2f3(j)).*th1ddf3(j)+l1f3.*((-0.5E0).*mr2f3+(-0.25E0).*ms+( ...
  -0.1E1).*mt).*sin(th1f3(j)+(-1).*th2f3(j)).*th1df3(j).^2+ ...
  l1f3.*((-0.1E1).*mr2f3+(-0.5E0).*ms+(-0.2E1).*mt).*sin( ...
  th1f3(j)+(-1).*th2f3(j)).*th1df3(j).*thd(j)+((-0.5E0).* ...
  mr2f3+(-0.25E0).*ms+(-0.1E1).*mt).*(d.*cos(th2f3(j))+l1f3.* ...
  sin(th1f3(j)+(-1).*th2f3(j))).*thd(j).^2+0.5E0.*l1f3.* ...
  mr2f3.*cos(th1f3(j)+(-1).*th2f3(j)).*thdd(j)+0.25E0.*l1f3.* ...
  ms.*cos(th1f3(j)+(-1).*th2f3(j)).*thdd(j)+0.1E1.*l1f3.*mt.* ...
  cos(th1f3(j)+(-1).*th2f3(j)).*thdd(j)+0.5E0.*d.*mr2f3.*sin( ...
  th2f3(j)).*thdd(j)+0.25E0.*d.*ms.*sin(th2f3(j)).*thdd(j)+ ...
  0.1E1.*d.*mt.*sin(th2f3(j)).*thdd(j)+0.5E0.*mr2f3.*cos(th(j) ...
  +th2f3(j)).*xdd(j)+0.25E0.*ms.*cos(th(j)+th2f3(j)).*xdd(j)+ ...
  0.1E1.*mt.*cos(th(j)+th2f3(j)).*xdd(j)+0.5E0.*mr2f3.*sin(th( ...
  j)+th2f3(j)).*ydd(j)+0.25E0.*ms.*sin(th(j)+th2f3(j)).*ydd(j) ...
  +0.1E1.*mt.*sin(th(j)+th2f3(j)).*ydd(j))));


end
%-------------------------------------------------------------------------%
end
```

check_strength.m

```
function [din,dout,flag] = check_strength(fx,fy,th,th1h1,l1h1,th2h1,l2h1...
    ,th1f2,l1f2,th2f2,l2f2,th1f3,l1f3,th2f3,l2f3,th1h4,l1h4,th2h4,l2h4)
%CHECK_STRENGTH takes as inputs the forces from the ground, the leg
%segment lengths and the joint trajectories and returns as output the inner
%and outer diameters of the tubular links if material strength is respected
%(flag==0). Four alternative ID/OD options should be provided from the
%user. If strength constraints are not valid for all of four alternatives,
%(flag==1), the user should provide ID/OD with thicker cross-sectional
%areas.

flag=1;
%------------------------------- OPTION 1 --------------------------------%
din_c=0.023;
dout_c=0.025;

A=pi*(dout_c^2-din_c^2)/4;
strength=200;

ft21=fx(1:500)/2.*cos(th(1:500)+th2h1(1:500))+fy(1:500)/2.*...
    sin(th(1:500)+th2h1(1:500));
fl21=fx(1:500)/2.*sin(th(1:500)+th2h1(1:500))-fy(1:500)/2.*...
    cos(th(1:500)+th2h1(1:500));
M21=ft21.*l2h1(1:500);
I21=pi*(dout_c^4-din_c^4)/64;
sb21=M21/I21*dout_c/2/10^6;
sc21=fl21/A/10^6;
ss21=ft21/A/10^6;

ft11=fx(1:500)/2.*cos(th(1:500)+th1h1(1:500))+fy(1:500)/2.*...
    sin(th(1:500)+th1h1(1:500));
fl11=fx(1:500)/2.*sin(th(1:500)+th1h1(1:500))-fy(1:500)/2.*...
    cos(th(1:500)+th1h1(1:500));
M11=ft11.*l1h1;
I11=pi*(dout_c^4-din_c^4)/64;
sb11=M11/I11*dout_c/2/10^6;
sc11=fl11/A/10^6;
ss11=ft11/A/10^6;

ft22=fx(501:1000)/2.*cos(th(501:1000)+th2f2(501:1000))+fy(501:1000)/2.*...
    sin(th(501:1000)+th2f2(501:1000));
fl22=fx(501:1000)/2.*sin(th(501:1000)+th2f2(501:1000))-fy(501:1000)/2.*...
    cos(th(501:1000)+th2f2(501:1000));
M22=ft22.*l2f2(501:1000);
I22=pi*(dout_c^4-din_c^4)/64;
sb22=M22/I22*dout_c/2/10^6;
sc22=fl22/A/10^6;
ss22=ft22/A/10^6;

ft12=fx(501:1000)/2.*cos(th(501:1000)+th1f2(501:1000))+fy(501:1000)/2.*...
    sin(th(501:1000)+th1f2(501:1000));
fl12=fx(501:1000)/2.*sin(th(501:1000)+th1f2(501:1000))-fy(501:1000)/2.*...
    cos(th(501:1000)+th1f2(501:1000));
M12=ft12.*l1f2;
I12=pi*(dout_c^4-din_c^4)/64;
sb12=M12/I12*dout_c/2/10^6;
sc12=fl12/A/10^6;
ss12=ft12/A/10^6;
```

```
ft23=fx(1:500)/2.*cos(th(1:500)+th2f3(1:500))+fy(1:500)/2.*...
    sin(th(1:500)+th2f3(1:500));
fl23=fx(1:500)/2.*sin(th(1:500)+th2f3(1:500))-fy(1:500)/2.*...
    cos(th(1:500)+th2f3(1:500));
M23=ft23.*l2f3(1:500);
I23=pi*(dout_c^4-din_c^4)/64;
sb23=M23/I23*dout_c/2/10^6;
sc23=fl23/A/10^6;
ss23=ft23/A/10^6;

ft13=fx(1:500)/2.*cos(th(1:500)+th1f3(1:500))+fy(1:500)/2.*...
    sin(th(1:500)+th1f3(1:500));
fl13=fx(1:500)/2.*sin(th(1:500)+th1f3(1:500))-fy(1:500)/2.*...
    cos(th(1:500)+th1f3(1:500));
M13=ft13.*l1f3;
I13=pi*(dout_c^4-din_c^4)/64;
sb13=M13/I13*dout_c/2/10^6;
sc13=fl13/A/10^6;
ss13=ft13/A/10^6;

ft24=fx(501:1000)/2.*cos(th(501:1000)+th2h4(501:1000))+fy(501:1000)/2.*...
    sin(th(501:1000)+th2h4(501:1000));
fl24=fx(501:1000)/2.*sin(th(501:1000)+th2h4(501:1000))-fy(501:1000)/2.*...
    cos(th(501:1000)+th2h4(501:1000));
M24=ft24.*l2h4(501:1000);
I24=pi*(dout_c^4-din_c^4)/64;
sb24=M24/I24*dout_c/2/10^6;
sc24=fl24/A/10^6;
ss24=ft24/A/10^6;

ft14=fx(501:1000)/2.*cos(th(501:1000)+th1h4(501:1000))+fy(501:1000)/2.*...
    sin(th(501:1000)+th1h4(501:1000));
fl14=fx(501:1000)/2.*sin(th(501:1000)+th1h4(501:1000))-fy(501:1000)/2.*...
    cos(th(501:1000)+th1h4(501:1000));
M14=ft14.*l1h4;
I14=pi*(dout_c^4-din_c^4)/64;
sb14=M14/I14*dout_c/2/10^6;
sc14=fl14/A/10^6;
ss14=ft14/A/10^6;

if strength/max(abs(sb21)+abs(sc21))>=3&&strength/max(abs(sb11)+...
        abs(sc11))>=3&&strength/max(abs(sb22)+abs(sc22))>=3&&strength/...
        max(abs(sb12)+abs(sc12))>=3&&strength/max(abs(sb23)+abs(sc23))>=...
        3&&strength/max(abs(sb13)+abs(sc13))>=3&&strength/max(abs(sb24)+...
        abs(sc24))>=3&&strength/max(abs(sb14)+abs(sc14))>=3

    din=din_c;
    dout=dout_c;
    flag=0;
end
%-------------------------------------------------------------------------%
%------------------------------ OPTION 2 ---------------------------------%
if flag==1
din_c=0.026;
dout_c=0.028;

A=pi*(dout_c^2-din_c^2)/4;
strength=200;

ft21=fx(1:500)/2.*cos(th(1:500)+th2h1(1:500))+fy(1:500)/2.*...
```

```
    sin(th(1:500)+th2h1(1:500));
fl21=fx(1:500)/2.*sin(th(1:500)+th2h1(1:500))-fy(1:500)/2.*...
    cos(th(1:500)+th2h1(1:500));
M21=ft21.*l2h1(1:500);
I21=pi*(dout_c^4-din_c^4)/64;
sb21=M21/I21*dout_c/2/10^6;
sc21=fl21/A/10^6;
ss21=ft21/A/10^6;

ft11=fx(1:500)/2.*cos(th(1:500)+th1h1(1:500))+fy(1:500)/2.*...
    sin(th(1:500)+th1h1(1:500));
fl11=fx(1:500)/2.*sin(th(1:500)+th1h1(1:500))-fy(1:500)/2.*...
    cos(th(1:500)+th1h1(1:500));
M11=ft11.*l1h1;
I11=pi*(dout_c^4-din_c^4)/64;
sb11=M11/I11*dout_c/2/10^6;
sc11=fl11/A/10^6;
ss11=ft11/A/10^6;

ft22=fx(501:1000)/2.*cos(th(501:1000)+th2f2(501:1000))+fy(501:1000)/2.*...
    sin(th(501:1000)+th2f2(501:1000));
fl22=fx(501:1000)/2.*sin(th(501:1000)+th2f2(501:1000))-fy(501:1000)/2.*...
    cos(th(501:1000)+th2f2(501:1000));
M22=ft22.*l2f2(501:1000);
I22=pi*(dout_c^4-din_c^4)/64;
sb22=M22/I22*dout_c/2/10^6;
sc22=fl22/A/10^6;
ss22=ft22/A/10^6;

ft12=fx(501:1000)/2.*cos(th(501:1000)+th1f2(501:1000))+fy(501:1000)/2.*...
    sin(th(501:1000)+th1f2(501:1000));
fl12=fx(501:1000)/2.*sin(th(501:1000)+th1f2(501:1000))-fy(501:1000)/2.*...
    cos(th(501:1000)+th1f2(501:1000));
M12=ft12.*l1f2;
I12=pi*(dout_c^4-din_c^4)/64;
sb12=M12/I12*dout_c/2/10^6;
sc12=fl12/A/10^6;
ss12=ft12/A/10^6;

ft23=fx(1:500)/2.*cos(th(1:500)+th2f3(1:500))+fy(1:500)/2.*...
    sin(th(1:500)+th2f3(1:500));
fl23=fx(1:500)/2.*sin(th(1:500)+th2f3(1:500))-fy(1:500)/2.*...
    cos(th(1:500)+th2f3(1:500));
M23=ft23.*l2f3(1:500);
I23=pi*(dout_c^4-din_c^4)/64;
sb23=M23/I23*dout_c/2/10^6;
sc23=fl23/A/10^6;
ss23=ft23/A/10^6;

ft13=fx(1:500)/2.*cos(th(1:500)+th1f3(1:500))+fy(1:500)/2.*...
    sin(th(1:500)+th1f3(1:500));
fl13=fx(1:500)/2.*sin(th(1:500)+th1f3(1:500))-fy(1:500)/2.*...
    cos(th(1:500)+th1f3(1:500));
M13=ft13.*l1f3;
I13=pi*(dout_c^4-din_c^4)/64;
sb13=M13/I13*dout_c/2/10^6;
sc13=fl13/A/10^6;
ss13=ft13/A/10^6;

ft24=fx(501:1000)/2.*cos(th(501:1000)+th2h4(501:1000))+fy(501:1000)/2.*...
```

```
    sin(th(501:1000)+th2h4(501:1000));
fl24=fx(501:1000)/2.*sin(th(501:1000)+th2h4(501:1000))-fy(501:1000)/2.*...
    cos(th(501:1000)+th2h4(501:1000));
M24=ft24.*l2h4(501:1000);
I24=pi*(dout_c^4-din_c^4)/64;
sb24=M24/I24*dout_c/2/10^6;
sc24=fl24/A/10^6;
ss24=ft24/A/10^6;

ft14=fx(501:1000)/2.*cos(th(501:1000)+th1h4(501:1000))+fy(501:1000)/2.*...
    sin(th(501:1000)+th1h4(501:1000));
fl14=fx(501:1000)/2.*sin(th(501:1000)+th1h4(501:1000))-fy(501:1000)/2.*...
    cos(th(501:1000)+th1h4(501:1000));
M14=ft14.*l1h4;
I14=pi*(dout_c^4-din_c^4)/64;
sb14=M14/I14*dout_c/2/10^6;
sc14=fl14/A/10^6;
ss14=ft14/A/10^6;

if strength/max(abs(sb21)+abs(sc21))>=3&&strength/max(abs(sb11)+...
        abs(sc11))>=3&&strength/max(abs(sb22)+abs(sc22))>=3&&strength/...
        max(abs(sb12)+abs(sc12))>=3&&strength/max(abs(sb23)+abs(sc23))>=...
        3&&strength/max(abs(sb13)+abs(sc13))>=3&&strength/max(abs(sb24)+...
        abs(sc24))>=3&&strength/max(abs(sb14)+abs(sc14))>=3

    din=din_c;
    dout=dout_c;
    flag=0;
end

end
%-------------------------------------------------------------------------%
%------------------------------ OPTION 3 ---------------------------------%
if flag==1
din_c=0.028;
dout_c=0.03;

A=pi*(dout_c^2-din_c^2)/4;
strength=200;

ft21=fx(1:500)/2.*cos(th(1:500)+th2h1(1:500))+fy(1:500)/2.*...
    sin(th(1:500)+th2h1(1:500));
fl21=fx(1:500)/2.*sin(th(1:500)+th2h1(1:500))-fy(1:500)/2.*...
    cos(th(1:500)+th2h1(1:500));
M21=ft21.*l2h1(1:500);
I21=pi*(dout_c^4-din_c^4)/64;
sb21=M21/I21*dout_c/2/10^6;
sc21=fl21/A/10^6;
ss21=ft21/A/10^6;

ft11=fx(1:500)/2.*cos(th(1:500)+th1h1(1:500))+fy(1:500)/2.*...
    sin(th(1:500)+th1h1(1:500));
fl11=fx(1:500)/2.*sin(th(1:500)+th1h1(1:500))-fy(1:500)/2.*...
    cos(th(1:500)+th1h1(1:500));
M11=ft11.*l1h1;
I11=pi*(dout_c^4-din_c^4)/64;
sb11=M11/I11*dout_c/2/10^6;
sc11=fl11/A/10^6;
ss11=ft11/A/10^6;
```

```
ft22=fx(501:1000)/2.*cos(th(501:1000)+th2f2(501:1000))+fy(501:1000)/2.*...
    sin(th(501:1000)+th2f2(501:1000));
fl22=fx(501:1000)/2.*sin(th(501:1000)+th2f2(501:1000))-fy(501:1000)/2.*...
    cos(th(501:1000)+th2f2(501:1000));
M22=ft22.*l2f2(501:1000);
I22=pi*(dout_c^4-din_c^4)/64;
sb22=M22/I22*dout_c/2/10^6;
sc22=fl22/A/10^6;
ss22=ft22/A/10^6;

ft12=fx(501:1000)/2.*cos(th(501:1000)+th1f2(501:1000))+fy(501:1000)/2.*...
    sin(th(501:1000)+th1f2(501:1000));
fl12=fx(501:1000)/2.*sin(th(501:1000)+th1f2(501:1000))-fy(501:1000)/2.*...
    cos(th(501:1000)+th1f2(501:1000));
M12=ft12.*l1f2;
I12=pi*(dout_c^4-din_c^4)/64;
sb12=M12/I12*dout_c/2/10^6;
sc12=fl12/A/10^6;
ss12=ft12/A/10^6;

ft23=fx(1:500)/2.*cos(th(1:500)+th2f3(1:500))+fy(1:500)/2.*...
    sin(th(1:500)+th2f3(1:500));
fl23=fx(1:500)/2.*sin(th(1:500)+th2f3(1:500))-fy(1:500)/2.*...
    cos(th(1:500)+th2f3(1:500));
M23=ft23.*l2f3(1:500);
I23=pi*(dout_c^4-din_c^4)/64;
sb23=M23/I23*dout_c/2/10^6;
sc23=fl23/A/10^6;
ss23=ft23/A/10^6;

ft13=fx(1:500)/2.*cos(th(1:500)+th1f3(1:500))+fy(1:500)/2.*...
    sin(th(1:500)+th1f3(1:500));
fl13=fx(1:500)/2.*sin(th(1:500)+th1f3(1:500))-fy(1:500)/2.*...
    cos(th(1:500)+th1f3(1:500));
M13=ft13.*l1f3;
I13=pi*(dout_c^4-din_c^4)/64;
sb13=M13/I13*dout_c/2/10^6;
sc13=fl13/A/10^6;
ss13=ft13/A/10^6;

ft24=fx(501:1000)/2.*cos(th(501:1000)+th2h4(501:1000))+fy(501:1000)/2.*...
    sin(th(501:1000)+th2h4(501:1000));
fl24=fx(501:1000)/2.*sin(th(501:1000)+th2h4(501:1000))-fy(501:1000)/2.*...
    cos(th(501:1000)+th2h4(501:1000));
M24=ft24.*l2h4(501:1000);
I24=pi*(dout_c^4-din_c^4)/64;
sb24=M24/I24*dout_c/2/10^6;
sc24=fl24/A/10^6;
ss24=ft24/A/10^6;

ft14=fx(501:1000)/2.*cos(th(501:1000)+th1h4(501:1000))+fy(501:1000)/2.*...
    sin(th(501:1000)+th1h4(501:1000));
fl14=fx(501:1000)/2.*sin(th(501:1000)+th1h4(501:1000))-fy(501:1000)/2.*...
    cos(th(501:1000)+th1h4(501:1000));
M14=ft14.*l1h4;
I14=pi*(dout_c^4-din_c^4)/64;
sb14=M14/I14*dout_c/2/10^6;
sc14=fl14/A/10^6;
ss14=ft14/A/10^6;
```

```
if strength/max(abs(sb21)+abs(sc21))>=3&&strength/max(abs(sb11)+...
        abs(sc11))>=3&&strength/max(abs(sb22)+abs(sc22))>=3&&...
        strength/max(abs(sb12)+abs(sc12))>=3&&strength/max(abs(sb23)...
        +abs(sc23))>=3&&strength/max(abs(sb13)+abs(sc13))>=3&&...
        strength/max(abs(sb24)+abs(sc24))>=3&&strength/max(abs(sb14)+...
        abs(sc14))>=3

    din=din_c;
    dout=dout_c;
    flag=0;
end

end
%-------------------------------------------------------------------------%
%------------------------------ OPTION 4 ---------------------------------%
if flag==1
    din_c=0.026;
    dout_c=0.03;

    A=pi*(dout_c^2-din_c^2)/4;
    strength=200;

    ft21=fx(1:500)/2.*cos(th(1:500)+th2h1(1:500))+fy(1:500)/2.*...
        sin(th(1:500)+th2h1(1:500));
    fl21=fx(1:500)/2.*sin(th(1:500)+th2h1(1:500))-fy(1:500)/2.*...
        cos(th(1:500)+th2h1(1:500));
    M21=ft21.*l2h1(1:500);
    I21=pi*(dout_c^4-din_c^4)/64;
    sb21=M21/I21*dout_c/2/10^6;
    sc21=fl21/A/10^6;
    ss21=ft21/A/10^6;

    ft11=fx(1:500)/2.*cos(th(1:500)+th1h1(1:500))+fy(1:500)/2.*...
        sin(th(1:500)+th1h1(1:500));
    fl11=fx(1:500)/2.*sin(th(1:500)+th1h1(1:500))-fy(1:500)/2.*...
        cos(th(1:500)+th1h1(1:500));
    M11=ft11.*l1h1;
    I11=pi*(dout_c^4-din_c^4)/64;
    sb11=M11/I11*dout_c/2/10^6;
    sc11=fl11/A/10^6;
    ss11=ft11/A/10^6;

    ft22=fx(501:1000)/2.*cos(th(501:1000)+th2f2(501:1000))+...
        fy(501:1000)/2.*sin(th(501:1000)+th2f2(501:1000));
    fl22=fx(501:1000)/2.*sin(th(501:1000)+th2f2(501:1000))-...
        fy(501:1000)/2.*cos(th(501:1000)+th2f2(501:1000));
    M22=ft22.*l2f2(501:1000);
    I22=pi*(dout_c^4-din_c^4)/64;
    sb22=M22/I22*dout_c/2/10^6;
    sc22=fl22/A/10^6;
    ss22=ft22/A/10^6;

    ft12=fx(501:1000)/2.*cos(th(501:1000)+th1f2(501:1000))+...
        fy(501:1000)/2.*sin(th(501:1000)+th1f2(501:1000));
    fl12=fx(501:1000)/2.*sin(th(501:1000)+th1f2(501:1000))-...
        fy(501:1000)/2.*cos(th(501:1000)+th1f2(501:1000));
    M12=ft12.*l1f2;
    I12=pi*(dout_c^4-din_c^4)/64;
    sb12=M12/I12*dout_c/2/10^6;
    sc12=fl12/A/10^6;
```

```
    ss12=ft12/A/10^6;

    ft23=fx(1:500)/2.*cos(th(1:500)+th2f3(1:500))+fy(1:500)/2.*...
        sin(th(1:500)+th2f3(1:500));
    fl23=fx(1:500)/2.*sin(th(1:500)+th2f3(1:500))-fy(1:500)/2.*...
        cos(th(1:500)+th2f3(1:500));
    M23=ft23.*l2f3(1:500);
    I23=pi*(dout_c^4-din_c^4)/64;
    sb23=M23/I23*dout_c/2/10^6;
    sc23=fl23/A/10^6;
    ss23=ft23/A/10^6;

    ft13=fx(1:500)/2.*cos(th(1:500)+th1f3(1:500))+fy(1:500)/2.*...
        sin(th(1:500)+th1f3(1:500));
    fl13=fx(1:500)/2.*sin(th(1:500)+th1f3(1:500))-fy(1:500)/2.*...
        cos(th(1:500)+th1f3(1:500));
    M13=ft13.*l1f3;
    I13=pi*(dout_c^4-din_c^4)/64;
    sb13=M13/I13*dout_c/2/10^6;
    sc13=fl13/A/10^6;
    ss13=ft13/A/10^6;

    ft24=fx(501:1000)/2.*cos(th(501:1000)+th2h4(501:1000))+...
        fy(501:1000)/2.*sin(th(501:1000)+th2h4(501:1000));
    fl24=fx(501:1000)/2.*sin(th(501:1000)+th2h4(501:1000))-...
        fy(501:1000)/2.*cos(th(501:1000)+th2h4(501:1000));
    M24=ft24.*l2h4(501:1000);
    I24=pi*(dout_c^4-din_c^4)/64;
    sb24=M24/I24*dout_c/2/10^6;
    sc24=fl24/A/10^6;
    ss24=ft24/A/10^6;

    ft14=fx(501:1000)/2.*cos(th(501:1000)+th1h4(501:1000))+...
        fy(501:1000)/2.*sin(th(501:1000)+th1h4(501:1000));
    fl14=fx(501:1000)/2.*sin(th(501:1000)+th1h4(501:1000))-...
        fy(501:1000)/2.*cos(th(501:1000)+th1h4(501:1000));
    M14=ft14.*l1h4;
    I14=pi*(dout_c^4-din_c^4)/64;
    sb14=M14/I14*dout_c/2/10^6;
    sc14=fl14/A/10^6;
    ss14=ft14/A/10^6;

    if strength/max(abs(sb21)+abs(sc21))>=3&&strength/max(abs(sb11)+...
            abs(sc11))>=3&&strength/max(abs(sb22)+abs(sc22))>=3&&...
            strength/max(abs(sb12)+abs(sc12))>=3&&...
            strength/max(abs(sb23)+abs(sc23))>=3&&...
            strength/max(abs(sb13)+abs(sc13))>=3&&...
            strength/max(abs(sb24)+abs(sc24))>=3&&...
            strength/max(abs(sb14)+abs(sc14))>=3

        din=din_c;
        dout=dout_c;
        flag=0;
%-------------------------------------------------------------------------%
    else
        din=din_c;
        dout=dout_c;
    end
end
end
```

Animation.m

```
%-------------------------------------------------------------------------%
%----------------------------- Animation.m -------------------------------%
%-------------------------------------------------------------------------%
% Produces an animation of the centroidal robot running for the duration of
% one stride. Run after PostProcessing.m.

f1 = figure(1);
clf(f1);

a=1;
b=0.15;
step=10;

% Stop button to stop animation
stop = uicontrol('style','toggle','string','stop','background','white');

for i = 1:step:n-1

    % If stop button is unpressed
    if get(stop,'value')==0

        % Background color (in+out plot)
        whitebg([1 1 1])

        % Plot Center of Mass
        plot(x(i),y(i),'o','MarkerSize',10)
        hold on
        grid on

        % Plot body circumference
        xe1=x(i)+a/2*cos(th(i))+b/2*sin(th(i));
        ye1=y(i)+a/2*sin(th(i))-b/2*cos(th(i));
        xe2=x(i)+a/2*cos(th(i))-b/2*sin(th(i));
        ye2=y(i)+a/2*sin(th(i))+b/2*cos(th(i));
        xe3=x(i)-a/2*cos(th(i))-b/2*sin(th(i));
        ye3=y(i)-a/2*sin(th(i))+b/2*cos(th(i));
        xe4=x(i)-a/2*cos(th(i))+b/2*sin(th(i));
        ye4=y(i)-a/2*sin(th(i))-b/2*cos(th(i));

        xcmf=[xe1;xe2;xe3;xe4;xe1];
        ycmf=[ye1;ye2;ye3;ye4;ye1];

        plot(xcmf,ycmf,'k')

        % Location of hips
        xh=x(i)-d*cos(th(i));
        yh=y(i)-d*sin(th(i));
        xf=x(i)+d*cos(th(i));
        yf=y(i)+d*sin(th(i));

        % Plot footfalls, CoM to toe and hip to toe lines.
        [c1,c2,c3,c4]=c_i(tt1,tt2,tt3,tt4,t(i),T);
        if c1==1
            plot(xh,yh,'bo','Markersize',5)
            plot([x(i) x1],[y(i) yg],':k')
            plot([x1 xh],[yg yh],'b')
        end
```

```
        if c2==1
            plot(xf,yf,'ro','Markersize',5)
            plot([x(i) x2],[y(i) yg],':k')
            plot([x2 xf],[yg yf],'r')
        end
        if c3==1
            plot(xf,yf,'co','Markersize',5)
            plot([x(i) x3],[y(i) yg],':k')
            plot([x3 xf],[yg yf],'c')
        end
        if c4==1
            plot(xh,yh,'mo','Markersize',5)
            plot([x(i) x4],[y(i) yg],':k')
            plot([x4 xh],[yg yh],'m')
        end

        % Plot ground
        plot([x(i)-1 x(i)+1],[yg yg],'Color',[0.5 0.5 0.5],'LineWidth',1)

        % Moving axis
        axis equal
        axis([x(i)-1 x(i)+1 yg-0.1 1.4])

        drawnow
        hold off


        % Break if Stop button is pressed
    elseif get(stop,'value')==1
        break
    end
%     hold off
end

% Turn the stop button into close button
set(stop,'style','pushbutton','string','close','callback','close(gcf)');
```

AnimationFull.m

```
%-------------------------------------------------------------------------%
%--------------------------- AnimationFull.m -----------------------------%
%-------------------------------------------------------------------------%
% Produces an animation of the quadruped robot running for the duration of
% one stride. Run after PostProcessing.m.


f2 = figure(2);
clf(f2);

step=10;

% Stop button to stop animation
stop = uicontrol('style','toggle','string','stop','background','white');

% Equally distributed forces
fxi=fx/2;
```

```
fyi=fy/2;
mafyi=max(abs(fyi));

for i = 1:step:n-1

    % If stop button is unpressed
    if get(stop,'value')==0

        % Background color (in+out plot)
        whitebg([1 1 1])

        % Plot Center of Mass
        plot(x(i),y(i),'o','MarkerSize',10)
        hold on
        grid on

        % Plot body circumferencei
        xe1=x(i)+a/2*cos(th(i))+b/2*sin(th(i));
        ye1=y(i)+a/2*sin(th(i))-b/2*cos(th(i));
        xe2=x(i)+a/2*cos(th(i))-b/2*sin(th(i));
        ye2=y(i)+a/2*sin(th(i))+b/2*cos(th(i));
        xe3=x(i)-a/2*cos(th(i))-b/2*sin(th(i));
        ye3=y(i)-a/2*sin(th(i))+b/2*cos(th(i));
        xe4=x(i)-a/2*cos(th(i))+b/2*sin(th(i));
        ye4=y(i)-a/2*sin(th(i))-b/2*cos(th(i));

        xcmf=[xe1;xe2;xe3;xe4;xe1];
        ycmf=[ye1;ye2;ye3;ye4;ye1];

        plot(xcmf,ycmf,'k')

        % Location of hips
        xh=x(i)-d*cos(th(i));
        yh=y(i)-d*sin(th(i));
        xf=x(i)+d*cos(th(i));
        yf=y(i)+d*sin(th(i));

        % Plot legs
        [c1,c2,c3,c4]=c_i(tt1,tt2,tt3,tt4,t(i),T);

        % leg 1
        xk1=xh+l1h1*sin(thh1(i)+th1h1(i));
        yk1=yh-l1h1*cos(thh1(i)+th1h1(i));
        xft1=xk1+l2h1(i)*sin(thh1(i)+th2h1(i));
        yft1=yk1-l2h1(i)*cos(thh1(i)+th2h1(i));
        plot(xh,yh,'bo','Markersize',5)
        plot([xft1 xh],[yft1 yh],':b')
        plot([xft1 xk1 xh],[yft1 yk1 yh],'b')

        % leg 2
        xk2=xf+l1f2*sin(thf2(i)+th1f2(i));
        yk2=yf-l1f2*cos(thf2(i)+th1f2(i));
        xft2=xk2+l2f2(i)*sin(thf2(i)+th2f2(i));
        yft2=yk2-l2f2(i)*cos(thf2(i)+th2f2(i));
        plot(xf,yf,'ro','Markersize',5)
        plot([xft2 xf],[yft2 yf],':r')
        plot([xft2 xk2 xf],[yft2 yk2 yf],'r')

        % leg 3
```

```
        xk3=xf+l1f3*sin(thf3(i)+th1f3(i));
        yk3=yf-l1f3*cos(thf3(i)+th1f3(i));
        xft3=xk3+l2f3(i)*sin(thf3(i)+th2f3(i));
        yft3=yk3-l2f3(i)*cos(thf3(i)+th2f3(i));
        plot(xf,yf,'co','Markersize',5)
        plot([xft3 xf],[yft3 yf],':c')
        plot([xft3 xk3 xf],[yft3 yk3 yf],'c')

        % leg 4
        xk4=xh+l1h4*sin(thh4(i)+th1h4(i));
        yk4=yh-l1h4*cos(thh4(i)+th1h4(i));
        xft4=xk4+l2h4(i)*sin(thh4(i)+th2h4(i));
        yft4=yk4-l2h4(i)*cos(thh4(i)+th2h4(i));
        plot(xh,yh,'mo','Markersize',5)
        plot([xft4 xh],[yft4 yh],':m')
        plot([xft4 xk4 xh],[yft4 yk4 yh],'m')

        % Plot ground
        plot([x(i)-1 x(i)+1],[yg yg],'Color',[0.5 0.5 0.5],'LineWidth',1)

        % Moving axis
        axis equal
        axis([x(i)-1 x(i)+1 yg-0.1 yg+1.4])

        % Plot Ground Force Vector
        if c1==1
            xf1=xft1+0.2*fxi(i)/mafyi;
            yf1=yft1+0.2*fyi(i)/mafyi;
            arrow([xft1,yft1],[xf1,yf1])
        end
        if c2==1
            xf2=xft2+0.2*fxi(i)/mafyi;
            yf2=yft2+0.2*fyi(i)/mafyi;
            arrow([xft2,yft2],[xf2,yf2])
        end
        if c3==1
            xf3=xft3+0.2*fxi(i)/mafyi;
            yf3=yft3+0.2*fyi(i)/mafyi;
            arrow([xft3,yft3],[xf3,yf3])
        end
        if c4==1
            xf4=xft4+0.2*fxi(i)/mafyi;
            yf4=yft4+0.2*fyi(i)/mafyi;
            arrow([xft4,yft4],[xf4,yf4])
        end

        drawnow
        hold off


        % Break if Stop button is pressed
    elseif get(stop,'value')==1
        break
    end
%     hold off
end

% Turn the stop button into close button
set(stop,'style','pushbutton','string','close','callback','close(gcf)');
```

PlotTorqueNRadPerSec.m

```
%-------------------------------------------------------------------------%
%----------------------- PlotTorquesNRadPerSec.m -------------------------%
%-------------------------------------------------------------------------%
% Plots joint torques and leg joint angular velocities with time for all
% legs and calculates and displays rms torques. Max values per joint are
% also calculated and displayed. All variables are expressed as a percenta-
% ge of the corresponding constraint. Run after PostProcessing.m

% torque_ji^*=torque_ji/taujmaxst
% thjdi*=thjdi/thjdmax

%-------------------------- short term torques ---------------------------%
figure(3)
plot(t/T*100,torque1h1/tau1maxst)
hold on
grid on
plot(t/T*100,torque2h1/tau2maxst)
plot(t/T*100,torque1h4/tau1maxst)
plot(t/T*100,torque2h4/tau2maxst)
title('Torques On Hind Leg Joints')
legend('torque_1_,_1^*','torque_2_,_1^*','torque_1_,_4^*','torque_2_,_4^*')
xlabel('time[% stride]')
hold off

figure(4)
plot(t/T*100,torque1f2/tau1maxst)
hold on
grid on
plot(t/T*100,torque2f2/tau2maxst)
plot(t/T*100,torque1f3/tau1maxst)
plot(t/T*100,torque2f3/tau2maxst)
title('Torques On Front Leg Joints')
legend('torque_1_,_2^*','torque_2_,_2^*','torque_1_,_3^*','torque_2_,_3^*')
xlabel('time[% stride]')
hold off
%-------------------------------------------------------------------------%
%------------------------- angular velocities ----------------------------%
figure(5)
plot(t/T*100,th1dh1/th1dmax)
hold on
grid on
plot(t/T*100,th2dh1/th2dmax)
plot(t/T*100,th1dh4/th1dmax)
plot(t/T*100,th2dh4/th2dmax)
title('Angular velocities On Hind Leg Joints')
legend('thd_1_,_1^*','thd_2_,_1^*','thd_1_,_4^*','thd_2_,_4^*')
xlabel('time[% stride]')
hold off

figure(6)
plot(t/T*100,th1df2/th1dmax)
hold on
grid on
plot(t/T*100,th2df2/th2dmax)
plot(t/T*100,th1df3/th1dmax)
plot(t/T*100,th2df3/th2dmax)
title('Angular velocities On Front Leg Joints')
legend('thd_1_,_2^*','thd_2_,_2^*','thd_1_,_3^*','thd_2_,_3^*')
xlabel('time[% stride]')
```

```
hold off
%-------------------------------------------------------------------------%
%----------------------------- rms torques -------------------------------%
ind11=rms(torque1h1)/tau1maxct;
ind21=rms(torque2h1)/tau2maxct;
ind12=rms(torque1f2)/tau1maxct;
ind22=rms(torque2f2)/tau2maxct;
ind13=rms(torque1f3)/tau1maxct;
ind23=rms(torque2f3)/tau2maxct;
ind14=rms(torque1h4)/tau1maxct;
ind24=rms(torque2h4)/tau2maxct;
%-------------------------------------------------------------------------%
%------------------------- max values per joint --------------------------%
mxthdhip=max(abs([th1dh1;th1df2;th1df3;th1dh4]))/th1dmax

mxthdkn=max(abs([th2dh1;th2df2;th2df3;th2dh4]))/th2dmax

mxsttauhip=max(abs([torque1h1;torque1f2;torque1f3;torque1h4]))/tau1maxst

mxsttaukn=max(abs([torque2h1;torque2f2;torque2f3;torque2h4]))/tau2maxst

mxcttauhip=max([rms(torque1h1);rms(torque1f2);rms(torque1f3);...
    rms(torque1h4)])/tau1maxct

mxcttaukn=max([rms(torque2h1);rms(torque2f2);rms(torque2f3);...
    rms(torque2h4)])/tau2maxct
%-------------------------------------------------------------------------%
```

PlotStress.m

```
%-------------------------------------------------------------------------%
%----------------------------- PlotStress.m ------------------------------%
%-------------------------------------------------------------------------%
% Displays in graphs the absolute stresses exerted on the tubular leg
% segments and compares them to the strength of the material.

A=pi*(dout^2-din^2)/4;
strength(1:500)=200;

%------------------------------- LEG 1 -----------------------------------%
%--- link 2 ---%
figure(7)
subplot(1,2,2)
ft21=fx(1:500)/2.*cos(th(1:500)+th2h1(1:500))+fy(1:500)/2.*...
    sin(th(1:500)+th2h1(1:500));
fl21=fx(1:500)/2.*sin(th(1:500)+th2h1(1:500))-fy(1:500)/2.*...
    cos(th(1:500)+th2h1(1:500));
M21=ft21.*l2h1(1:500);
I21=pi*(dout^4-din^4)/64;
sb21=M21/I21*dout/2/10^6;
sc21=fl21/A/10^6;
ss21=ft21/A/10^6;
plot(t(1:500),abs(sb21))
hold on
grid on
plot(t(1:500),abs(sc21))
plot(t(1:500),abs(ss21))
plot(t(1:500),strength,'--')
```

```
legend('bending','compresive','shear','bending strength','Location','best')
xlabel('time [s]')
ylabel('stress [MPa]')
ylim([0,220])
title('HL Leg, Segment 2')
hold off

%--- link 1 ---%
subplot(1,2,1)
ft11=fx(1:500)/2.*cos(th(1:500)+th1h1(1:500))+fy(1:500)/2.*...
    sin(th(1:500)+th1h1(1:500));
fl11=fx(1:500)/2.*sin(th(1:500)+th1h1(1:500))-fy(1:500)/2.*...
    cos(th(1:500)+th1h1(1:500));
M11=ft11.*l1h1;
I11=pi*(dout^4-din^4)/64;
sb11=M11/I11*dout/2/10^6;
sc11=fl11/A/10^6;
ss11=ft11/A/10^6;
plot(t(1:500),abs(sb11))
hold on
grid on
plot(t(1:500),abs(sc11))
plot(t(1:500),abs(ss11))
plot(t(1:500),strength,'--')
legend('bending','compresive','shear','bending strength','Location','best')
xlabel('time [s]')
ylabel('stress [MPa]')
ylim([0,220])
title('HL Leg, Segment 1')
hold off
%-------------------------------------------------------------------------%
%------------------------------ LEG 2 ------------------------------------%
%--- link 2 ---%
figure(8)
subplot(1,2,2)
ft22=fx(501:1000)/2.*cos(th(501:1000)+th2f2(501:1000))+fy(501:1000)/2.*...
    sin(th(501:1000)+th2f2(501:1000));
fl22=fx(501:1000)/2.*sin(th(501:1000)+th2f2(501:1000))-fy(501:1000)/2.*...
    cos(th(501:1000)+th2f2(501:1000));
M22=ft22.*l2f2(501:1000);
I22=pi*(dout^4-din^4)/64;
sb22=M22/I22*dout/2/10^6;
sc22=fl22/A/10^6;
ss22=ft22/A/10^6;
plot(t(501:1000),abs(sb22))
hold on
grid on
plot(t(501:1000),abs(sc22))
plot(t(501:1000),abs(ss22))
plot(t(501:1000),strength,'--')
legend('bending','compresive','shear','bending strength','Location','best')
xlabel('time [s]')
ylabel('stress [MPa]')
ylim([0,220])
title('FL Leg, Segment 2')
hold off

%--- link 1 ---%
subplot(1,2,1)
ft12=fx(501:1000)/2.*cos(th(501:1000)+th1f2(501:1000))+fy(501:1000)/2.*...
    sin(th(501:1000)+th1f2(501:1000));
```

```
fl12=fx(501:1000)/2.*sin(th(501:1000)+th1f2(501:1000))-fy(501:1000)/2.*...
    cos(th(501:1000)+th1f2(501:1000));
M12=ft12.*l1f2;
I12=pi*(dout^4-din^4)/64;
sb12=M12/I12*dout/2/10^6;
sc12=fl12/A/10^6;
ss12=ft12/A/10^6;
plot(t(501:1000),abs(sb12))
hold on
grid on
plot(t(501:1000),abs(sc12))
plot(t(501:1000),abs(ss12))
plot(t(501:1000),strength,'--')
legend('bending','compresive','shear','bending strength','Location','best')
xlabel('time [s]')
ylabel('stress [MPa]')
ylim([0,220])
title('FL Leg, Segment 1')
hold off
%-------------------------------------------------------------------------%
%------------------------------ LEG 3 ------------------------------------%
%--- link 2 ---%
figure(9)
subplot(1,2,2)
ft23=fx(1:500)/2.*cos(th(1:500)+th2f3(1:500))+fy(1:500)/2.*...
    sin(th(1:500)+th2f3(1:500));
fl23=fx(1:500)/2.*sin(th(1:500)+th2f3(1:500))-fy(1:500)/2.*...
    cos(th(1:500)+th2f3(1:500));
M23=ft23.*l2f3(1:500);
I23=pi*(dout^4-din^4)/64;
sb23=M23/I23*dout/2/10^6;
sc23=fl23/A/10^6;
ss23=ft23/A/10^6;
plot(t(1:500),abs(sb23))
hold on
grid on
plot(t(1:500),abs(sc23))
plot(t(1:500),abs(ss23))
plot(t(1:500),strength,'--')
legend('bending','compresive','shear','bending strength','Location','best')
xlabel('time [s]')
ylabel('stress [MPa]')
ylim([0,220])
title('FR Leg, Segment 2')
hold off

%--- link 1 ---%
subplot(1,2,1)
ft13=fx(1:500)/2.*cos(th(1:500)+th1f3(1:500))+fy(1:500)/2.*...
sin(th(1:500)+th1f3(1:500));
fl13=fx(1:500)/2.*sin(th(1:500)+th1f3(1:500))-fy(1:500)/2.*...
    cos(th(1:500)+th1f3(1:500));
M13=ft13.*l1f3;
I13=pi*(dout^4-din^4)/64;
sb13=M13/I13*dout/2/10^6;
sc13=fl13/A/10^6;
ss13=ft13/A/10^6;
plot(t(1:500),abs(sb13))
hold on
grid on
plot(t(1:500),abs(sc13))
```

```
plot(t(1:500),abs(ss13))
plot(t(1:500),strength,'--')
legend('bending','compresive','shear','bending strength','Location','best')
xlabel('time [s]')
ylabel('stress [MPa]')
ylim([0,220])
title('FR Leg, Segment 1')
hold off
%-------------------------------------------------------------------------%
%------------------------------ LEG 4 ------------------------------------%
%--- link 2 ---%
figure(10)
subplot(1,2,2)
ft24=fx(501:1000)/2.*cos(th(501:1000)+th2h4(501:1000))+fy(501:1000)/2.*...
    sin(th(501:1000)+th2h4(501:1000));
fl24=fx(501:1000)/2.*sin(th(501:1000)+th2h4(501:1000))-fy(501:1000)/2.*...
    cos(th(501:1000)+th2h4(501:1000));
M24=ft24.*l2h4(501:1000);
I24=pi*(dout^4-din^4)/64;
sb24=M24/I24*dout/2/10^6;
sc24=fl24/A/10^6;
ss24=ft24/A/10^6;
plot(t(501:1000),abs(sb24))
hold on
grid on
plot(t(501:1000),abs(sc24))
plot(t(501:1000),abs(ss24))
plot(t(501:1000),strength,'--')
legend('bending','compresive','shear','bending strength','Location','best')
xlabel('time [s]')
ylabel('stress [MPa]')
ylim([0,220])
title('HR Leg, Segment 2')
hold off

%--- link 1 ---%
subplot(1,2,1)
ft14=fx(501:1000)/2.*cos(th(501:1000)+th1h4(501:1000))+fy(501:1000)/2.*...
    sin(th(501:1000)+th1h4(501:1000));
fl14=fx(501:1000)/2.*sin(th(501:1000)+th1h4(501:1000))-fy(501:1000)/2.*...
    cos(th(501:1000)+th1h4(501:1000));
M14=ft14.*l1h4;
I14=pi*(dout^4-din^4)/64;
sb14=M14/I14*dout/2/10^6;
sc14=fl14/A/10^6;
ss14=ft14/A/10^6;
plot(t(501:1000),abs(sb14))
hold on
grid on
plot(t(501:1000),abs(sc14))
plot(t(501:1000),abs(ss14))
plot(t(501:1000),strength,'--')
legend('bending','compresive','shear','bending strength','Location','best')
xlabel('time [s]')
ylabel('stress [MPa]')
ylim([0,220])
title('HR Leg, Segment 1')
hold off
%-------------------------------------------------------------------------%
```