Optimization-based Motion Planning for Humanoid Fall Recovery

by

Shihao Wang

Department of Mechanical Engineering and Material Science
Duke University

Date: _____

Approved:

_____
Kris Hauser, Supervisor

_____
Brian Mann

_____
Leila Bridgeman

_____
Michael Zavlanos

_____
Thomas Witelski

Dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the Department of Mechanical Engineering and Material Science
in the Graduate School of Duke University
2020

Abstract

Optimization-based Motion Planning for Humanoid Fall Recovery

by

Shihao Wang

Department of Mechanical Engineering and Material Science
Duke University

Date: _____

Approved:

_____
Kris Hauser, Supervisor

_____
Brian Mann

_____
Leila Bridgeman

_____
Michael Zavlanos

_____
Thomas Witelski

An abstract of a dissertation submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the Department of Mechanical Engineering and Material Science
in the Graduate School of Duke University
2020

Abstract

Humanoid robots are created to look like humans, behave like humans, and ultimately reason in the same level as humans. Carrying on the hopes for emulating human beings' capabilities of performing dexterously and reliably in real world scenarios, humanoids are expected to undertake a wide variety of tasks including taking care of the elderly, doing house cleaning and disinfecting (especially important during COVID-19), and conducting operations in dangerous situations such as search and rescue during fire or earthquakes. The realization of these expectations of humanoid platforms needs continuous coordination of their upper and lower limbs in challenging environments and legged robots' inherent terrain adaptability makes them competent to provide assistance in these hazardous conditions. Though being considered as attractive candidates, humanoid robots suffer from a great risk of falling resulting from a relatively high center of mass position and a limited area of region of support. This prone-to-falling characteristic of humanoids' bipedal walking makes them much harder to control, and falls can cause costly failures. As a result, the ability to regain balance from falling is a prerequisite before humanoids can be confidently applied to execute significant tasks. Despite the rise of relevant research on humanoid fall recovery in recent decades, humanoid's self-balancing in response to unexpected disturbances in arbitrary environment remains to be a difficult problem due to humanoid's high degrees of freedom, complicated nonlinear system dynamics, and a "real-time" computational requirement owing to falling.

This dissertation focuses humanoid fall recovery with optimization-based motion planning approach. To advance state-of-the-art recovery strategies which mainly focus on open environment, I introduce motion planning algorithms which generalize fall recovery to both open and cluttered environments. I demonstrate two main contributions in this dissertation:

1. The development and implementation of an efficient motion planner which enables humanoid to recover from falling by making hand contact with walls or other surfaces in the cluttered environment. This approach extends humanoid's balancing capability to cluttered environment with making hand contact and this ability to make use of environmental object for fall prevention improves humanoids' efficiency and reliability.

2. The proposal and development of a multi-contact motion planner which generalizes humanoid fall recovery in both open and cluttered environment. This algorithm unifies existing recovery strategies, such as inertial shaping, protective stepping, and hand contact, and automatically plans one strategy or a combination of strategies to regain robot's balance based on its disturbed state and nearby environment features. By enabling humanoid to reason how to regain balance on its own, this algorithm makes a significant contribution to the improvement of humanoid's sustainability in arbitrary environment.

Overall, these contributions advance state-of-the-art humanoid technologies with the ability to 1). use hand contact for fall prevention in cluttered environment and 2). reason how to regain balance in both open and cluttered environments. By further enhancing legged machines' capability of self-balancing, methods discussed in this dissertation have the potential to realize a more effective and more reliable humanoid performance in real world.

Contents

List of Tables

List of Figures

List of Abbreviations and Symbols

Abbreviations

CoM	Center of Mass
CoP	Center of Pressure
CP	Capture Point
DOF	Degree of Freedom
FRI	Foot Rotation Indicator
IPM	Inverted Pendulum Model
LIPM	Linear Inverted Pendulum Model
PCA	Principal Component Analysis
PCoM	Projection of Center of Mass
SP	Support Polygon
VK	Viability Kernel
ZMP	Zero Moment Point

Symbols

m	mass.
$\boldsymbol{q}$	configuration
$\dot{\boldsymbol{q}}$	velocity
$\ddot{\boldsymbol{q}}$	acceleration
$\boldsymbol{x}$	state ($\boldsymbol{q}$, $\dot{\boldsymbol{q}}$)

$\mathcal{X}$	set of robot state
$\boldsymbol{u}$	control input
$\boldsymbol{\psi}$	stability margin
g	gravitational acceleration $(9.81 m/s^2)$
μ	Couloumb friction coefficient

Acknowledgements

I would like to first express my deepest appreciation to my advisor, Dr. Kris Hauser, for all of his support, guidance, motivation and encouragement throughout my Ph.D. study. He not only provides me with the freedom and flexibility to explore my own research projects but also helps me with his wise advice and invaluable feedback through numerous individual meetings. His passion and enthusiasm in research have always been inspirational and motivational for me. I'm extremely grateful to have the opportunity to join his lab.

I would like to extend my sincere thanks to my dissertation committee members, Dr. Brian Mann, Dr. Leila Bridgeman, Dr. Michael Zavlanos, and Dr. Thomas Witelski, for their time and valuable discussions. I very much appreciate Dr. Wilkins Aquino for being in my previous milestone exams.

I am also grateful to my lab friends and colleagues, including Yifan Zhu, Fan Wang, João Marcos Correia Marques, Gao Tang, Mengchao Zhang, William Edwards and Jaejun Park. I cannot leave Duke without mentioning Boyang Zhang, who always gives me encouragement with his sincere friendship. I also have great pleasure of making friend with Iordanis Chatzinikolaidis since beginning of our Ph.D. programs. I cannot believe that we have attended so many conferences together! I gratefully acknowledge the help that I received from Dr. Christine Payne and Michell Tampe.

Most importantly, I would like to send my deepest thanks to my parents, Jianjun Wang and Fenglan Li, for their unconditional and endless love.

1

Introduction

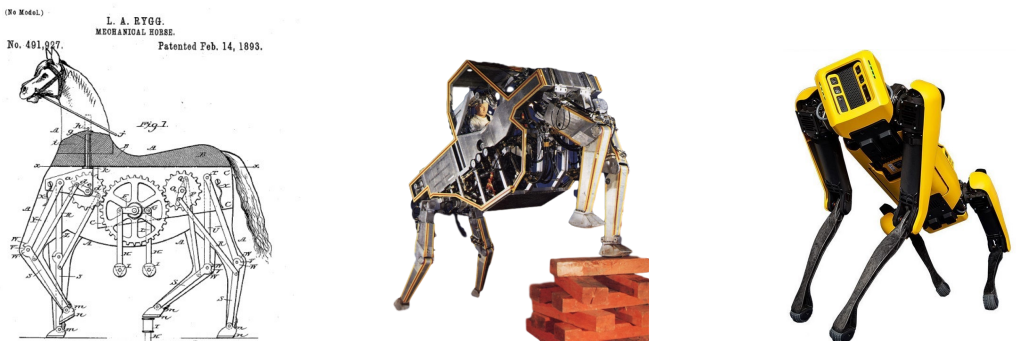
1.1 Motivation

What will the world look like in fifty years? This is a fairly unknown question because it is even hard to forsee what will happen in a year from now, not to say in fifty years. However, what can be predicted with confidence is that cutting-edge technologies, such as artificial intelligence (AI), will be highly developed at that time and smart machines will undertake critical roles where previously human's participation and engagement is a must. Even though it is inconsequential to imagine what specific tasks will be assigned to intelligent machines, developing robotic systems which can think, reason and act as humans is considerably important and in great need. Among a wide variety of robotic platforms, legged robot is of particular and increasing interest to robotics researchers resulting from its advantageous mobility on challenging terrains. Compared with wheeled robots, legged robots have the potential to traverse environments where only sparse handholds and footholds are accessible and their capabilities of dynamic locomotion, such as running and jumping, further enhance their adaptability to difficult terrains [MES85, LO88, SYZ97, PO06]. This sort of dexter-

FIGURE 1.1: Bighorn sheep demonstrate mobility skills in cliff-climbing, adopted from [FS17]

ous mobility has been commonly seen in legged animals in nature. For example, Figure 1.1 shows that bighorn sheep can climb around cliffs with narrow footholds space. Despite the fact that this animal-level mobility is far greater than the overall mobility of current legged robots, the realization of even a ratio of animal's agile capability can largely increase the application of legged robots to numerous tasks, for example, search-and-rescue in dangerous environments.

The idea to utilize legged locomotion is not new. Concept of legged mechanism can be traced back to late 19th century when Lewis. A. Rygg patented a quadrupedal walking machine called The Mechanical Horse (Figure 1.2a). This hand-drawn me-



(a) Mechanical Horse (b) Walking Truck (c) Spot

FIGURE 1.2: Evolution of legged machines from Lewis A. Rygg's Mechanical Horse [Ryg93] in 1893 to General Electric's Walking Truck in 1969 [Dep69] to Boston Dynamics's Spot in 2020 [Dyn20].

chanical figure directly makes use of rider's applied force on pedals as power to step forward and allows change of moving direction by steering the reins in front. Even though it is unclear whether this legged machine was physically made or not, this design provides inspirations for development of legged robots [Rai86]. Later legged platforms, such as General Electric's Walking Truck (Figure 1.2b) and Boston Dynamics's Spot (Figure 1.2c) show the efforts researchers have made so far towards a more **autonomous**, **agile**, **reliable** and **energy-efficient** legged robot. For the realization of such a robot, a number of technological challenges are expected to be overcome (i.e. environment representation, object modeling, whole-body motion planning and control, etc). While there are lots of open questions from each challenge, this dissertation focuses on motion planning for a specific type of legged robots, humanoid robot, with application to fall recovery.

Humanoid robot is an anthropomorphic robot whose appearance is designed to resemble human beings. With legged mobility from bipedal locomotion and manipulability from its hands, humanoid robots should be able to navigate in cluttered environment and handle sophisticated tools which require high accuracy to manipulate. Equipped with the most advanced robotics technologies, humanoids are expected to undertake a wide variety of tasks ranging from taking care of the elderly to doing house cleaning and disinfecting (especially important during pandemic). While these expectations are natural and these tasks are seemingly straightforward to human beings, current humanoid robots have not been translated into sustainable solutions for real-world use. The truth is that the development of modern humanoids mostly remains in the experimental stage and validation test is commonly conducted in a highly-controlled environment [KKM+11, EWO+14, EMW14, RSH+15, AOI17, KKS+19]. This limitation to laboratory environment results from humanoid robot's great risk of falling due to a relatively high position of center of mass (CoM) and a limited area of foot support. This idea can be clearly understood with an inverted

pendulum model (IPM) in upright posture. While IPM can remain static at this configuration, this state is unstable since any disturbance will break the equilibrium and bring pendulum fall completely down until it hits the support platform. This intrinsic instability of humanoids' bipedal walking makes them much harder to control, and falls can cause costly failures. As a result, protecting humanoid robot from falling failure is a topic of active research [FKK+02, FKH+06, Ste07, OTK08, WWS12, ARW12, LDHW16, TK16, KKK+17, SCBK17].

Fall recovery refers to regaining robot's balance when the robot is going to fall. Despite being important, humanoid fall recovery is a significantly difficult problem.

- Firstly, humanoid robot is a high-dimensional nonlinear system, which has already resulted in a large-size problem. This large number needs to be doubled or tripled, if kinodynamic constraints on robot's velocity, or velocity and acceleration, are to be considered. Figure 1.3 illustrates some representative humanoid platforms whose DOFs range from 28 (Atlas and HRP-2) to 37 (Xing Tian). The synthesize of dynamic motion for a robotic system with that large DOFs is challenging. For instance, grid-based planning algorithms, such as A* search algorithm and Dijkstra's algorithm, are not suitable due to the curse of

(a) Johnny 06 (b) Atlas (c) HRP-2 (d) Running Man (e) Xing Tian

FIGURE 1.3: Representative Robots in the Finals of DARPA Robotics Challenge 2015, adopted from [GA15b]

dimensionality [Bel03].

- Secondly, humanoid system is of a hybrid nature due to its intermittent contacts with environment. Whenever a new contact is established by the robot, there will be instantaneous changes in both robot's state and its system dynamics depending on the contact [HM94]. This combination of continuous whole-body motion and discrete contact results in a multi-modal planning problem where a planner must determine both the discrete contact mode sequence and the continuous transition trajectories under contact mode. Each contact mode constrained robot's feasible state to be on a sub-manifold denoted by $\mathcal{X}$. An efficient motion planner should be capable of computation of feasible state $\boldsymbol{x} \in \mathcal{X}$, computation of feasible path $P(s) : [0, 1] \rightarrow \boldsymbol{x} \in \mathcal{X}$, and computation of transition motions between two contact modes, whereas each of these three computations can be computationally expensive and challenging [HL10, Sti10, BSK11, JP13, KUSP16, EFRS20].

- Thirdly, recovery strategy has to be generated in "real-time". Based on the observation of robots falling down at the DARPA Robotics Challenge 2015 (Figure 1.4), a human size robot can fall completely down to the ground from

(a) ATLAS robot falling failure (b) ESCHER robot falling failure

FIGURE 1.4: A number of falling failures were observed in DARPA Robotics Challenge 2015, adopted from [GA15a]

a static upright configuration within 2 seconds after being slightly pushed [JSB+15, KHJ+17]. The time to fall will be even shorter if robot's height is lower or robot's initial velocity in falling direction is larger. Since robot's velocity in the vertical direction keeps on increasing during the falling process, linear momentum will be accumulated from gravitational force. Therefore, to prevent robot from severe collision damage and recover robot's balance while its centroidal velocity is small, a desired fall recovery strategy should be an online approach whose computation finishes much earlier than robot's falling to the ground.

A summary of the above three factors reveals the essence of humanoid fall recovery problem which is **a motion planning problem for a high-dimensional and hybrid nonlinear dynamical system with real-time computation requirement**.

Because directly working with a complicated dynamical system is difficult, **simple-model based** approaches are adopted to reduce the computational burden for online planning. Kajita's Linear Inverted Pendulum Model (LIPM) has been widely used to approximate humanoid's centroidal dynamics [KKK+01, KKK+03a, KKK+03b, PT06, EK09, ARW12, EOA13]. By modeling robot's body to be a point-mass and controlling its height to be on a horizontal plane, LIPM enables an analytic computation of a footstep location for instantaneous fall avoidance [PCDG06, MB11]. In addition, LIPM's decoupled dynamics in horizontal plane makes it convenient to verify whether a disturbed robot can return to a upright static state by taking N or less steps [KdBR+12]. If the size of the support foot is considered in LIPM, ankle torque can be used to balance robot against some less severe disturbances without taking a protective step [Ste07, SA10b]. In addition to this point-mass model with only linear momentum, angular momentum has also been introduced for body balancing. Pratt

proposed a Linear Inverted Pendulum Plus Flywheel Model to incorporate whole-body movement for disturbance rejection [PCDG06]. Similar ideas have been seen in Lee's Reaction Mass Pendulum [LG07] and Singh's Linearized Double Inverted Pendulum [SS20]. While these simplified models have the merit of online planning, their synthesized inertial shaping and protective stepping strategies are not general to cluttered environments due to the lack of consideration of environment geometry. Protective strategies with these simplified models presume a sufficiently large open environment for robot's translational and rotational movement and these methods can be infeasible in the presence of environment obstacles, such as tables and walls, in an indoor workspace. Here, an important **gap** in the state-of-the-art of model-based online planning is the lack of fall recovery method in cluttered environment. This dissertation **fills in this gap** with introduction of a hand contact strategy for balance recovery. Within this new planning algorithm, humanoid robot is able to actively reach its hand to make contact with these obstacles to prevent falling or to minimize the damage from falls. This extension to hand contact strategy completes the missing part in humanoid fall recovery literature where upper limb balancing strategy has not received much attention and this ability to adaptively make use of environmental object for fall prevention can improve humanoids' efficiency and reliability in human-centered environment.

After the introduction of hand contact strategy for cluttered environment, humanoid robot is capable of balancing itself from external disturbances with *fixed contact* approach such as inertial shaping strategy, and *contact modification* approach such as stepping and hand contact strategies. Despite the existence of various disturbance recovery strategies, it still remains unclear which strategy or combination of strategies should be adopted to stabilize a humanoid if an arbitrary pushed is imposed. Existing unified strategies use heuristic decision functions for mode switching [OTK08, KKK+17], and machine learning-based methods for fall planning [SL15],

[YZHL13]. However, these approaches can only handle a limited range of conditions, due to oversimplified assumptions of robot dynamics and contact, or a restrictive number of protective strategies. Thus, another critical **gap** is the lack of a unified fall recovery planner which generates full-body control trajectory that balances robot in arbitrary environment. This dissertation **fills in this gap** with a generalized humanoid whole-body motion planner that unifies *fixed contact* and *contact modification* strategies for fall recovery in both open and cluttered environment. By simultaneously generating the contact mode sequence and optimized whole-body trajectories to achieve a stabilizing multi-contact trajectory, this planning framework is capable of balancing a disturbed humanoid in arbitrary environment.

1.2 Prerequisite Definitions

This dissertation makes use of the following definitions:

Definition 1. *A **robot state** $\boldsymbol{x}$ is a collection of variables whose physical meaning can be robot's position, velocity, and acceleration. In this dissertation, robot state is a pair of robot's configuration variable $\boldsymbol{q}$ and its velocity $\dot{\boldsymbol{q}}$, so $\boldsymbol{x} = (\boldsymbol{q}, \dot{\boldsymbol{q}})$.*

Definition 2. *A **contact mode** indicates the contact active/inactive status, and is a vector $\boldsymbol{\sigma} \in \{0,1\}^{l\times 1}$ where l is the number of contact extremities allowable on the robot. $\boldsymbol{\sigma}_i = 0$ indicates that the i'th extremity has no contact (inactive), while $\boldsymbol{\sigma}_i = 1$ indicates contact (active).*

Definition 3. ***Failure**: a state in which a point on the robot makes undesired contact with the environment [PT06].*

Definition 4. ***Failure set** $\mathcal{F}$: the set of **Failure** states.*

Definition 5. ***Viable state**: a state $\boldsymbol{x}_0$ if there exists at least one control trajectory $\boldsymbol{u}(t)$ such that $\boldsymbol{x}(t)$ starting at $\boldsymbol{x}_0$ never enters $\mathcal{F}$.*

1.3 Literature Review and Background

1.3.1 Locomotion Stability Criteria

The locomotion stability evaluation problem — a prerequisite to fall recovery — asks to classify whether a robot is sufficiently disturbed from its nominal behavior so that it needs to initiate a rapid fall recovery action. This evaluation should be *responsive*: the earlier an impending failure can be detected, the more time the robot has available to plan mitigating motions. It should also be *accurate*: false positives will cause the robot to behave erratically and inefficiently. For this purpose, a number of stability criteria have been proposed in the literature of legged robots.

Viability Kernel

Based on Wieber's introduction of viability kernel (VK) [Wie02], a set of robot states from which a fall can be avoided with the balancing controller , the determination of locomotion stability is equivalently to a state classification problem of whether this disturbed robot state is included in VK.

The viability kernel, originally proposed for nonlinear control systems to characterize initial states from which there exists at least one feasible state trajectory to reach a target state in finite time [Aub90], has been conceptually accepted as a generalized stability criteria for humanoid robots. Embracing the definitions in Sec. 1.2, VK is defined as

$$\mathcal{V} := \{\boldsymbol{x}_0 | \exists \boldsymbol{u} \in \mathcal{U} : \boldsymbol{x}(t) \cap \mathcal{F} = \varnothing, \boldsymbol{x}(0) = \boldsymbol{x}_0, \forall t \geqslant 0\} \tag{1.1}$$

where robot's dynamical system can be written as $\dot{\boldsymbol{x}} = f(\boldsymbol{x}, \boldsymbol{u})$, and $\mathcal{U}$ is the set of valid control trajectories.

Given a control policy $\pi(\boldsymbol{x})$, the control trajectory $\boldsymbol{u}(t)$ is uniquely determined from the robot's state initial. The policy-specific VK is a slight modification of (1.1)

$$\mathcal{V}_\pi := \{\boldsymbol{x}_0 | \boldsymbol{u} \leftarrow \pi(\boldsymbol{x}) : \boldsymbol{x}(t) \cap \mathcal{F} = \varnothing, \boldsymbol{x}(0) = \boldsymbol{x}_0, \forall t \geqslant 0\}. \tag{1.2}$$

Then a stability margin ψ of state $\boldsymbol{x}$ under control policy $\pi(\boldsymbol{x})$ can be written as

$$\psi_\pi(\boldsymbol{x}) = \text{sgn} \cdot \min_{\boldsymbol{x}^* \in \mathcal{V}_\pi} ||\boldsymbol{x} - \boldsymbol{x}^*||, \ \text{sgn} = \begin{cases} 1, \boldsymbol{x} \in \mathcal{V}_\pi \\ -1, \boldsymbol{x} \notin \mathcal{V}_\pi \end{cases} \tag{1.3}$$

Different policies may or may not be able to stabilize a disturbed state due to different capabilities of disturbance rejection and thus produce different Viability Kernels. Figure. 1.5 presents a conceptual visualization of three Viability Kernels. Depsite the fact that VK presents a general fashion to evaluate robot's stability, explicit computation of a VK for legged robots suffers three considerable challenges:

1. High dimensionality, typically dozens of degrees of freedom (DOFs)
2. Nonlinear dynamics
3. Computation expense of simulating balance controller outcomes

Due to these difficulties, a number of simplified dynamics models, control policies, or heuristics are adopted from past researchers to approximate VK.

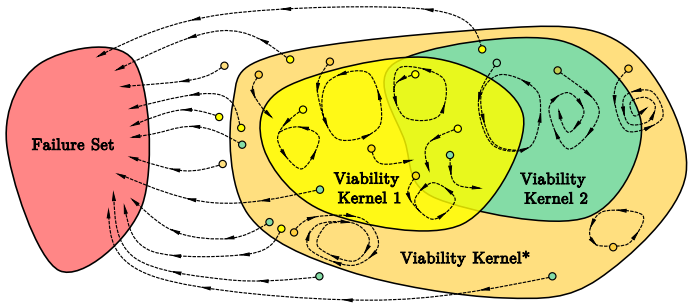


FIGURE 1.5: Illustration of Viability Kernels for three controllers $\pi^*(\cdot)$, $\pi_1(\cdot)$, $\pi_2(\cdot)$. State spaces of VK*, VK1, VK2 and **Failure Set** are denoted in orange, yellow, green and red, respectively. Each dot represents a robot state and the color inside the dot denotes the controller implemented to that state. Dashed curves show trajectories of how states evolve with the chosen controller.

Ground Projection of Center of Mass

The first stability margin of walking robot was proposed by McGhee and Frank in 1968 [MF68]. In their pioneering work, the projection of center of mass (PCoM) to the ground is used as an indicator for stability characterization and a legged robot is said to be statically stable if this PCoM lies within its support polygon (Figure 1.6). This idea essentially introduces two simplifications to stability margin in Equation 1.3:

1). $\boldsymbol{x} \rightarrow$ Ground Projection of Center of Mass

2). $\mathcal{V}_\pi \rightarrow$ Area of Support Polygon.

For walking machine programmed with a static or quasi-static gait, a deviation of PCoM from the interior of support polygon implies a tip-over momentum to be accumulated around certain edge, so an impending fall can be foreseen. This stability criteria was later extended to uneven terrain and a metric to measure the intensity of instability is defined to be minimum distance among all distances from PCoM to support polygon edges [MI79, MW89, ZS90]. Similar ideas have been seen with directly calculating robot's rotational momentum around each rotation axes [LS93], comparing whether robot's net force vector lies within its support force cone [PR96].

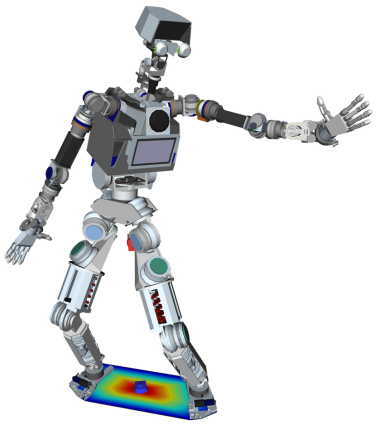


FIGURE 1.6: Visualization of support polygon with ARMAR-4 where blue dot denotes Ground Projection of Center of Mass, adopted from [VWK+15]

Although this evaluation of ground PCoM is effective to determine whether an uncompensated rotational momentum will bring the robot to tip-over around its support polygon, this criteria does not hold true for legged robots in dynamic locomotion [Gos99, Wie02]. For example, when a bipedal robot changes from double-support phase to single-support phase during dynamic walking, its ground PCoM may exit the support polygon to push the robot forward. Therefore, it is insufficient to characterize dynamic stability with ground PCoM.

Zero-Moment Point

During dynamic locomotion, a legged robot is said to be stable if its dynamic balance is maintained [VB04]. Here, dynamic balance refers to linear momentum balance (Equation 1.4a) and angular momentum balance (Equation 1.4b):

$$\sum_{i=1}^{N} \boldsymbol{f}_i + m\boldsymbol{g} = \dot{\boldsymbol{I}}_{\text{CoM}} \tag{1.4a}$$

$$\sum_{i=1}^{N} \boldsymbol{r}_{\text{CoM}\to i} \times \boldsymbol{f}_i = \dot{\boldsymbol{L}}_{\text{CoM}} \tag{1.4b}$$

where $\boldsymbol{f}_i$ denotes contact force at contact i and $\boldsymbol{r}_{\text{CoM}\to i}$ represents position vector pointing from CoM to contact i. $\dot{\boldsymbol{I}}_{\text{CoM}}$ and $\dot{\boldsymbol{L}}_{\text{CoM}}$ are rate of change of robot's centroidal linear momentum and angular momentum, respectively. The realization of desired momentum for robot locomotion is determined by the existence of cor-

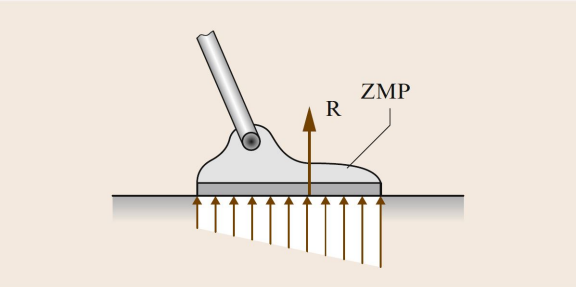


FIGURE 1.7: Illustrating ZMP/COP with a planar foot, adopted from [SK07].

responding contact forces between robot's end effectors and the environment. To ensure the dynamic balance of a bipedal walking robot in single-support phase on flat terrain, its stance foot should contact the ground with whole area [VB04]. This indicates that the support foot link is in static equilibrium. If ground's friction force is assumed to be enough, zero-moment point, a dynamic version of "Ground PCoM", is proposed to determine dynamic stability.

Zero-moment point (ZMP), was proposed by Vukobratovic in early 1970s [VS72a, VS73]. It is generally understood to be a point on the flat ground where the ground reaction does not have horizontal moments to balance the robot [TIYK85, AF97, VB04]. Since the effect of ground reaction can be equivalently replaced by a vertical support force and a vertical moment at zero-moment point, ZMP is also stated as the center of pressure (COP) [ORI76], which can be directly measured with force/torque sensors mounted on the bottom of foot, as shown in Fig. 1.7. This idea simplifies stability margin in Equation 1.3 with

1). $\boldsymbol{x} \rightarrow$ ZMP

2). $\mathcal{V}_\pi \rightarrow$ Area of Support Polygon.

Even though the original statement of ZMP is not directly related to the concept of stability since it only states that a desired dynamical locomotion can be balanced if ZMP remains within the support polygon, ZMP turns out to be appealing in motion planning of bipedal locomotion because the constraint of having ZMP within the support polygon guarantees a face-to-face contact between robot's stance leg and flat ground [KKK$^+$01, KKN$^+$02, KKK$^+$03a, KKK$^+$03b, NNY$^+$03, Wie06, EK09, HDW$^+$10]. Being widely successful in the motion generation of bipeds, ZMP and its variants have then been used to define stability margins. The most commonly used stability margin with ZMP is defined to be the minimum distance from robot's ZMP to edges of its support polygon [HG77, TIYK85, SLC$^+$90, LTK92, LTK93, HHHT98]. Goswami introduced the Foot-Rotation Indicator point to extend ZMP margin out-

side the support polygon [Gos99]. Research on stability with ZMP in the new century has been shifted to the analysis of contact stability to achieve dynamic balance. Harada and his colleagues first applied ZMP to a multi-contact situation [HKKH03]. Three years later, Hirukawa and his colleagues proposed a universal stability criterion to reformulate the check of dynamic balance problem into a check of geometrical inclusion of dynamics wrench in the polyhedral convex cone of frictional contact wrench [HHH$^+$06]. Up to this contribution, the theory to characterize instantaneous linear/angular momentum balance has been well-established. A recent related work has been in analytically specifying the convex cone of frictional contact wrench for linear inverted pendulum model [CPN15, CPN17].

Energy-based Criteria

Energy-based failure metric aims to measure the instability magnitude with a comparison between robot's present kinetic energy with the minimum kinetic energy that will lead to a tip-over motion around any edge of SP.

Messuri first proposed an energy stability margin (ESM) with the minimum potential energy to be accumulated to cause a tip-over motion around edges of support polygon [MES85]. With the consideration of ground compliance and deformation at foot contact point, Nagy extended Messuri's ESM to soft terrains [NDW94]. This ESM has further been normalized by robot's weight, unit from J to m, to increase the intuitive interpretability of this margin [HTY01]. The linear inverted pendulum model (LIPM) proposed by Kajita initially for bipedal walking has also shed light upon the design of stabiliy margin for legged robots [KYK92]. This model consists of a massless telescopic leg and a point-mass whose vertical position is controlled to be on a horizontal plane, as shown in Figure 1.8. With these assumptions, pendulum dynamics in vertical direction and horizontal plane are completely decoupled and the acceleration in horizontal direction is proportional to its horizontal position

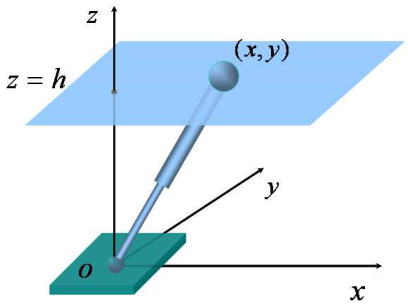


FIGURE 1.8: Illustrating linear inverted pendulum model, adopted from [JV08]

displacement:

$$\ddot{x} = \frac{g}{h}x \tag{1.5a}$$

$$\ddot{y} = \frac{g}{h}y \tag{1.5b}$$

where x and y denote LIPM's Cartesian coordinates in horizontal plane and h is the height of CoM. With these interesting equations of motion, orbital energy is introduced to evaluate the instantaneous stability, capture point (CP), or N-step stability [PCDG06, KdBR+12]. Defined to be the sum of pendulum's kinetic energy and "fictitious" potential energy, orbital energy is written as

$$E_x = \frac{1}{2}\dot{x}^2 - \frac{1}{2}\frac{g}{h}x \tag{1.6a}$$

$$E_y = \frac{1}{2}\dot{y}^2 - \frac{1}{2}\frac{g}{h}y \tag{1.6b}$$

where kinetic energy is proportional to velocity squared ($\dot{x}^2$ and $\dot{y}^2$) and potential energy is expressed to be one half of a product between LIPM's fictitious stiffness ($-\frac{g}{h}$) with its horizontal displacement (x and y). E_x and E_y's value can indicate LIPM's instantaneous stability status based on their signs where positive value shows robot's centroidal velocity cannot decrease to zero when CoM reaches edge of SP, thus leading to a tip-over failure. This idea simplifies stability margin in Equation 1.3 with

1). $\boldsymbol{x} \rightarrow (x, \dot{x})$ and $(y, \dot{y})$

2). $\mathcal{V}_\pi \rightarrow \{(x, \dot{x}), (y, \dot{y}) | E_x \leqslant 0, E_y \leqslant 0\}$

Other heuristics embrace a similar idea and calculate the total mechanical energy using some variants of inverted pendulum models [LZC+15, WH18a].

Data-driven-based Criteria

Viability kernel $\mathcal{V}_\pi$ depends on the balance control policy $\pi(\cdot)$ implemented on the robot. Balance controllers are capable of returning the robot back to a nominal state or a nominal trajectory after a disturbance. Even rigidly controlled robots may not actually fall after being tipped onto an edge, because they may tip back and settle into equilibrium (wobbling). Although the direct computation of $\mathcal{V}_\pi$ is difficult for a specific controller $\pi(\cdot)$, $\mathcal{V}_\pi$ can be numerically approximated with sufficient robot's locomotion data under that controller $\pi(\cdot)$. Data-driven approach for fall detection aims to predict robot's failure based on the analysis and interpretation of robot's performance under disturbances. This model-free approach always involves a two-stage process:

1. Data collection of safe/failure states through experimentation.

   This step aims to gather a database that captures the essential pattern for robot failure classification. Representative features at each record time will be saved for pattern regression later.

2. Failure pattern regression through data processing.

   With sufficient data collected, a data mining process will be conducted to build a failure detection model by deciphering the implicit characteristics of failure state from those measurements.

The described methodology has been used by a number of researchers to develop failure detection models [RB06, OK07, OTK08, KW09, HG09a, KG11]. A direct comparison of these work is shown in Table. 1.1.

Table 1.1: Comparison of data-drive-based methods on fall detection

Author	Year	Feature	Method	Time (s)	Accuracy(%)
Renner Behnke	2006	Euler angles & Angular speeds	Fourier Decomposition	0.1	> 90
Ogata	2007	Robot state	Discriminant Analysis	< 0.2	NA
Karssen Wisee	2008	Robot state	Multi-way PCA	NA	> 90
Hohn Gerth	2009	Contact force & Orientations	GMM HMM	0.05 $0.15 \sim 0.2$	> 64 > 86
Shivaram Goswami	2011	Customized attributes	Supervised Learning	0.7	> 90

GMM: Gaussian-Mixture-Models, HMM: Hidden-Markov-Models

1.3.2 Strategies for Balance Recovery

Balance recovery is achieved by altering how robot reacts to the external disturbance. Small disturbances can be easily resisted by adjusting body posture, while larger disturbances may require one or more protective steps, and possibly even hand contact. For the realization of such recovery behaviors, a variety of strategies have been proposed by humanoids researchers. These strategies can be classified into three classes depending on their *principles*: **Control-based**, **Optimization-based**, and **Machine-learning-based** approaches.

Control-based approach

Control-based approach focuses on solving one problem: How to design control policies to balance robot from unexpected pushes based on simplified humanoid models? Among all proposed models, LIPM is highly preferred for balancing controller design due to its simplified decoupled system dynamics. This advantage enables a direct application of classical control theory to the balancing of this model and its variants. Methods such as system linearization around equilibrium point, feedback lineariza-

tion and local PID controller have been used to recover the robot back into upright posture [GHW81, HC76, MZS09]. LIPM has also been used to derive an analytic expression of a protective foothold, capture point (CP), to arrest robot from falling [PCDG06, MB11]. Capture point has further been extended into 3D environment as N-step capture region in which the robot can make at most N steps to capture itself into a upright static state, as shown in Fig. 1.9.

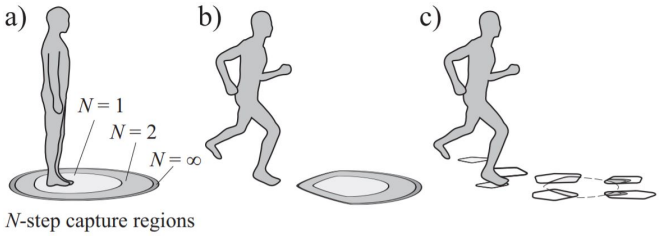


FIGURE 1.9: Illustrating N-step capture regions, adopted from [KdBR$^+$12]

In addition to these methods, balance strategies have been developed with bang-bang control [Ste07], virtual model control [SA10a], and model predictive control to generate human-like protective behaviors such as ankle strategy, hip strategy and stepping [SA10b, ARW12]. With the high-level CoM trajectory generated with LIPM, researchers have proposed algorithms to generate whole-body control policy to follow this nominal trajectory [GK04, YGS09, Aa10, CTD$^+$10, OGL13, RMS14].

Control-based approaches use simplified models to plan protective motions, such as stepping and inertial shaping. Since this sort of method does not consider robot's 3-D geometry model or environmental features, configuration for stepping or inertial shaping can be infeasible for the robot to reach due to the risk of collision with its self or environment object(s). In addition, the fact that arm links are always neglected in this method prohibits the possibility of robot stabilization with hand contact.

Optimization-based approach

Optimization-based approach generates an optimal policy which can drive robot from its disturbed initial state through a feasible trajectory to a goal state while minimizing a performance index. One significant advantage of optimization-based approach over control-based approach is the capability of generating complicated mitigation motion with contact planning since it can consider robot models which are more sophisticated and much closer to real humanoid hardwares. Mitigation with knee contact and hand contact has been proposed by Fujiwara and his colleagues using Pontryagin's minimum principle [FKH$^+$06, FKH$^+$07]. A similar knee contact damage minimization behavior is generated with a trajectory optimization using Chebyshev polynomials in [WWS12]. Mansour proposed a direct collocation approach with quadratic splines to enable the utilization of 3D spatial contacts for push recovery [MML11]. A similar trajectory optimization approach with path planning for robot's CoM is recently proposed to detect the robot postural stability [DTM18]. Ha and Liu proposed a multiple contact planning algorithm to reduce collision impact using dynamic programming [HL15]. Researchers have also used quadratic programming to adaptively optimize controller gains post-impact stabilization [SCBK17].

Although optimization approach can take into computation robot's whole body model, complicated constraints, such as system dynamics and contact, and generate a number of protective behaviors with different contact strategies, this type of approach has long computational time in general. Furthermore, it can only converge to a local optima provided promising initial guess.

Machine-learning-based approach

Machine learning (ML) methods enable artificial devices to have the capability of self-training and then automatically developing task-oriented behaviors without explicitly being programmed. The fact that Google DeepMind successfully generates

several flexible human-like locomotion in simulated environment with Reinforcement Learning (RL) shows RL's powerful competency in hybrid motion planning. The most relevant works in dynamic motion planning with deep RL have been seen in [PBvdP15, PBYVDP17a], where a 3D toy model of bipedal robot is able to navigate itself through cluttered environment on uneven terrains. However, just like other RL methods to address legged locomotion problems, robot trained policies generated with deep RL cannot generalize. They can only work for situations similar to the ones used in training process. To be an effective solution to a general hybrid motion planning problem, deep RL should demonstrate its capability of handling fall recovery in any arbitrary environment. The diversity in environment reflects various constraints to the motion planning problem, thus causing the number of training data to be too large to be gathered. In addition to RL, other supervised learning approaches have also been proposed for fast motion planning. Strategies such as re-use of pre-computed results [PCCL12], centralized node sampling with previous experience [AT15], learning heuristic function for node expansion [BCS17], and learning from optimal trajectories [TSH18] all contribute to accelerate motion planning process. However, if test will be conducted in scenarios which are different from the ones used for training, the learned control policy is not able to produce useful solutions. In addition, the data collection task for supervised learning in hybrid systems turns out to be extremely expensive since it needs sufficient number of successful planned trajectories to learn a unified policy but generation of those trajectories takes considerably amount of time even with state-of-the-art methods. Machine learning has also been used by researchers as an efficient tool to generate fall mitigation strategies. Yun and Goswami employs a reinforcement learning approach to produce a tripod configuration for the robot to reduce collision impact [YG14a]. In Yi et al, a machine learning approach is used to select between predefined push recovery controllers for a small humanoid robot, including use of the ankle, hip, or

stepping [YZHL13]. A more comprehensive learning approach was taken in Kumar et. al., in which reinforcement learning is used to find a full body fall-recovery policy that can use protective stepping or hands, and this outperforms direct optimization by orders of magnitude [KHL17].

Machine-learning-based approach outperforms optimization-based approach in terms of computational speed. However, protective motions generated with machine learning approach cannot adapt the learned policy well to a different situation if this situation is not included within the training data.

1.3.3 Optimization-based Motion Planning

General Description

Motion planning is a problem of finding a feasible path that moves a robot from initial configuration to goal configuration without causing any collision. This problem has been extensively studied over the past few decades and three major approaches have been seen in motion planning literature:

- Grid-based method

  This type of approach decomposes search space into a graph and finds a path over this graph according to some objective (i.e. shortest distance). Researcher have introduced Voronoi diagram [TS89, MAN04, GMAM06, GMB06, BG07, NSTJ19], visibility graphs [LPW79, JLK95, KOK+03, HS04], and grids [Yap02, CFS06, FS06] for graph construction and Dijkstra's algorithm and A* algorihtm for path query [Dij59, HNR68]. Grid-based methods are generally fast for problems with low-dimensionality, but their extension to high-dimensional problems can be computationally intractable due to the curse of dimensionality. Their incapability to handle constraints, such as dynamics and contact, makes them less appealing for legged motion planning.

- Sampling-based method

  This motion planning strategy searches collision-free path with randomly sampled waypoints from a probability distribution. It begins with a construction of a connectivity graph by connecting neighbouring configurations whose connection path is collision-free. It then finds a path from initial configuration to goal configuration within this graph. State-of-the-art techniques, such as Probabilistic Roadmaps (PRM) and Rapidly-exploring random trees (RRT), have shown to be effective in high-dimensional robotic systems [KSLO96, HKL$^+$98, ABD$^+$98, BK00, KL00, LJJK01, KF11] and methods such as Constrained Bi-directional Rapidly-Exploring Random Tree [BSFK09] further extend efficiently sampling on constrained manifolds. It should be noted that sampling-based algorithms have the property of probabilistic completeness, which means that the algorithm is guaranteed to find a solution if there exists one. Sampling-based methods address kinematic constraints easily, but their ability to handle dynamics constraint is limited [KF10]. There are are some techniques which plan quasi-static motions for legged systems whose dimensions are considerably high [KL00, HL10, HNTH11, BK12]. However, those techniques are not able to take care of hybrid system dynamics and contact velocity/acceleration constraints.

- Optimization-based method

  Optimization-based motion planning algorithms generate feasible robot trajectories by solving an optimization problem with an objective function and a set of constraints. Robot trajectories (i.e. configuration, velocity, acceleration, contact force, etc) are parameterized into a sequence of discrete knot points and numerical optimization algorithms, such as Sequential Quadratic Programming (SQP) or Interior-point methods, are used to solve for optimal solutions in

the parameter space [NW06, RZBS09, SDH$^+$14]. Provided a promising initial seed, these techniques can convergence quickly to a local optima and complicated constraints such as system dynamics and frictional contact, can be easily addressed with this type of method. State-of-the-art approaches are able to plan motion trajectories $\boldsymbol{x}(t)$ and $\boldsymbol{u}(t)$ given contact sequence of hybrid system [PKT16], and to simultaneous plan motion trajectories and mode sequence for walking machines [MTP12, PCT14]. Optimization-based method has advantageous ability in handling complicated constraints for high-dimensional dynamical system and this advantage over grid-based method and sampling-based method makes it attractive to the study of humanoid fall recovery problem. Admittedly, optimization-based approach relies heavily on "good" initial seeds to converge to local optima. If initial seeds are not provided properly, the optimization solver will not be able to make any progress. For large size non-convex optimization problems, the initialization of a promising seed is considerably difficult. To compensate for this limitation and ensure the implementability of the motion planning algorithms in this dissertation to different robotic platforms, seed initialization method for each proposed planner is clearly documented and experimentally tested.

Problem Formulation

A general optimization-based motion planning problem can be formulated into the following form:

$$\min_{\mathbf{x}} \quad f(\mathbf{x}) \tag{1.7a}$$

$$\text{s.t} \quad g_i(\mathbf{x}) \geqslant 0, \quad i = 1, 2, ..., n_{\text{ieq}} \tag{1.7b}$$

$$h_j(\mathbf{x}) = 0, \quad j = 1, 2, ..., n_{\text{eq}} \tag{1.7c}$$

where f, g and h denote objective function, inequality constraint whose number is n_{ieq} and equality constraint whose number is n_{eq}, respectively.

$\mathbf{x}$ is the variable to be optimized and it may include the following terms:

- t_f: final time.
- $\boldsymbol{q}(t)$, $\dot{\boldsymbol{q}}(t)$, $\ddot{\boldsymbol{q}}(t)$: trajectories of robot configuration, velocity, acceleration.
- $\boldsymbol{u}(t)$: robot control trajectory
- $\boldsymbol{f}(t)$: contact force trajectory
- $\boldsymbol{\sigma}_0, ..., \boldsymbol{\sigma}_N$: a sequence of contact mode.

Depending on the problem, some terms may not appear in the formulation, but for the humanoid fall recovery studied in this dissertation, all of these terms will be considered.

Objective function f penalizes the deviation of robot motion from a nominal reference and it can take a variety of forms. A common expression of an objective function is a weighted sum of several quadratic cost functions where each function represents the extent that a "soft" constraint can be violated [dLMH10, HHL14, DMN+14, HRG+14, FWXA15, HDO16, DJF+17]. As for humanoid fall recovery, a

desired objective should be chosen to stabilize humanoid from falling. This dissertation considers two objective functions: one focuses on reducing collision impulse and contact slippage, and the other focuses on minimizing robot's kinetic energy.

Constraint function g and h impose restrictions that robot's motion has to satisfy to remain feasible. Some general constraints in robotics field are presented as follows:

- *Joint limits, velocity limits, acceleration limits, and torque limits*:

$$\begin{aligned} \boldsymbol{q}_{min} \leqslant & \boldsymbol{q} \leqslant \boldsymbol{q}_{max} \\ \dot{\boldsymbol{q}}_{min} \leqslant & \dot{\boldsymbol{q}} \leqslant \dot{\boldsymbol{q}}_{max} \\ \ddot{\boldsymbol{q}}_{min} \leqslant & \ddot{\boldsymbol{q}} \leqslant \ddot{\boldsymbol{q}}_{max} \\ \boldsymbol{u}_{min} \leqslant & \boldsymbol{u} \leqslant \boldsymbol{u}_{max} \end{aligned} \tag{1.8}$$

- *Dynamics constraint*: Robots can be generally viewed as a chain structure of rigid links connected by joints. The equation of motion (EoM) of the robot is the system dynamics equation of these rigid links subjected to constraints from joints and contacts. A general form of EoM for a robot with n DOFs, m control inputs and l extremities can be written as follows

$$D(\boldsymbol{q})\ddot{\boldsymbol{q}} + C(\boldsymbol{q}, \dot{\boldsymbol{q}}) + G(\boldsymbol{q}) = J(\boldsymbol{q})^T \boldsymbol{\lambda} + B\boldsymbol{u} \tag{1.9}$$

  where $D(\boldsymbol{q}) \in \mathbb{R}^{n\times n}$ is the inertia matrix. $C(\boldsymbol{q}, \dot{\boldsymbol{q}}) \in \mathbb{R}^{n\times 1}$ is the centrifugal and coriolis matrix. $G(\boldsymbol{q}) \in \mathbb{R}^{n\times 1}$ is the generalized gravitional matrix. $\boldsymbol{u} \in \mathbb{R}^{m\times 1}$ is the joint torque vector and $B \in \mathbb{R}^{n\times m}$ is the input matrix. $\boldsymbol{\lambda} = [\boldsymbol{\lambda}_1^T, ..., \boldsymbol{\lambda}_l^T]^T \in \mathbb{R}^{3l\times 1}$ be the contact force vector. $J(\boldsymbol{q}) \in \mathbb{R}^{3l\times n}$ is the task-space Jacobian matrix of robot's end effector positions [Kha87, NCM+08, BSK11].

  This form of EoM can be analytically derived with Newton-Euler equation or Euler-Lagrangian method [Gre03]. Meanwhile, efficient algorithms also exist to compute desired quantities from equation (1.9) numerically. Inverse dynamics,

the calculation of generalized external force (right hand side of Equation 1.9) that must be applied to produce a desired acceleration, can be solved with the recursive Newton-Euler algorithm (RNEA) [LWP80]. Forward dynamics, the calculation of generalized acceleration $\ddot{\boldsymbol{q}}$ from a given generalized external force, can be achieved with Inertia Matrix Methods and Propogation Methods [Fea07].

- *Coulumb friction constraints*: If the contact force between the robot and the environment follows the Coulomb friction model, this contact stability constraint is written as

$$\mu^2 \lambda_{i\boldsymbol{n}}^2 \geqslant \lambda_{i\boldsymbol{t}}^2, i = 1, ..., l \tag{1.10}$$

  where $\lambda_{i\boldsymbol{n}}$ and $\lambda_{i\boldsymbol{t}}$ respectively denote the contact force $\boldsymbol{\lambda}_i$ in the normal and tangential direction of contact i.

  This quadratic constraint sets the feasible contact force to be within a cone volume and is usually approximated with a set of linear constraints to fit into the framework of linear complementarity problem (LCP) or quadratic programming (QP).

- *Complementarity constraints*: Let $\boldsymbol{\phi}(\boldsymbol{q}) = [\boldsymbol{\phi}(T_1(\boldsymbol{q})), ..., \boldsymbol{\phi}(T_l(\boldsymbol{q}))]^T$ and $\boldsymbol{\lambda_n} = [\lambda_{1\boldsymbol{n}}, .., \lambda_{l\boldsymbol{n}}]^T$ where $\boldsymbol{\phi}(\cdot)$ denotes a signed distance measured from position $\cdot$ to the environment, $T_i(\boldsymbol{q})$ is end effector i's position, and $\lambda_{i\boldsymbol{n}}$ is defined in Eq. (1.10). Then,

$$\boldsymbol{\phi}(\boldsymbol{q}) \geqslant 0 \tag{1.11a}$$

$$\boldsymbol{\lambda_n} \geqslant 0 \tag{1.11b}$$

$$\boldsymbol{\phi}(\boldsymbol{q})^T \boldsymbol{\lambda_n} = 0 \tag{1.11c}$$

  This constraint together with Eq. (1.10) ensures that contact force only exists at extremity whose signed distance is zero.

- *Contact holonomic constraints*: For a robot extremity to remain fixed, its velocity and acceleration have to vanish. Let $Diag(\cdot)$ generate a diagonal matrix with each element triplicated from vector $\cdot$ on its diagonal.

$$Diag(\boldsymbol{\sigma})J(\boldsymbol{q})\dot{\boldsymbol{q}} = \mathbf{0} \tag{1.12a}$$

$$Diag(\boldsymbol{\sigma})(\dot{J}(\boldsymbol{q})\dot{\boldsymbol{q}} + J(\boldsymbol{q})\ddot{\boldsymbol{q}}) = \mathbf{0} \tag{1.12b}$$

  where $J(\boldsymbol{q}) = \frac{\partial T(\boldsymbol{q})}{\partial \boldsymbol{q}}$ is task-space Jacobian matrix with $T(\boldsymbol{q}) = [T_1(\boldsymbol{q})^T, ..., T_l(\boldsymbol{q})^T]^T$ denoting the position of each extremity on the robot.

- *Impact mapping constraint*: Impact happens when a contact is added, and assumptions held in this dissertation are that impact is inelastic and instantaneously changes the pre-impact velocities $\dot{\boldsymbol{q}}^-$ to post-impact velocities $\dot{\boldsymbol{q}}^+$ through an infinitesimal time duration.

  $\dot{\boldsymbol{q}}^+$ must satisfy the goal contact mode holonomic constraint, so the impulse $\bar{\boldsymbol{\lambda}}$ and post-impact state can be calculated by solving the linear equation [HM94].

$$\begin{bmatrix} D(\boldsymbol{q}) & -J_{\boldsymbol{\sigma}}(\boldsymbol{q})^T \\ J_{\boldsymbol{\sigma}}(\boldsymbol{q}) & \mathbf{0} \end{bmatrix} \begin{bmatrix} \dot{\boldsymbol{q}}^+ \\ \bar{\boldsymbol{\lambda}} \end{bmatrix} = \begin{bmatrix} D(\boldsymbol{q})\dot{\boldsymbol{q}}^- \\ \mathbf{0}. \end{bmatrix} \tag{1.13}$$

  where $J_{\boldsymbol{\sigma}}(\boldsymbol{q})$ denotes active task-space Jacobian matrix which is a concatenation of Jacobian matrix of in-contact (active) extremities. If $J_{\boldsymbol{\sigma}}(\boldsymbol{q})$ is a full-row rank matrix, then an analytic solution is available for collision impulse and post-impact velocity:

$$\bar{\boldsymbol{\lambda}} = -(J_{\boldsymbol{\sigma}}(\boldsymbol{q})D(\boldsymbol{q})J_{\boldsymbol{\sigma}}(\boldsymbol{q})^T)^{-1}J_{\boldsymbol{\sigma}}(\boldsymbol{q})\dot{\boldsymbol{q}}^- \tag{1.14a}$$

$$\dot{\boldsymbol{q}}^+ = -D(\boldsymbol{q})^{-1}J_{\boldsymbol{\sigma}}(\boldsymbol{q})(J_{\boldsymbol{\sigma}}(\boldsymbol{q})D(\boldsymbol{q})J_{\boldsymbol{\sigma}}(\boldsymbol{q})^T)^{-1}J_{\boldsymbol{\sigma}}(\boldsymbol{q})\dot{\boldsymbol{q}}^- + \dot{\boldsymbol{q}}^- \tag{1.14b}$$

The overall optimization-based multi-modal problem that is solved is

$$\begin{aligned} \underset{\mathbf{x}=\{t_f,\boldsymbol{x}_1(t),...,\boldsymbol{x}_N(t),\boldsymbol{u}_1(t),...,\boldsymbol{u}_N(t),\boldsymbol{\lambda}_1(t),...,\boldsymbol{\lambda}_N(t)\}}{\text{minimize}} \quad & f(\mathbf{x}) \\ \text{subject to} \quad & (1.9) - (1.13) \end{aligned} \tag{1.15}$$

where N is the number of contact modes, $\boldsymbol{x}_i(t)$, $\boldsymbol{u}_i(t)$ and $\boldsymbol{\lambda}_i(t)$ are trajectories of robot state, control and contact force under contact mode $\boldsymbol{\sigma}_i$. This continuous trajectory optimization problem is generally transcribed into a constrained parameter optimization problem which can be solved with nonlinear programming techniques such as direct shooting method and direct collocation method [Bet98, BDLS00, PCT14, Kel17].

1.4 Summary of Contributions

This dissertation consists of the following contributions:

- Development of a "real-time" humanoid fall recovery motion planning algorithm which enables robot to make use of hand contact for fall protection in cluttered environment. This technique outperforms state-of-the-art optimization-based fall recovery method by orders of magnitude (Chapter 2).

- Implementation of the hand contact motion planner into a self-contain fall recovery system including fall detection, fall stabilization and push-up recovery (Sec. 2.5).

- Development of a generalized humanoid fall recovery planning that unifies inertial shaping, protective stepping, and hand contact strategies in arbitrary environment and proposal of an effective algorithm to generate promising initial seeds for trajectory optimization (Chapter 3).

- Development and implementation of an efficient online calibration algorithm for autonomous vehicle's longitudinal dynamical system (Chapter 4). This work is conducted during my summer internship at DeepRoute.ai in 2020. A patent based on this work has been filed.

2

Fall Recovery with Hand Contact in Cluttered Environment

This chapter presents an efficient motion planner with which a humanoid robot can prevent falling using hand contact with walls and other surfaces in the cluttered environment. Instead of ignoring or avoiding interaction with environmental obstacles, this planner considers these obstacles as hand rails and determines a contact point on them for fall stabilization. Formulated as an optimal control problem, the proposed algorithm achieves real-time planning with three techniques: 1). humanoid dynamics simplification with three-link model, 2). problem size reduction with direct single shooting method, 3). pre-computation of computationally expensive terms in evaluating cost function. This planner is firstly tested in a simulated environment and then integrated into a self-contained fall recovery system which is capable of fall detection, fall stabilization and push-up recovery. System integration is performed on the Darwin-Mini robot and validation is conducted in several environments and falling scenarios [1].

[1]This chapter is reproduced from Shihao Wang and Kris Hauser, "Real-time Stabilization of a Falling Humanoid Robot using Hand Contact: An Optimal Control Approach," in 2017 International Conference on Humanoid Robots, Birmingham, UK, 2017, and Shihao Wang and Kris Hauser,

2.1 Introduction

Humanoid robots can navigate in challenging terrains, step over environmental obstacles and reduce energy consumption by utilizing natural dynamics [PP98]. However, they suffer a high risk of falling due to the intrinsic instability of their bipedal locomotion [McG88]. Falls can cause costly failures to the robot and expensive human time and effort may have to be spent on the repairing process before the robot can be used again. To reduce the expensive human labor and equip humanoid robot with the autonomy to protect itself when it is going to fall, many strategies have been proposed to recover the robot from falling failure [PCDG06, FKH$^+$06, YZHL13, YG14a].

Among all existing fall recovery strategies, protective stepping is the most widely used one. This method plans future capture footsteps for the robot to stop or slow its rate of fall. However, protective footsteps may not be feasibly taken if the robot were to be used in a cluttered environment with obstacles such as walls and tables. These environmental obstacles can occupy positions of the planned footsteps and preclude the robot from being recovered from fall. However, the existence of environment obstacles provides the robot with a novel strategy for fall recovery. Many human environments are designed with rails and handles that can be grasped with the hands to maintain stability. In clutter or when rails are available, the robots hands could be used to prevent falling or to minimize the damage from falls (Fig. 2.1). However, it remains a major challenge for robots to fully exploit full-body contact with environmental geometry in real-time. To address this issue, an optimal control approach is presented for exploiting hand contact with walls and other environmental obstacles to stabilize a falling robot.

"Realization of a Real-Time Optimal Control Strategy to Stabilize a Falling Humanoid Robot with Hand Contact, " in 2018 International Conference on Robotics and Automation (ICRA), Brisbane, Australia, 2018. This research is supported by NSF grant NRI #1527826.

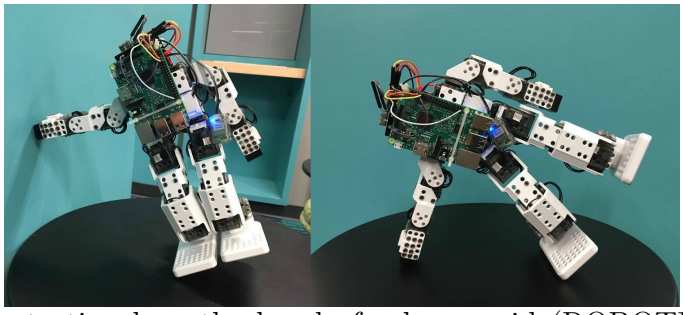


FIGURE 2.1: Illustrating how the hand of a humanoid (ROBOTIS Mini [INC20]) can be used to stabilize itself on a wall and to reduce falling damage on flat ground.

The following assumptions are used in this approach:

- A fall detector predicts the robot has begun to fall towards a wall or other environmental features.
- The robot's starting configuration and velocities are known.
- All dynamics are computed in the *falling plane*, which is the plane containing the center of mass velocity and the gravity vector.
- The shape of the nearby environment in the falling plane is known.
- Centers of masses and inertia matrices of the robot's links are known.
- The hand makes contact on the environment at a single point of contact, and this point of contact must stick (rather than slide or bounce) for the robot to avoid further falling. This assumption will be enhanced to hold from the proposed optimal controller.

2.2 Problem Formulation

A general optimization-based motion planning framework is discussed in Sec. 1.3.3 whose instantiation in this problem is as follows:

- Variable to be optimized:

  To reduce the problem size, a direct single shooting method is adopted to solve this nonlinear programming problem. Thus, robot control trajectory $\boldsymbol{u}(t)$ is the only term to be optimized in this formulation. Since hand contact is specified in advance, this formulation simplifies the exploration of contact mode sequence with two determined contact modes (foot contact, foot contact + hand contact). With $\boldsymbol{u}(t)$, other terms, such as final time t_f and robot state trajectories $\boldsymbol{q}(t), \dot{\boldsymbol{q}}(t)$, can all be computed accordingly.

- Objective function:

  Objective function guides the search of an "optimal" contact point where robot can be stabilized after making hand contact at that position. Due to the fact that robot has some residual velocity after making hand contact, the whole fall stabilization motion is divided into two phases:

  1. **pre-impact**: where the robot is pivoting about an edge of its foot and its hand is brought to touch the environment.
  2. **post-impact**: where the hand is in contact with the environment and any residual velocity is dampened.

  Each phase has a *catastrophic event* that can potentially prevent robot from being stabilized:

  - In **pre-impact** phase, robot makes hand contact with the environment, so a collision impact will be introduced to the robot. This presents the first *catastrophic event*: the impact from collision damages the robot.
  - In **post-impact** phase, the robot should be stabilized to a static configuration. To avoid the robot slipping to cause a secondary damage,

robot's active contacts should remain fixed during **post-impact** phase. As a result, the second *catastrophic event* describes how securely robot's contact can be kept fixed until robot reaches final state. According to Coulomb friction assumption, a contact will not move if its contact force resides within the friction cone defined by environment's friction coefficient. Here, a *necessary sticking friction coefficient* can be defined to be the minimum friction coefficient needed to keep a contact point fixed. The evaluation of contact slippage is equivalently to the comparison between *necessary sticking friction coefficient* and environment friction coefficient.

With these two *catastrophic events*, an overall objective function can be clearly formulated to reduce the probability that either of these disastrous events happens. The probability of each event is calculated using the cumulative distribution function (CDF) of a normal distribution with some estimated means and standard deviations. Specifically, for each value X, $P(y \geqslant X)$ is the value of the CDF(y) of a normal distribution model $X \sim N(Avg(x), Std(x)^2)$ with mean $Avg(x)$ and standard deviation $Std(x)$. Here, $Avg(x)$ and $Std(x)$ can be coarsely estimated by a human engineer (lower values lead to a more conservative controller) or measured empirically. Then, the overall probability that at least one of the two *catastrophic events* occurs:

$$J = 1 - (1 - P(I \geqslant I_{\text{crit}}))(1 - P(\mu_{\text{foot}} \geqslant \mu^*_{\text{foot}}))(1 - P(\mu_{\text{hand}} \geqslant \mu^*_{\text{hand}})) \quad (2.1)$$

where $P(I \geqslant I_{\text{crit}})$ measures the probability of first *catastrophic* event by comparing whether the actual impact I exceeds the critical amount I_{crit} needed to damage the robot. $P(\mu_{\text{foot/hand}} \geqslant \mu^*_{\text{foot/hand}})$ evaluates the probability of occurrence of second *catastrophic* event by comparing the *necessary sticking friction coefficient* at the hands and feet μ_{foot} and μ_{hand} of a hypothetical trajectory with the actual environmental friction coefficients μ^*_{foot} and μ^*_{hand}.

The computation of actual impact is shown in Equation. 1.13 and the evaluation of *necessary sticking friction coefficient* will be detailed in Sec. 2.3.3.

- Constraints:
  - Joint Limits (Eqn. 1.8)
  - Dynamics (Eqn. 1.9)
  - Coulumb Friction is formulated as "soft" constraint in objective function (Eqn. 1.10).
  - Impact Mapping (Eqn. 1.13)

Then an explicit trajectory optimization problem is written as,

$$\min_{\boldsymbol{u}(t)} \quad J(\boldsymbol{u}(t)) \tag{2.2a}$$

$$\text{s.t} \quad (1.8), (1.9), (1.10), (1.13) \tag{2.2b}$$

This trajectory optimization problem is transcribed into a nonlinear programming problem and an off-the-shelf optimization solver is adopted to find a locally optimal controller that brings the robot hand into contact with the environment and then stabilizes the post-impact dynamics.

2.3 Techniques for Real-time Planning

Humanoid robots generally have a large amount of DOFs. For a trajectory optimization problem with this amount of decision variables to be optimized, its computation is too expensive to be finished within real-time. To address this computational time issue, three techniques are proposed to accelerate this process to achieve a real-time planning ($< 0.1\,s$) performance with a standard PC.

2.3.1 Simplified Three-link Model

Instead of directly making use of humanoids' whole body joint actuators, actuators at significant positions, such as shoulder and hip, are selected for hand contact planning and control. With this concentration, the robot dynamics in the falling plane is approximated as a planar 3-link model. This model is computationally light but complex enough to allow for the use of pre-impact inertia shaping to reduce impact and the use of post-impact compliance to achieve closed loop stability.

Humanoid's full-body is approximated as a rigid, planar three-link inverted pendulum that captures only a few essential degrees of freedom in the falling plane. Joints with unessential degrees of freedom are locked to reduce the modeling error from robot to the pendulum. The pendulum rests on a single point contact at the foot, and after impact the hand will make a fixed contact with the wall (Fig. 2.2). The three DOFs of this model are denoted θ, α, and β, and correspond to the foot center of rotation (which is not assumed actuated), the hip, and the shoulder respectively. The one or two points of contact are assumed to be rigid, point contact, with a Coloumb friction model. The rigidity of contact is assumed to facilitate the computation of the collision impulse and post-impact state using the assumption that the contact forces at the hand and foot are impulsive [HM94]. Hence, the configuration is $\boldsymbol{q} = [\theta, \alpha, \beta]^T$ and the control is $\boldsymbol{u} = [u_\alpha, u_\beta]$.

This model strikes a balance between simplicity and complexity. Benefits of its simplicity include:

- Only two actuated links are assumed in the pre-impact dynamics, which reduces the dimensionality of trajectory optimization.

- The post-impact system becomes a 1 DOF closed-chain system, which ensures a simple analytic expression for the post-impact controller. It also becomes easy to compute the *necessary sticking friction coefficients* after impact.

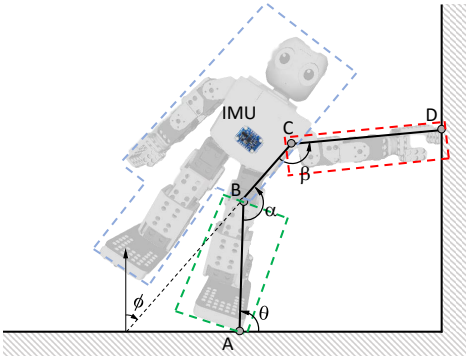


FIGURE 2.2: Three-link model used in this work, illustrated in post-impact. The robot is divided into three blocks, the stance leg, torso, and contact arm, each of which is modeled as a rigid link. The free arm and swing leg are assumed to be fixed to the torso.

- It is practical to precompute a 5-D database that saves the *necessary sticking friction coefficients* of the post-impact controller for all possible post-impact states. Applying the same technique to a 4 link-model would require a 7-D database, which is more time consuming and memory intensive.

It is also sufficiently complex to:

- Match a wide range of initial conditions.
- Allow the robot to use inertia-shaping to reduce collision impact (such as by lunging the hand outward).
- Allow the robot to use compliance during post-impact stabilization, whereas a 2-link model would stop instantaneously.

Pre-impact Dynamics

Its pre-impact system dynamics can be written as ,

$$D\ddot{\boldsymbol{q}} + CG = B\boldsymbol{u} \tag{2.3a}$$

$$\begin{bmatrix} D_{00} & D_{01} & D_{02} \\ D_{10} & D_{11} & D_{12} \\ D_{20} & D_{21} & D_{22} \end{bmatrix} \begin{bmatrix} \ddot{\theta} \\ \ddot{\alpha} \\ \ddot{\beta} \end{bmatrix} + \begin{bmatrix} CG_0 \\ CG_1 \\ CG_2 \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} u_\alpha \\ u_\beta \end{bmatrix} \tag{2.3b}$$

where $D_{00}, D_{01}, ..., D_{22}$ and CG_0, CG_1, CG_2 can be easily derived with Newtonian Mechanics or Lagrangian Mechanics.

Post-impact Dynamics and Stabilizing Controller

After the hand contact, three-link model becomes a closed-chain where there is only one degree of freedom. This enables the reduction of the dynamics in Eqn 2.3 to be a linear function of a single variable. If post-impact system is parameterized at shoulder joint, its system dynamics is reduced to be a linear function of the shoulder torque u_β:

$$\ddot{\beta} = f(\beta, \dot{\beta}) + g(\beta, \dot{\beta})u_\beta \tag{2.4}$$

where $g(\beta, \dot{\beta})$ is the linear term and $f(\beta, \dot{\beta})$ is the constant term.

As long as $g(\beta, \dot{\beta}) \neq 0$ (verified via experiment), a stabilizing controller can be designed with feedback linearization technique to dampen post-impact velocity $\ddot{\beta} = -K\dot{\beta}$ where K is the derivative gain. The gain is chosen such that $\forall|\dot{\beta}| \in [\dot{\beta}_{min}, \dot{\beta}_{max}]$, $|K\dot{\beta}|$ remains within the joint acceleration bound $\ddot{\beta}_{max}$. Here the bounds of joint angular velocities and accelerations are determined based on the configuration of the motor on the robot. This controller also leads to an analytic expression of how the post-impact system evolves

$$\beta(t) = \beta^+ + \dot{\beta}^+(1 - e^{-Kt})/K \tag{2.5}$$

where β^+ is the post-impact angle and $\dot{\beta}^+$ is the post-impact angular velocity.

2.3.2 Direct Shooting Method

Direct shooting method addresses the system dynamics constraint using a numerical simulation of the control states. This simulation updates state $\boldsymbol{x}_{k+1}$ based on state $\boldsymbol{x}_k$ and control $\boldsymbol{u}_k$. Each control $\boldsymbol{u}_k$ can be integrated using first-order Euler integration or higher accurate integration methods such as Runge-–Kutta methods.

Compared with direct collocation method, direct shooting method can outperform it in small-size problems. For real-time planning for fall mitigation, the total time budget is about one-tenth of a second, so any tactics that can lead to a reduction of computational time should be adopted.

Since the impact time is not fixed, a direct shooting method integrates the pre-impact dynamics forward with the chosen control sequence until impact is detected. Then an impact mapping is used to get the post-impact state from the pre-impact state. The magnitude of change of the linear momentum is the collision impulse $\bar{\boldsymbol{\lambda}}$. For a post-impact trajectory, the *necessary sticking friction coefficients* μ_{foot} and μ_{hand} are then calculated.

For its implementation in this hand contact planner, $\boldsymbol{u}(t)$ is taken as a piecewise constant control sequence of length N, with each segment lasting for Δ_t. The time step Δ_t is determined heuristically. To seed the optimizer, a seed control $\boldsymbol{u}_s$ is determined through an inverse dynamics approach to reach heuristic reference angles $(\alpha_{ref}, \beta_{ref})$ with the arm extended in the direction of falling. Eqn. (2.3) can be rewritten as

$$\ddot{\boldsymbol{q}} = D^{-1}(B\boldsymbol{u} - CG)) \tag{2.6}$$

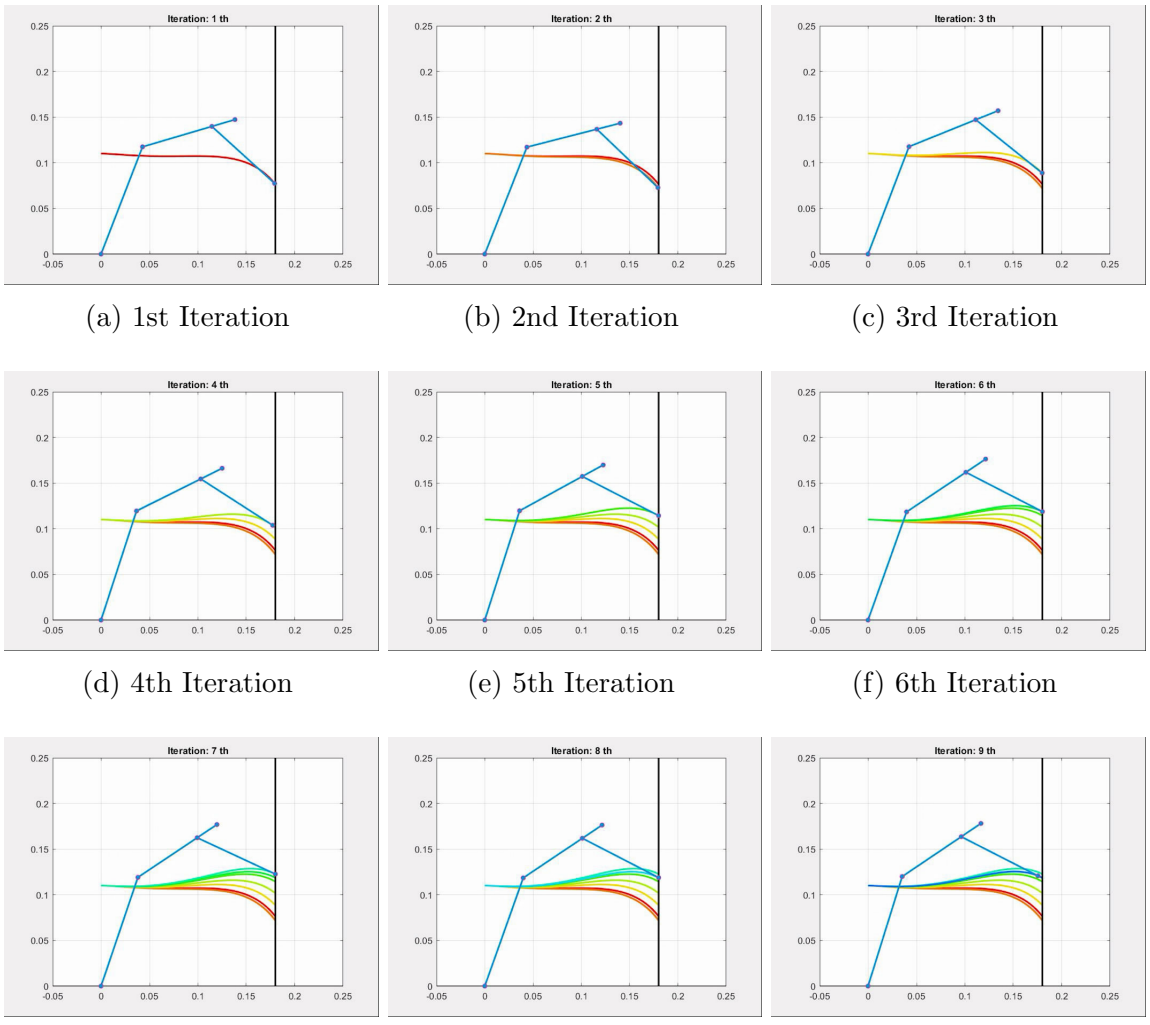
so that the equations of motion for α and β can be fully controlled via a second order linear system with gains $(k_{P\alpha}, k_{P_\beta}, k_{D_\alpha}, k_{D_\beta})$ as follows:

$$\begin{aligned} \begin{bmatrix} \ddot{\alpha} \\ \ddot{\beta} \end{bmatrix} &= B^T D^{-1}(B\boldsymbol{u} - CG) \\ &= \begin{bmatrix} -k_{P\alpha}(\alpha - \alpha_{ref}) - k_{D\alpha}\dot{\alpha} \\ -k_{P\beta}(\beta - \beta_{ref}) - k_{D\beta}\dot{\beta} \end{bmatrix} \end{aligned} \tag{2.7}$$

With the desired acceleration, the control $\boldsymbol{u}$ at each point in time is determined by solving this linear system, which has a solution because $B^T D^{-1} B$ is full rank. Then this $\boldsymbol{u}$ is plugged back in to Eqn (2.3) to integrate the pre-impact dynamics. The

seed control sequence $\boldsymbol{u}_s$ is then determined by discretizing the pre-impact control trajectory at a given resolution.

A numerical optimizer with the quasi-Newton method is used to minimize the objective function over the control trajectory. Fig. 2.3 illustrates the shooting procedures to find optimal solution.



(a) 1st Iteration (b) 2nd Iteration (c) 3rd Iteration

(d) 4th Iteration (e) 5th Iteration (f) 6th Iteration

(g) 7th Iteration (h) 8th Iteration (i) 9th Iteration

FIGURE 2.3: Illustration of direct shooting method with trajectories of hand contact link during optimization iteration. The optimal hand contact is determined after 9 iterations.

2.3.3 Precomputation of Necessary Sticking Friction Coefficients

Necessary Sticking Friction Coefficient

Successful fall recovery requires both contact points to be fixed during the post-impact motion, which requires the friction force to be strictly less than the maximum friction that can be supplied by the environment. This can be equivalently converted into a comparison between the environment friction coefficient and *necessary sticking friction coefficient*. For a given trajectory, *necessary sticking friction coefficient* is the minimum friction coefficients that would be required to keep the robot from slipping.

One challenge to resolve *necessary sticking friction coefficient* is that at each point in time the contact equations are indeterminate. For the planar robot model making contact at foot and hand, there are 4 force variables ($\boldsymbol{F}_{\text{foot}} \in \mathbb{R}^2$, $\boldsymbol{F}_{\text{hand}} \in \mathbb{R}^2$) but only 3 constraints (2 from linear momentum balance (Eqn 2.8) and 1 from angular momentum balance (Eqn 2.9)). In particular, the three-link mechanism can apply a continuum of internal forces via joint torques. Then, an analytical method is used to determine the internal forces to keep the necessary sticking friction coefficient at a minimum.

The forces at foot contact point and hand contact point (Fig. 2.2) must respect linear momentum balance

$$\boldsymbol{F}_{\text{foot}} + \boldsymbol{F}_{\text{hand}} = \sum_{i=1}^{3} m_i \boldsymbol{a_i} - m\boldsymbol{g} \tag{2.8}$$

and angular momentum balance (taking foot contact point to be the origin)

$$(\boldsymbol{r}_{\text{hand}} - \boldsymbol{r}_{\text{foot}}) \times \boldsymbol{F}_{\text{hand}} = \sum_{i=1}^{3} \boldsymbol{r}_{m_i} \times m_i(\boldsymbol{a}_i - \boldsymbol{g}) + \sum_{i=1}^{3} I_i \ddot{\boldsymbol{\Omega}}_i - \boldsymbol{u}_{\boldsymbol{\beta}} \tag{2.9}$$

where $\boldsymbol{F}_{\text{foot}} = [F_{\text{foot}_x}, F_{\text{foot}_y}]^T$ and $\boldsymbol{F}_{\text{hand}} = [F_{\text{hand}_x}, F_{\text{hand}_y}]^T$ are the contact forces at foot contact and hand contact, m is the total mass of the robot, $\boldsymbol{g}$ is the gravity

vector, and for link i, $\boldsymbol{r}_{m_i}$ is the position of its center of mass relative to A, $\boldsymbol{a}_i$ is the acceleration vector of its center of mass, I_i is its moment of inertia and $\ddot{\boldsymbol{\Omega}}_i$ is its angular acceleration vector, and $\boldsymbol{u_\beta} = [0, 0, u_\beta]^T$. These two balance equations can be rearranged into a linear form

$$W\boldsymbol{F} = \boldsymbol{\omega} \tag{2.10}$$

where W is a 3×4 time-dependent matrix, $\boldsymbol{F}^T = [\boldsymbol{F}_{\text{foot}}^T, \boldsymbol{F}_{\text{hand}}^T]$, and $\boldsymbol{\omega}$ is a 3×1 vector stacking the right-hand sides of (2.8) and (2.9). The solution to (2.10) is a combination of external force (particular solution) and internal force (general solution),

$$\boldsymbol{F}(t) = \boldsymbol{F}_p(t) + c\boldsymbol{F}_g \tag{2.11}$$

where c is an arbitrary scalar, $\boldsymbol{F}_p(t) = [F_{\text{foot}_{x_p}}, F_{\text{foot}_{y_p}}, F_{\text{hand}_{x_p}}, F_{\text{hand}_{y_p}}]^T$ is solved with Pseudo-Inverse, and $\boldsymbol{F}_g$ is in the null space of W. In this problem the rank of W is 3, so the null space is 1-D. Specifically, $\boldsymbol{F}_g$ consists of two constant equal and opposite 2-D internal forces parallel to $(\boldsymbol{r}_{\text{hand}} - \boldsymbol{r}_{\text{foot}})$:

$$\boldsymbol{F}_g = \begin{bmatrix} \boldsymbol{F}_{\text{foot}_g} \\ \boldsymbol{F}_{\text{hand}_g} \end{bmatrix} \propto \begin{bmatrix} (\boldsymbol{r}_{\text{hand}} - \boldsymbol{r}_{\text{foot}}) \\ -(\boldsymbol{r}_{\text{hand}} - \boldsymbol{r}_{\text{foot}}) \end{bmatrix}. \tag{2.12}$$

where $\boldsymbol{F}_{\text{foot}_g} = [F_{\text{Foot}_{x_g}}, F_{\text{Foot}_{y_g}}]^T$ and $\boldsymbol{F}_{\text{hand}_g} = [F_{\text{Hand}_{x_g}}, F_{\text{Hand}_{y_g}}]^T$ are the internal force vectors at foot contact point and hand contact point. $\boldsymbol{F}_g$ vector is then normalized so that c gives the magnitude of the internal force.

Letting now $\boldsymbol{F}(t, c)$ be a function both of time t and internal force magnitude c, the friction coefficients at foot contact and hand contact are respectively

$$\mu_{\text{foot}}(t, c) = \left| \frac{F_{\text{hand}_x}(t, c)}{F_{\text{hand}_y}(t, c)} \right|, \; \mu_{\text{hand}}(t, c) = \left| \frac{\boldsymbol{n}_{\text{hand}} \times \boldsymbol{F}_{\text{hand}}(t, c)}{\boldsymbol{n}_{\text{hand}}^T \boldsymbol{F}_{\text{hand}}(t, c)} \right| \tag{2.13}$$

where $\boldsymbol{n}_{\text{hand}}$ is the surface normal vector at hand contact with its polar angle denoted as ψ. Now an optimal value of c should minimize the sum of μ_{foot} and μ_{hand}. Hence,

$$c^*(t) = \arg\min_c \mu_{\text{foot}}(t, c) + \mu_{\text{hand}}(t, c) \tag{2.14}$$

and hence the *necessary sticking friction coefficients* of the post-impact trajectory are computed as follows,

$$\begin{aligned}\mu(\beta^+, \dot{\beta}^+, r_x^*, r_y^*, \psi) &= \max_t \mu_{\text{foot}}(t, c^*(t)) + \mu_{\text{hand}}(t, c^*(t)) \\ &= \max_t (\min_c \mu_{\text{foot}}(t, c) + \mu_{\text{hand}}(t, c)).\end{aligned} \tag{2.15}$$

where $r_x^* = r_{\text{hand}_x} - r_{\text{foot}_x}$ and $r_y^* = r_{\text{hand}_y} - r_{\text{foot}_y}$. The optimization of (2.14) can be performed analytically, which makes (2.15) easier to evaluate.

Necessary Sticking Friction Coefficients Database Pre-computation

For a given trajectory, the necessary sticking friction coefficients μ given by Eqn 2.15 could be calculated by integrating Eqn 2.5 over time and computing Eqn 2.14, but this would be inefficient. This procedure is accelerated by precomputing a database over the five index variables $(\beta^+, \dot{\beta}^+, r_x^*, r_y^*, \psi)$. Given the robot kinematics and velocity limits, each variable has a bounded valid range, so the necessary sticking friction coefficients from all possible post-impact states are pre-computed and saved into a database. A single database is sufficient to cover any initial state of the robot and environment obstacles of any shape. However, a new database is needed if any of the kinematic parameters of the robot change (such as falling in the sagittal plane vs frontal plane).

The precomputation time and database size depend on the resolution of discretization. In our experiments, each of the five variables are discretized into at most 45 quantities. The calculation time is 70 min with OpenMP using four threads. The size for this database is around 120 MB, which easily fits in the RAM of a small embedded PC.

2.4 Simulated Experiment

The speed and quality of our strategy are tested in a MATLAB planar simulation with 8 scenarios. All tests are on a 64-bit Intel Core i7 2.50GHz CPU with 8GB RAM. The optimal control trajectory is computed with the Dlib library's numerical optimization solver with the stopping strategy to be a comparison between a termination threshold $\epsilon = 1e^{-7}$ and the change in the objective function from one iteration to the next iteration [Kin09]. The robot parameters used are shown in Table 2.1, and Table 2.2 shows the weight coefficients and heuristic values used for the simulation.

Table 2.1: Model Parameters for Simulation Experiment

	Mass (kg)	Length (m)	Moment of Inertia $(kg \cdot m^2)$
Leg	0.12	0.125	3.1677e-04
Torso	0.53	0.070	8.8277e-04
Arm	0.05	0.100	4.4271e-05

Table 2.2: Misc. Coefficients and Heuristic Values for Simulation Experiment

Controller Gains		Target Values		Failure Parameters	
k_{P_α}	250	α_{ref1}	$5\pi/6\,rad$	$Avg(I)$	$0.35\,N \cdot s$
k_{P_β}	250	β_{ref1}	$5\pi/8\,rad$	$Std(I)$	$0.1\,N \cdot s$
k_{D_α}	20	α_{ref2}	$\pi/2\,rad$	$Avg(\mu_A)$	0.5
k_{D_β}	20	β_{ref2}	$3.6\pi/5\,rad$	$Std(\mu_A)$	0.1
K	14			$Avd(\mu_E)$	0.4
				$Std(\mu_E)$	0.1

The eight scenarios illustrate a variety of initial conditions and environments. **Wall 1** and **2** are vertical walls at distance 0.09 m and 0.18 m, respectively. **Wall 3a** and **3b** are a cubic curve passing through (0.2,0.5), (0.07,-0.25) and (0.135,0.125) in which the derivative of this curve vanishes. **Table** and **Flat** are flat surfaces at height 0.1 m and 0.0 m. **Circle a** and **b** provide a floating circular support whose center is at (0.15 m,0.12 m) and 0.05 m radius. The initial conditions in each simu-

Table 2.3: Simulation Initial Conditions for Simulation Experiment

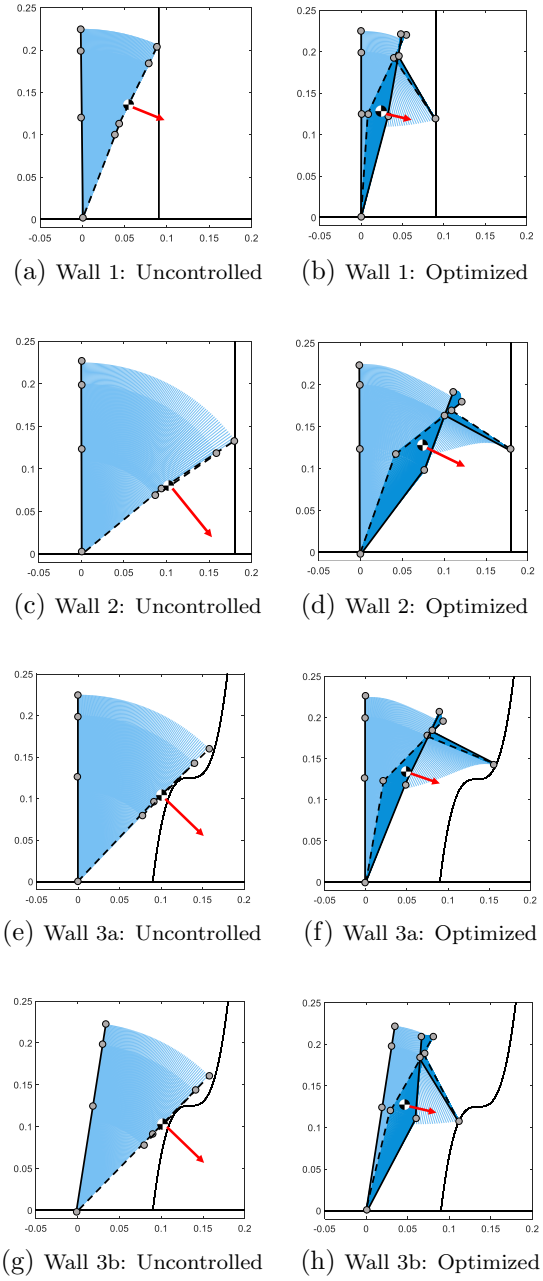
Problem	θ (rad)	α (rad)	β (rad)	$\dot{\theta}$ (rad/s)	$\dot{\alpha}$ (rad/s)	$\dot{\beta}$ (rad/s)
Wall 1	1.57	3.14	0	-2.0	-2.0	0
Wall 2	1.57	3.14	0	-2.0	-2.0	0
Wall 3a	1.57	3.14	0	-2.0	0.0	0
Wall 3b	1.41	3.14	0	-2.0	0.0	0
Table	1.41	3.14	0.78	-2.0	0.0	0
Flat	1.41	3.14	0.78	-2.0	0.0	0
Circle a	1.57	3.14	0	-2.0	0.0	0
Circle b	1.57	3.14	-0.62	-2.0	0.0	0

lation is listed in Table 2.3. In all cases except for **Table** and **Flat**, the control is initialized using the method described in Section 2.3.2 with heuristic target values $\alpha_{ref} = \alpha_{ref1}$ and $\beta_{ref} = \beta_{ref1}$. In **Table** and **Flat**, $\alpha_{ref} = \alpha_{ref2}$ and $\beta_{ref} = \beta_{ref2}$ are used as given in Tab. 2.2.

The optimization results are listed in Table 2.4. Timing is indicated in the Time column, showing that each example is optimized in under 0.1 s. The Impact, μ_{foot}, and μ_{hand} indicate the objective function components in the optimized trajectory. For comparison, the Heuristic Objective column gives the objective function value using the heuristic seed trajectory (raising the arm) as a fraction of the uncontrolled

Table 2.4: Simulation results. Objective values are shown as a % of the uncontrolled value.

Problem	Time (ms)	Impact $(N \cdot s)$	μ_{foot}	μ_{hand}	Heuristic Objective (%)	Optimized Objective (%)
Wall 1	47	0.199	0.246	0.104	13.03	7.31
Wall 2	87	0.370	0.356	0.093	85.58	61.31
Wall 3a	74	0.282	0.347	0.072	66.47	29.67
Wall 3b	52	0.211	0.260	0.078	55.95	8.96
Table	53	0.170	0.086	0.110	26.36	3.82
Flat	75	0.326	0.232	0.242	99.95	44.75
Circle a	59	0.176	0.231	0.212	26.68	7.32
Circle b	71	0.175	0.193	0.117	25.47	4.33



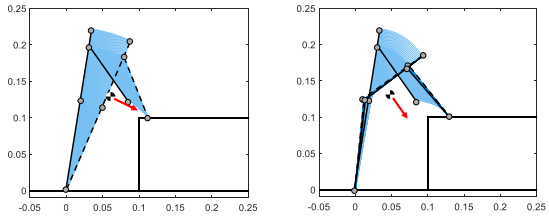
(a) Wall 1: Uncontrolled (b) Wall 1: Optimized

(c) Wall 2: Uncontrolled (d) Wall 2: Optimized

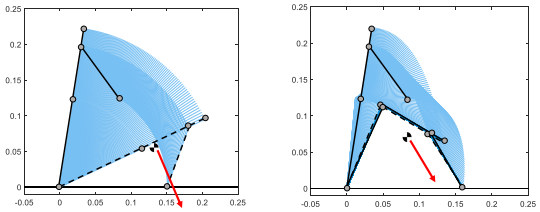
(e) Wall 3a: Uncontrolled (f) Wall 3a: Optimized

(g) Wall 3b: Uncontrolled (h) Wall 3b: Optimized

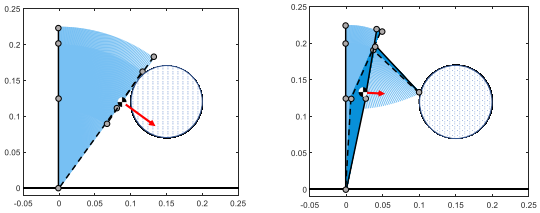
FIGURE 2.4: Simulation of uncontrolled and optimal controllers for examples **Wall 1** to **Wall 3b**. For the controlled motion, we draw the initial state (solid), pre-impact state (dashed) and the steady state (solid). The light blue trace is the pre-impact motion and the dark blue trace is the post-impact motion. The red arrow denotes the velocity of the robot's center of mass at the pre-impact state (scaled by 0.1).



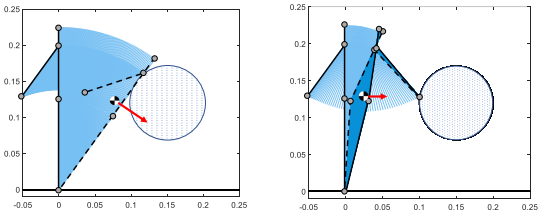
(a) Table: Uncontrolled (b) Table: Optimized



(c) Flat: Uncontrolled (d) Flat: Optimized



(e) Circle a: Uncontrolled (f) Circle a: Optimized



(g) Circle b: Uncontrolled (h) Circle b: Optimized

FIGURE 2.5: Simulation of uncontrolled and optimal controllers, examples **Table** to **Circle b**.

trajectory, and the Optimized Objective column gives the optimized value. These two columns show that although the heuristic is clearly better than uncontrolled falling, our optimization method can reduce the probability of failure often by several factors.

Traces of these simulations in the 3-link model are shown in Fig. 2.4 and Fig. 2.5. In each figure, the left column shows the uncontrolled motion and the right shows

the optimized motion. In every case, the optimized velocity of the center of mass at pre-impact is lower than the uncontrolled velocity, showing arm extension and a backward movement of the hip. The post-impact motion then zeroes out the residual velocity. Comparing **Wall 1** to **2**, the robot reaches its hand forward to catch itself against the wall, and reaching is more extreme for further walls. Comparing **Wall 3a** and **3b**, the method chooses a different contact point depending on the initial velocity. For the **Flat** case, the optimized falling reduces the impact velocity with a bent-over pre-impact posture in which the robot's center of mass remains relatively high above the ground. This result is consistent with observations of optimal falling trajectories from prior work [YG14b].

2.5 Self-contained Fall Recovery System

This section presents the implementation of a real-time fall recovery system that uses hand contact with the environment to prevent a humanoid robot from falling. This system is capable of 1) fall detection and fall direction prediction, 2) using hand contact to stabilize the robot, 3) if physically possible, utilizing a push-up motion to recover to the standing posture. The diagram of the hardware and control flow of the proposed system is in Fig. 2.6.

2.5.1 Hardware Setup and System Integration

- Darwin Mini Robot: Our experimental platform is the ROBOTIS Darwin Mini, a small humanoid robot with 16 actuated joint motors (Dynamixel XL 320). The motor encode can read the motor's present angle and angular velocity. These motors can be directly controlled by the OpenCM 9.04 C board featuring a 32-bit ARM Cortex-M3 processor. The baudrate between the board and motor is configured to be 1Mbps. This board supports the serial data transmission via Tx, Rx and Bluetooth (BT210) with a maximum baudrate

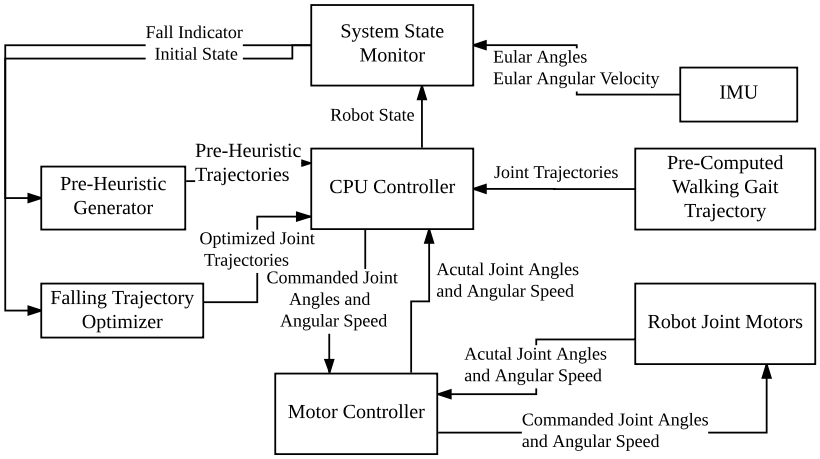


FIGURE 2.6: Diagram of the hardware and control flow of the proposed system.

of 115200bps. Two ROBOTIS touch sensors are connected to this board to monitor the status of the robot hand contact.

- Raspberry Pi 3: Raspberry Pi 3 (Rpi3) is chosen as the main CPU due to its size, cost and sufficient computational capability. Raspbian Jessie is the OS for Rpi3. To increase the efficiency of the data transmission between the robot and Rpi3, the two microcontroller boards are connected in both Bluetooth (OpenCM sending data to Rpi3) and Tx, Rx (Rpi3 sending data to OpenCM). This central computational unit undertakes all the calculation including fall detection and pre-impact optimal controller computation.

- IMU-Adafruit BNO055: The IMU provides readings of Euler angles and angular velocities at 100Hz. However, the default connection port between the Rpi3 and BNO055 has been occupied for the data transmission between RPi3 and OpenCM. An additional Arduino UNO board is used to read the IMU data and send the data via Universal asynchronous receiver-transmitter (UART) to Rpi3 with a 115200 bps.

2.5.2 Fall Detection

Fall detection is a critical component of fall recovery. The earlier an inevitable fall is accurately detected, the more likely the robot will be stabilized. The proposed system uses an inverted pendulum model for fall detection due to its simplicity to satisfy the strict real-time requirement. This pendulum is modeled from the foot edge where the robot rotates around to the center of the mass of the robot (Fig. 2.7). The foot edge location is estimated via kinematics given the current IMU and joint encoder readings.

In the falling plane, if the robot is not falling, the kinetic energy of the robot at each time is less than the critical kinetic energy T_{crit} which is the minimum kinetic energy that the inverted pendulum needs to move from its current angle position to its unstable equilibrium point. Therefore,

$$T_{\text{crit}} = MgL(1 - \cos(\gamma)) \tag{2.16}$$

where M is the total mass of the robot, g is the gravitational constant, L is distance from the foot edge to the center of mass of the robot and γ is angle between the pendulum and the vertical axis in the falling plane, . Meanwhile, the kinetic energy

(a) Falling in Sagittal Plane (b) Falling in Frontal Plane

FIGURE 2.7: Falling detection with inverted pendulum model in sagittal plane and frontal plane. The red arrow denotes the velocity at the center of the mass.

T at each instant of time is

$$T = \frac{1}{2} M L^2 \dot{\zeta}^2 \tag{2.17}$$

where ζ is the angle between the pendulum and the horizontal axis, and $\dot{\zeta}$ is the derivative of ζ. A fall is said to be detected if $T > T_{crit}$.

2.5.3 Push-up Recovery

After fall stabilization, the robot will remain in a steady state and can either 1) wait to be relocated by human to start a new gait, or 2) recover to an upright position by pushing off of the wall. We use a flexing motion of the elbow to allow the robot to gain sufficient momentum to recover a standing posture.

Suppose the hand contact point remains fixed during the push recovery process. Again we apply a three-link model to analyze this push recovery motion, where the three links are the body link (foot rotational point to the shoulder), upper arm (shoulder to elbow) and lower arm (elbow to hand), θ^* is the angle between the ground and the body link, α^* is the angle between the body link and upper arm and β^* is the angle between the upper arm and the lower arm (Fig. 2.8). The mass of the arm is assumed negligible compared to the mass of the body so the behavior of

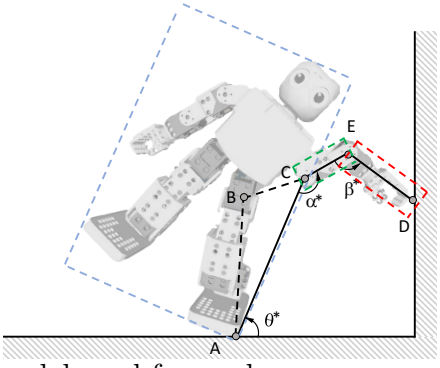


FIGURE 2.8: Three-link model used for push-up recovery. The robot is divided into three blocks, the body, upper arm, and lower arm, each of which is modeled as a rigid link. The upper arm is assumed to be fixed to the body. The two black dashed lines are the stance leg link and torso link from the post-impact model.

the robot can be approximated as an inverted pendulum.

To gain the momentum needed to move away from the environment support, the robot can first bend its elbow to reach an angle β^*_{min} and then stretch the elbow angle to its initial value $\beta^* = 180^\circ$ with a commanded angular velocity $\dot{\beta}^*_{cmd}$. In this paper, we use a heuristic approach to determine this distance range, β^*_{min} and $\dot{\beta}^*_{cmd}$. However, this mechanism works under a constraint of the robot tilt angle in the falling plane. If that angle is beyond a feasible range, the push up will not be able to generate enough momentum for the robot to recover.

2.5.4 Real-world Experiment

In this section, we present results of applying our hand contact planner in 8 scenarios. The robot parameters used are shown in Table 2.5, and Table II shows the weight coefficients and heuristic values used in the experiment.

Table 2.5: Three-link Model Parameters for Real-world Experiment

	Mass (kg)	Length (m)	Moment of Inertia $(kg \cdot m^2)$
Leg	0.12	0.125	3.1677e-04
Torso	0.53	0.070	8.8277e-04
Arm	0.05	0.100	4.4271e-05

Table 2.6: Robot State at the Instant of Falling for Real-world Experiment

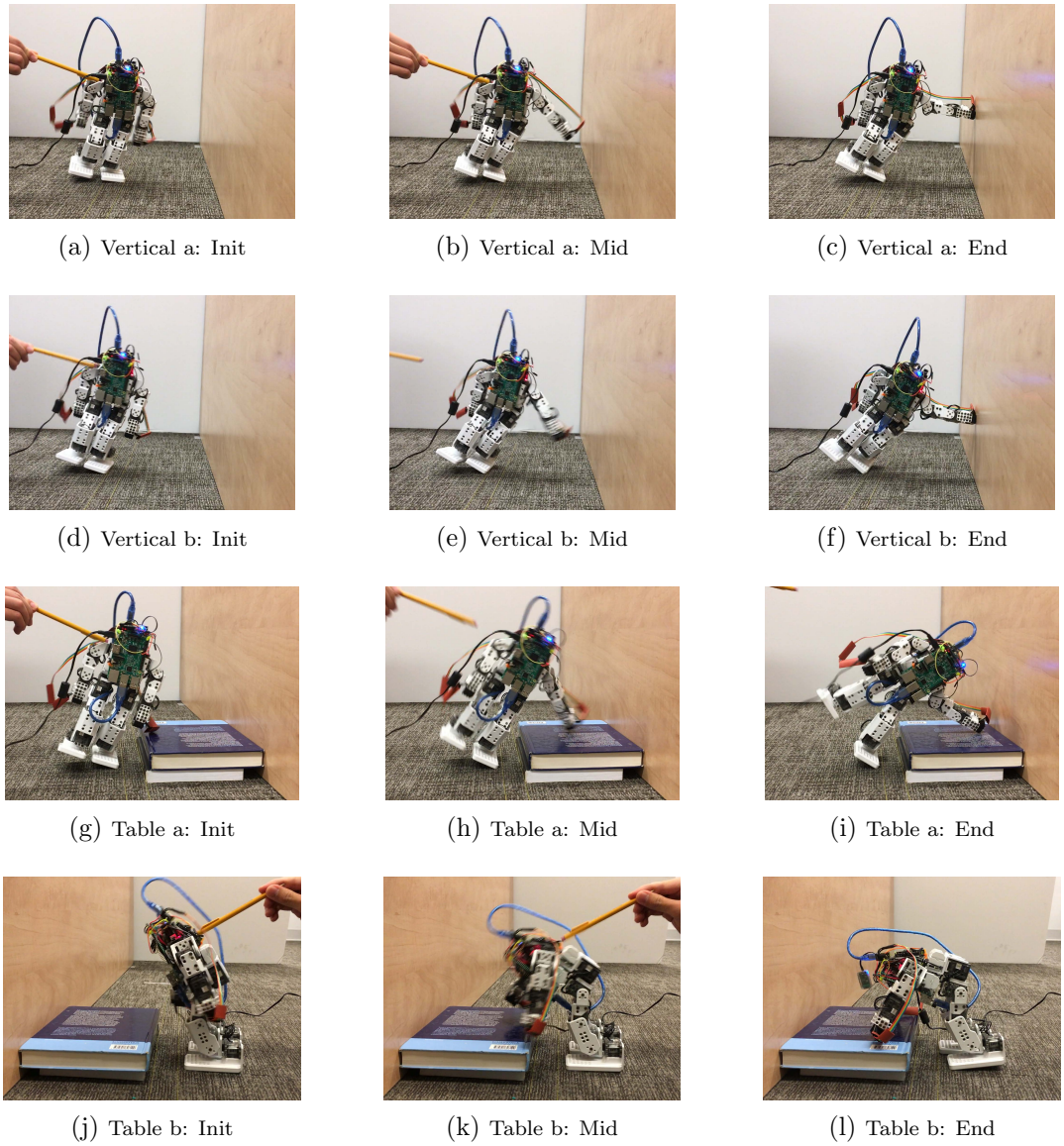
Problem	θ (rad)	α (rad)	β (rad)	$\dot{\theta}$ (rad/s)	$\dot{\alpha}$ (rad/s)	$\dot{\beta}$ (rad/s)
Vertical a	1.48	2.61	0.52	-0.62	0.19	0.00
Vertical b	1.43	2.59	0.52	-1.09	-0.14	0.00
Table a	1.39	2.60	0.52	-3.22	-0.18	0.00
Table b	1.66	2.89	0.17	-2.71	0.24	0.66
Ground a	1.41	2.65	0.52	-2.41	0.14	0.00
Ground b	1.69	2.96	0.14	-3.24	-0.81	1.85
Push Up a	1.47	2.67	0.52	-0.54	0.14	0.00
Push Up b	1.49	2.62	0.52	-0.55	-0.19	0.00

Table 2.7: Model-based Optimization Results for Each Experiment

Problem	Comp Time (ms)	Contact Time (ms)	Impact $(N \cdot s)$	μ_{foot}	μ_{foot}	Optimized Failure %
Vertical a	23	366	0.184	0.503	0.102	17.04
Vertical b	113	358	0.326	0.804	0.429	98.56
Table a	29	386	0.326	0.338	0.304	13.40
Table b	110	391	0.206	0.226	0.110	0.73
Ground a	144	361	0.494	0.351	0.116	67.32
Ground b	54	385	0.527	0.299	0.269	78.10
Push Up a	119	353	0.191	0.548	0.068	30.76
Push Up b	83	368	0.196	0.527	0.083	23.66

Tab. 2.6. For each robot state, an optimal controller is computed in real-time to bring the robot into contact with the environment walls. Tab. 2.7 shows the time for optimization (Comp. time), the time at which contact is made after fall detection (Contact time), the predicted collision impulse, necessary sticking friction coefficients at contact points (μ_{foot} and μ_{hand}), and optimized probability of failure. In all cases the computation time is within $150\,ms$ and the optimal contact is made within $400\,ms$. The next four columns of this table show that besides the three extreme cases (**Vertical b**, **Ground a** and **Ground b**), our optimal controller manages to confidently reduce the probability that a *catastrophic event* would occur.

Fig. 2.9, 2.10, and 2.11 illustrate the trace for each experiment. In each example, the robot is stopped using hand contact with the environment obstacles. Comparing **Vertical a** to **Vertical b**, the robot reaches its hand towards the wall to arrest itself from falling and reaching is further for walls at further distance. Comparing **Table a** and **Table b**, our method can exploit the flat table surface to stabilize the robot and extend the falling stabilization strategy to both the sagittal plane and frontal plane instead of the sole 2-D sagittal plane. Comparing **Ground a** and **Ground b**, our method finds the optimal contact with a bent-over posture in which the robot's center of mass remains relatively high above the ground. This result is consistent



(a) Vertical a: Init (b) Vertical a: Mid (c) Vertical a: End

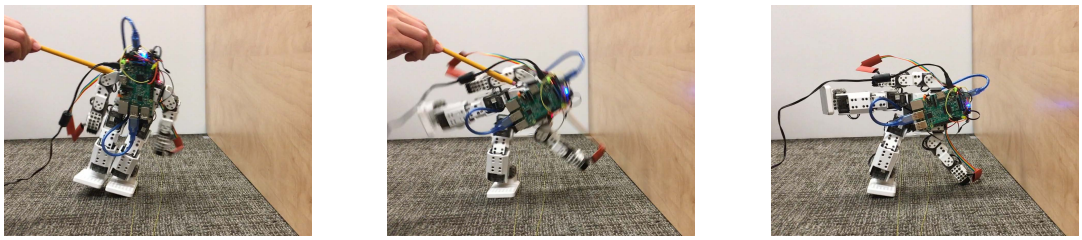
(d) Vertical b: Init (e) Vertical b: Mid (f) Vertical b: End

(g) Table a: Init (h) Table a: Mid (i) Table a: End

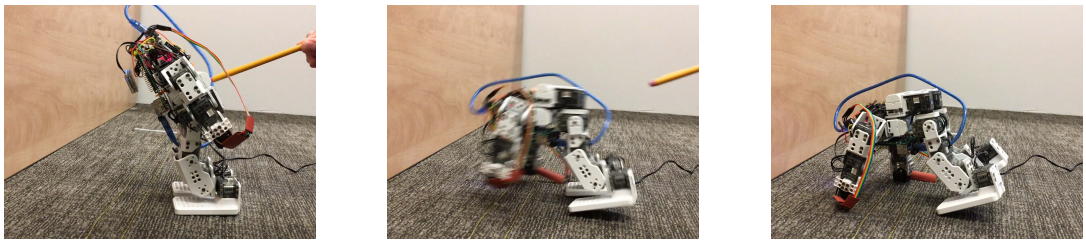
(j) Table b: Init (k) Table b: Mid (l) Table b: End

FIGURE 2.9: Experiments of optimal controllers for examples **Vertical 1** to **Table b**.

with prior work on optimal fall mitigation strategy to minimize collision damage. **Push Up a** and **Push Up b** demonstrate that our proposed push-up strategy can recover the robot to a standing posture and enable the robot to continue on its previous task interrupted by the push. Fig. 2.11 demonstrates this recovery process. Fig. 2.12 shows a representative diagram of the timing of when the fall is detected,

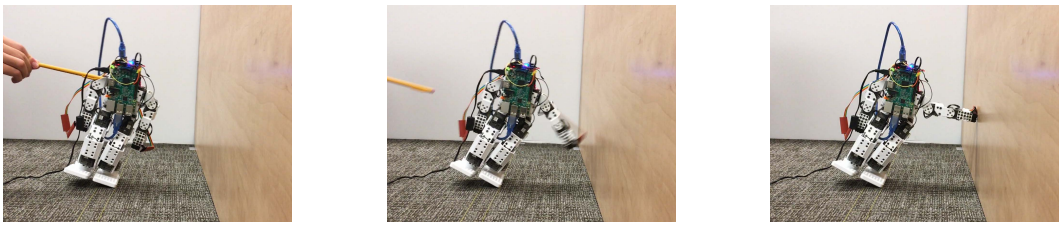


(a) Ground a: Init (b) Ground a: Mid (c) Ground a: End

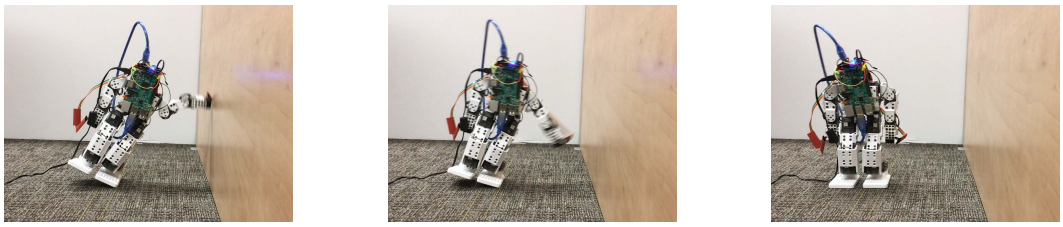


(d) Ground b: Init (e) Ground b: Mid (f) Ground b: End

FIGURE 2.10: Experiments of optimal controllers for examples **Ground a** and **Ground b**.

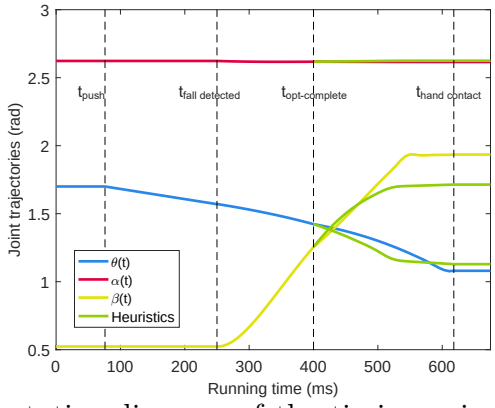


(a) Falling detected (b) Pre-impact contact (c) Stabilization



(d) Elbow bending (e) Elbow releasing (f) Standing posture

FIGURE 2.11: Representative traces of **Push Up a** or **b** case recovery motion



b

FIGURE 2.12: A representative diagram of the timing using **Push Up** case. Vertical dashed lines denote the timing of when the robot is pushed t_{push}, when the fall is detected $t_{fall\ detected}$, when the computation of optimal controller is completed $t_{opt\text{-}complete}$ and when the contact is established $t_{hand\ contact}$. Green curves are the heuristic trajectories used in the pre-impact optimization.

when the pre-heuristic trajectory is executed, and when the optimal trajectory is executed to demonstrate the real-time nature of the proposed strategy.

2.6 Summary

This chapter presents an efficient motion planner for humanoid fall recovery with making hand contact with walls and other environmental obstacles for balance recovery. It is based on a three-link dynamic model and optimal control principles to choose a hand contact point. This optimal control approach is designed to maintain stability while minimizing the risk of damage to the robot and the risk of contacts slipping. The method works for environmental obstacles of different shapes ranging from vertical walls to flat ground. For the integrated fall recovery system, it is capable of failure detection, fall stabilization and push-up recovery and it aslo overcomes several practical challenges regarding computation and communication delays. Experimental results on the Darwin Mini platform, augmented with additional sensors, show that the proposed falling stabilization system can stabilize the robot with several walls, environmental obstacles, and falling directions.

3

Unified Fall Recovery in Arbitrary Environment

Previous chapter introduces an efficient motion planner for fall recovery in cluttered environment where hand contact is specified to be used. This chapter presents a multi-contact motion planner which generalizes humanoid fall recovery in both open and cluttered environment. This planner unifies existing recovery strategies, such as inertial shaping, protective stepping, and hand contact, and automatically plans one strategy or a combination of strategies to regain robot's balance based on its disturbed state and nearby environment features. This chapter also introduces an efficient and effective algorithm to generate promising initial seeds for trajectory optimization. Experiments demonstrate that the proposed algorithm can generate complex stabilization strategies for a simulated planar robot under varying initial pushes and environment shapes [1].

[1]This chapter is reproduced from Shihao Wang and Kris Hauser, "Unified Multi-Contact Fall Mitigation Planning for Humanoids via Contact Transition Tree Optimization," in International Conference on Humanoid Robots, Beijing, China, 2018. This research is supported by NSF grant NRI #1527826.

3.1 Introduction

What is the *optimal* strategy by which a robot should mitigate an impending fall? The answer to address this question is not straightforward since the planning of protective strategies depends on many reference variables, such as the magnitude of external pushes, robot's kinetic energy at the falling time, nearby environment features, current contact status and so on. Theoretically speaking, existing strategies, such as *inertial shaping*, *protective stepping* and *hand contact*, should be able to mitigate any robot fall damage as long as they are combined in a suitable sequence. However, these strategies have been considered in isolation and there have been limited attempts to unify multiple strategies into a single fall mitigation system. As a result, it still remains to be unclear about how the robot should be to mitigate an inevitable fall since a reliable high-level planning of protective strategies cannot be generated. To address this problem. we make the following hypothesis that robot should always choose the *strategy that yields the fastest decrease in kinetic energy*. In order to study this hypothesis, we develop a planning method that simultaneously generates the contact sequence and optimized whole-body trajectories to achieve a stabilizing multi-contact trajectory.

This research aims to extend the current framework on simultaneously planning of contact sequence and robot trajectories with two contributions:

1. A new formulation of planning while searching for contact is proposed. Instead of directly incorporating the contact sequence into the formulation of a single optimization problem, we embrace the tree-search idea to conduct the search for contact sequence layer by layer. In this fashion, each optimization is reduced to a node connection problem while minimizing a cost function.

2. A seed initialization algorithm is proposed to enable the automatic genera-

tion of promising initial values for decision variables to be converged to local optimal.

3.2 Related Work

The problem of balancing a biped or humanoid in response to external disturbance has been an active topic of research for some time [FKH+06, PCDG06]. Strategies proposed to address this topic can be classified into two main categories: *fixed contact* and *contact modification*. Fixed contact strategies aim to recover the robot purely through joint effort to regulate linear and angular momentum, all while maintaining the current contact state. Two representatives are the ankle strategy and hip strategy [Ste07][ARW12]. For larger disturbances it is impossible to recover to a stationary state using joint torques alone. Contact modification strategies reduce momentum by making contact at the robot extremities, which transfers kinetic energy from the robot into the environment. Examples of this type of strategy include protective stepping [SA10a], hand contact [OK07] [MDG+17a], knee contact [JWS12], tripod posture [YG14a], and contact with accessories such as a backpack [LG11] and walking sticks [TK16].

Despite the existence of various disturbance recovery strategies, it still remains unclear which strategy or combination of strategies should be adopted to stabilize a humanoid if an arbitrary pushed is imposed. Stephens determines the decision boundary between strategies using a simplified LIPM model [Ste07]. Our proposed method unifies both fixed contact and contact modification strategies, and can also devise novel contact sequences. It does so by planning trajectories from the initial state to minimize kinetic energy via a multi-contact transition tree approach. This approach is related to other multi-contact planning algorithms, such as manipulation planning in contact configuration space [JX01], multi-modal motion planning for legged robots [HL10], and robot whole-body transition synthesis using motion

capture dataset [MBJA15]. However, these approaches only address quasi-static systems and feasible planning. Our approach is based on a dynamic model of the robot and uses trajectory optimization to generate state-space paths.

3.3 Method

3.3.1 Contact Transition Tree

Given an initial robot state, initial contact mode, and environment geometry, we wish to generate a joint space trajectory and contact sequence to stabilize to a stationary (zero-velocity) state. Our method integrates a high-level tree search, to explore contact sequences, with trajectory optimization, to plan connecting trajectories and self-stabilization trajectories.

A feasible *fixed-contact trajectory* must respect contact constraints of its mode, as well as dynamic constraints, friction limits, torque limits, and joint limits. Let the kinetic energy of a state be $E_k(\boldsymbol{x}) = \frac{1}{2}\dot{\boldsymbol{q}}^T D(\boldsymbol{q})\dot{\boldsymbol{q}}$. We define a stationary state as one in which $E_k(\boldsymbol{x})$ is sufficiently small. Our goal is to produce a *multi-contact trajectory* sequence of modes $\boldsymbol{\sigma}_0, \ldots, \boldsymbol{\sigma}_N$ and a continuous sequence of $N+1$ feasible trajectories starting at $\boldsymbol{x}_0$ and ending at $\boldsymbol{x}_N$. N is not fixed, and there is no restriction on the terminal contact mode.

To build a feasible multi-contact trajectory, our method incrementally builds a *contact transition tree* $\mathcal{T}$, rooted at the initial robot state's mode, and iteratively grows its edges to the most promising stabilizable nodes until a terminal self-stabilization has been achieved. Each tree node $\mathbf{d}_i$ contains three attributes:

1. Contact mode $\boldsymbol{\sigma}(\mathbf{d}_i)$
2. Robot state $\boldsymbol{x}(\mathbf{d}_i)$.
3. Self-motion trajectory $y_{\text{self}}(\mathbf{d}_i)$

The *self-motion trajectory* $y_{\text{self}}(\mathbf{d}_i)$, also denoted by y_i, is a feasible fixed-contact trajectory at $\boldsymbol{\sigma}(\mathbf{d}_i)$ starting at $x(\mathbf{d}_i)$. In other words, it is an inertial shaping trajectory.

Algorithm 1: Contact transition tree search

Input : Initial state $\boldsymbol{x}_0$, mode $\boldsymbol{\sigma}_0$, environment map
Output: Mode sequence: $(\boldsymbol{\sigma}_0 \rightarrow \boldsymbol{\sigma}_1 \rightarrow \cdots \rightarrow \boldsymbol{\sigma}_N)$
Trajectories: $(y_{0,1} \rightarrow \cdots \rightarrow y_{N-1,N} \rightarrow y_N)$

$\mathbf{d}_0 \leftarrow$`Node`$(\boldsymbol{x}_0, \boldsymbol{\sigma}_0)$
$\mathcal{T} \leftarrow \mathbf{d}_0$, $\mathbf{F} \leftarrow \{\mathbf{d}_0\}$
while $|\mathbf{F}| > 0$ **do**
  $\mathbf{d}_i \leftarrow$ `Pop`$(\mathbf{F})$
  $y_i \leftarrow$ `OptSelfMotion`$(\mathbf{d}_i)$
  if $y_i \neq nil$ and $E_k(\text{End}(y_i)) < \epsilon_{E_k}$ **then**
    $y_{\text{self}}(\mathbf{d}_i) \leftarrow y_i$
    Retrieve path from $\boldsymbol{d}_0$ to $\boldsymbol{d}_i$ in $\mathcal{T}$
    return Modes $(\boldsymbol{\sigma}_0 \rightarrow \cdots \rightarrow \boldsymbol{\sigma}_i)$
      Paths $(y_{0,1} \rightarrow \cdots \rightarrow y_{i-1,i} \rightarrow y_i)$
  end
  for $\boldsymbol{\sigma}_j \in$ AdjacentModes$(\boldsymbol{\sigma}(\mathbf{d}_i))$ **do**
    $y_{i,j} =$ `OptTransitionMotion`$(\mathbf{d}_i, \boldsymbol{\sigma}_j)$
    if $y_{i,j} \neq nil$ **then**
      $x_j \leftarrow \text{End}(y_{i,j})$
      If $|\boldsymbol{\sigma}_j| > |\boldsymbol{\sigma}_i|$, $\boldsymbol{x}_j \leftarrow \text{ImpactMap}(\boldsymbol{x}_j, \boldsymbol{\sigma}_j)$
      $\mathbf{d}_j \leftarrow \text{Node}(\boldsymbol{x}_j, \boldsymbol{\sigma}_j)$
      Add $\mathbf{d}_j$ to $\mathcal{T}$ as a child of $\mathbf{d}_i$
      Store $y_{i,j}$ with $\mathbf{e}_{i \rightarrow j}$
      Add $\mathbf{d}_i$ to $\mathbf{F}$ with priority $E_k(\text{End}(y_{i,j}))$
    end
  end
end
return no solution found

3.3.2 Tree Search and Expansion

The following search procedure is used to grow $\mathcal{T}$. Let $\mathbf{F}$ denote the *frontier* nodes, which is implemented as a priority queue sorted by increasing kinetic energy.

1. The node with lowest kinetic energy is extracted from $\mathbf{F}$, and a trajectory optimization will be conducted to calculate y_i.

2. If a stationary endpoint is found, then we are done.

3. Otherwise, the algorithm continues to expand to neighboring contact modes by attempting to find feasible trajectories to those modes.

4. Each successful connection to a neighbor is added as a new edge in $\mathcal{T}$, and each neighboring node is added to $\mathbf{F}$.

Specifically, our algorithm uses a greedy approach in which each trajectory optimization attempts to *minimize the kinetic energy of the endpoint*. This approach tries to eliminate kinetic energy from the initial robot state as quickly as possible, which limits the amount of search needed to find a solution.

Algorithm 1 illustrates the details of tree search and expansion procedure. Lines 5–11 attempt to stop the robot at the current contact mode via a self motion. If the optimization succeeds and the kinetic energy at the end point is within a small tolerance ϵ_{E_k}, we are done. The result traces back from this leaf node to the root node to extract the solution contact sequence and single-contact trajectory sequence. Lines 12-22 validate the connectivity to child nodes using optimization. If no further feasible paths can be found (Line 24), the algorithm terminates with failure.

The main computational tasks are undertaken by two subroutines:

- `OptSelfMotion` optimizes a robot stabilization trajectory at the contact mode $\boldsymbol{\sigma}$ of node $\mathbf{d}_i$.

- `OptTransitionMotion` optimizes a transition trajectory from node $\mathbf{d}_i$ to node $\mathbf{d}_{\text{child}}$.

These will be described in more detail below.

The following minor subroutines are also used:

- Pop takes out the node with the minimum kinetic energy among other nodes in the Frontier $\mathbf{F}$.

- Node creates a node at a given robot state and contact mode.
- End returns the robot state at the end of a trajectory.
- AdjacentModes produces the list of neighbouring contact modes that differ from $\boldsymbol{\sigma}$ by exactly one change of contact. Fig. 3.1 illustrates a representative node expansion example where each hand/foot contact can be modified to produce 4 adjacent nodes.
- ImpactMap calculates the post-impact robot state resulting from impact mapping (Eqn. 1.13).

For each edge $\mathbf{e}_{i\to j}$ from $\boldsymbol{d}_i$ to $\boldsymbol{d}_j$, the nodes may differ by exactly one limb in contact. Each edge also stores a *transition trajectory* y_{ij} starting at $\boldsymbol{x}(\mathbf{d}_i)$ and terminating in $\boldsymbol{x}(\mathbf{d}_j)$. A transition trajectory must satisfy the constraints of $\boldsymbol{\sigma}_i$. If the mode switch $\boldsymbol{\sigma}_i \to \boldsymbol{\sigma}_j$ removes a contact, then the final state must satisfy the dynamic constraints of $\boldsymbol{\sigma}_j$, specifically, that there are valid forces at the contacts active in $\boldsymbol{\sigma}_j$. If the mode switch adds a contact, then the final state must satisfy the kinematic contact conditions of $\boldsymbol{\sigma}_j$

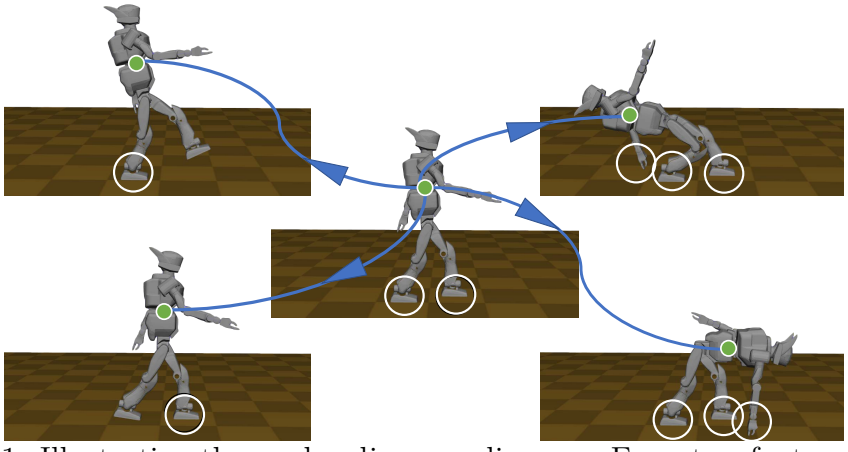


FIGURE 3.1: Illustrating the mode adjacency diagram. From two-foot contact mode LF/RF (center), the robot can switch to one-foot modes LF and RF (left) and two three-contact modes LF/RF/LH and LF/RF/RH (right).

3.3.3 Trajectory Optimization: Stabilization and Transition

Trajectory optimization is central to our method, and we use a direct collocation method that uses a high-accuracy spline representation [PKT16]. As an objective function we minimize the kinetic energy at the end state of the trajectory. We also develop a custom trajectory initialization that works well in practice. Both `OptSelfMotion` and `OptTransitionMotion`, use the same underlying method with only small modifications.

Direct collocation

The variables to be optimized are the time duration T and trajectories of the robot state, control and contact force. After the transcription of these continuous trajectories at N_d equally distributed knot points with timestep $h = \frac{T}{N_d - 1}$, we formulate this trajectory optimization into a non-linear programming (NLP) problem. The inputs to the NLP are timestep h, discretized robot state $(\boldsymbol{x}_1, ..., \boldsymbol{x}_{N_d})$, control $(\boldsymbol{u}_1, ..., \boldsymbol{u}_{N_d})$ and contact force $(\boldsymbol{\lambda}_1, ..., \boldsymbol{\lambda}_{N_d})$.

Due to second-order nature of the robot dynamical system and the holonomic constraints on contact position and velocity, using classic first-order Euler integration to update the robot state tends to cause numerical difficulties. This drawback can be avoided by approximating the robot state and control/contact force trajectories as implicit cubic splines and piecewise linear functions, respectively. A third-order integration accuracy $\mathcal{O}(h^3)$ has been reported with this spline choice [PKT16]. The construction of implicit cubic splines is associated with the system kinematics and dynamics. For a representative position variable q_i, its cubic spline path within a timestep can be expressed

$$q_i(s) = a_p s^3 + b_p s^2 + c_p s + d_p, s \in [0, 1] \tag{3.1}$$

The position $q_i(s)$ and its first time derivative $\frac{dq_i(s)}{dt}$ should match the robot state at

both edges $(\boldsymbol{x}_i, \boldsymbol{x}_{i+1})$. These matching conditions solve the four unknowns in $q_i(s)$ and any intermediate point can be then interpolated. However, the same methodology cannot be used to calculate the cubic spline coefficients of the velocity variable $\dot{q}_i(s)$ since the first order derivative of $\dot{q}_i(s)$, acceleration, is not a variable to be optimized. As a result, we have to adopt a different approach to get its cubic spline.

At sequential knots, states $(\boldsymbol{x}_i, \boldsymbol{x}_{i+1})$, controls $(\boldsymbol{u}_i, \boldsymbol{u}_{i+1})$ and contact forces $(\boldsymbol{\lambda}_i$, $\boldsymbol{\lambda}_{i+1})$ are optimization variables. Instead of enforcing the dynamics constraint (1.9) inside the optimization solver, we directly make use of this constraint to calculate the acceleration determined under the current set of robot state, control and contact force. With accelerations now available at both knots, the cubic spline coefficients of $\dot{q}_i(s)$ can be computed.

The guaranteed satisfaction of the dynamics constraints at knot points enable us to add a collocation point in the middle position ($s = 0.5$) to further decrease the dynamics violation within this interval. With the approximation of the control and contact force to be linear function, their interpolated value at the mid-point is the average of the edge values $\boldsymbol{u}_{mid} = \frac{\boldsymbol{u}_i + \boldsymbol{u}_{i+1}}{2}$, $\boldsymbol{\lambda}_{mid} = \frac{\boldsymbol{\lambda}_i + \boldsymbol{\lambda}_{i+1}}{2}$. Together with the interpolated robot position $\boldsymbol{q}_{mid}$, velocity $\dot{\boldsymbol{q}}_{mid}$ and acceleration $\ddot{\boldsymbol{q}}_{mid}$, a dynamics constraint is imposed at this collocation point

$$\begin{aligned} &D(\boldsymbol{q}_{mid})\ddot{\boldsymbol{q}}_{mid} + C(\boldsymbol{q}_{mid}, \dot{\boldsymbol{q}}_{mid}) + G(\boldsymbol{q}_{mid}) = \\ &J(\boldsymbol{q}_{mid})^T \boldsymbol{\lambda}_{mid} + B\boldsymbol{u}_{mid} \end{aligned} \tag{3.2}$$

By matching the cubic spline to the real trajectory at both knots and collocation, the dynamics constraint violation along this spline is significantly reduced. To get rid of the difficulty in formulating the undifferentiable self-collision avoidance constraint in 3 dimension environment, we assume at this stage the robot locomotion is in its sagittal plane. In addition, we constrain the relative distances of robot's internal joints to be always strictly away from the environmental features such that contact

can only be made at robot's hands and feet.

The overall NLP that is solved is

$$\begin{aligned} \underset{h,\boldsymbol{x}_1,\ldots,\boldsymbol{x}_{N_d},\boldsymbol{u}_1,\ldots,\boldsymbol{u}_{N_d},\boldsymbol{\lambda}_1,\ldots,\boldsymbol{\lambda}_{N_d}}{\text{minimize}} \quad & E_k(\boldsymbol{x}_{N_d}) \\ \text{subject to} \quad & (1.8), (1.9), (1.10), (1.11), (1.12), (1.13), (3.2) \end{aligned} \tag{3.3}$$

which we solve using the SNOPT library [GMS02].

3.3.4 Optimal Seed Initialization

An initial guess is needed for the NLP solver to find a feasible and high-quality solution. This is a nontrivial challenge. Our algorithm actually uses multiple initial guesses of increasing duration. For each duration, a smooth parabolic spline that obeys the initial and terminal constraints is generated. We try solving the NLP when seeded from each of these initial guesses, and terminate when the first feasible solution is found.

For a given duration guess T_i, the initial guess satisfies the kinematic transition constraint and minimizes the violation of the dynamics constraint at each knot point. This procedure is as follows:

1. Take $\boldsymbol{x}_0 = [\boldsymbol{q}_0^T, \dot{\boldsymbol{q}}_0^T]^T$ as a starting point and compute a reference goal configuration $\boldsymbol{q}_{\text{ref}}$ by assuming that $\dot{\boldsymbol{q}}_0$ linearly decreases to zero at T_i so $\boldsymbol{q}_{\text{ref}} = \boldsymbol{q}_0 + \dot{\boldsymbol{q}}_0 T_i + \frac{1}{2}\ddot{\boldsymbol{q}}_{\text{ref}} T_i^2$ where $\ddot{\boldsymbol{q}}_{\text{ref}} = -\frac{\dot{\boldsymbol{q}}_0}{T_i}$.

2. Project $\boldsymbol{q}_{\text{ref}}$ back to the constraint manifold to get the goal configuration $\boldsymbol{q}_g$ and construct a parabolic curve with duration T_i starting at $\boldsymbol{q}_0$ and ending at $\boldsymbol{q}_g$, with initial velocity $\dot{\boldsymbol{q}}_0$.

3. Discretize the parabolic spline into n segments and interpolate the configuration, velocity and acceleration at edges (knots) of the segments.

4. For each knot point, compute the left hand side of dynamics equation (1.9). Solve for $\boldsymbol{u}$ and $\boldsymbol{\lambda}$ in least squares fashion by multiplying the l.h.s. by the pseudo-inverse of $\left[B,\, J_{\boldsymbol{\sigma}}(\boldsymbol{q})^T\right]$ where $J_\sigma(\boldsymbol{q})$ is the Jacobian matrix of active contacts in the post-impact contact mode $\boldsymbol{\sigma}$.

The trajectory duration T_i directly affects the acceleration of the parabolic curve. Smaller values of T_i generally yield larger accelerations, and hence more extreme control and contact forces. We uniformly explore a range of durations T bounded between $[T_{min}, T_{max}]$, divided uniformly into N_{tot} points. Our optimizer explores these options via brute force under increasing duration T_i until a feasible solution has been found or all options are exhausted.

3.4 Examples

We evaluate the effectiveness of the proposed method with a simulated planar model of the HRP-2 robot under varying initial disturbed robot states and environment shapes. All experiments are conducted on a 64-bit Intel Quad-Core i7 2.50GHz workstation with 8GB RAM. The computational time in solving the NLP takes around $3 \sim 5$ min given a suitable initial seed. Robot parameters and optimization coefficients used in our experiment are listed in Tab. 3.1.

Table 3.1: Parameters used in multi-contact experiments

System Parameters		Optimization Coefficients		Tolerances	
n	13	T_{min}	0.25 s	ϵ_{E_k}	0.1 J
m	10	T_{max}	3.5 s	ϵ_{ct}	0.025 m
l	12	N_{tot}	40		
μ	0.35	N_d	8		

Multi-Contact Fall Mitigation

This subsection demonstrates the capability of the proposed algorithm to generate complex multi-contact stabilization strategies in different environments.

Fig. 3.2 shows the contact transition tree produced by our algorithm with a large $85J$ disturbance on flat ground. There is no stabilizing self-motion trajectory with either 0, 1, or 2 contact switches, so the tree is expanded until it finds a solution at depth 3. The optimal contact sequence for this case is: protective stepping + two-hand contact + inertial shaping.

Fig. 3.3 shows the contact transition tree produced for the robot pushed toward a vertical wall with an initial two-foot contact mode. There are no self-motion trajectories at depth 0, but it finds a solution using hand contact, yielding a contact sequence Left Foot + Right Foot→Left Foot + Right Foot + Left Hand.

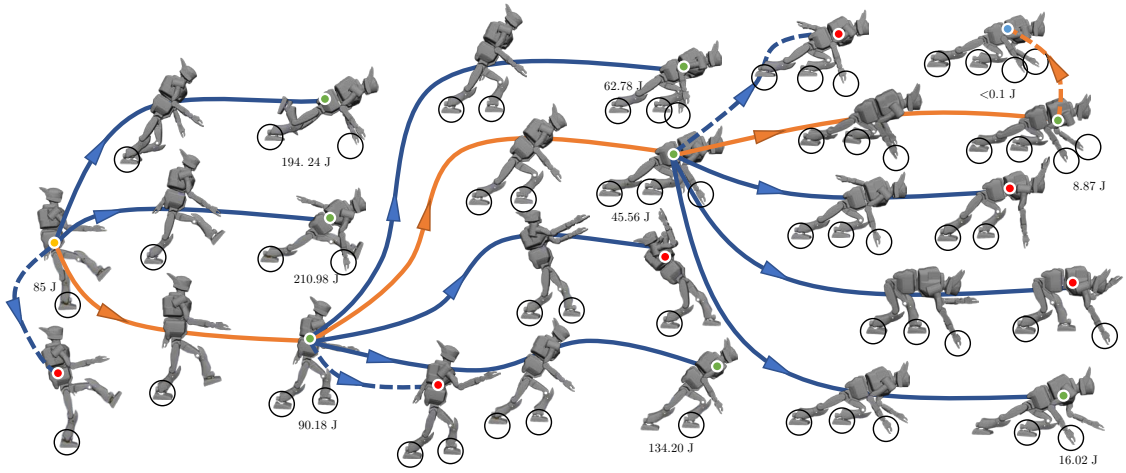


FIGURE 3.2: Contact transition tree on flat ground, starting from initial kinetic energy 85 J. Solution takes a protective step, makes ground contact with both hands and uses inertia shaping to achieve a terminal stabilization.

Change of Strategy with Increasing Initial Energy

This subsection demonstrates how our algorithm can explore the change in fall mitigation strategy necessary to handle pushes of increasing severity. We choose 8 initial

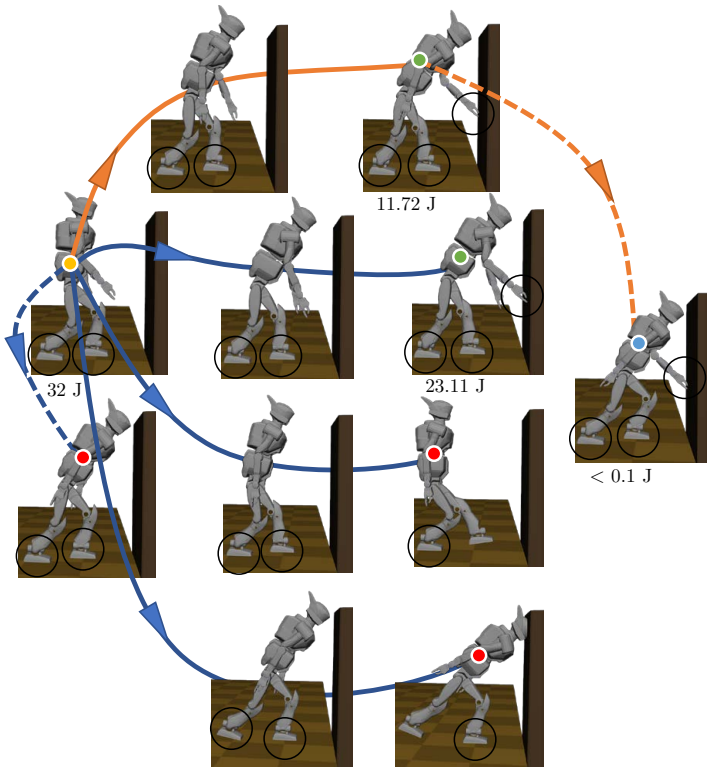


FIGURE 3.3: Contact transition tree with vertical wall, yielding a hand contact strategy.

states whose kinetic energies increase in an evenly spaced pattern all the way up to the extreme case.

1. $E_k(\boldsymbol{x_0}) = 10\,\text{J}, 20\,\text{J}$: The robot is able to dampen its momentum with inertial shaping, so its contact transition tree has only a single node (Fig. 3.4).

2. $E_k(\boldsymbol{x_0}) = 30\,\text{J}, 40\,\text{J}, 50\,\text{J}$: Inertial shaping cannot stabilize the robot, and protective stepping is needed (Fig. 3.5).

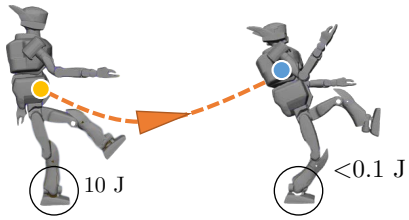


FIGURE 3.4: Inertial shaping with initial kinetic energy 10 J.

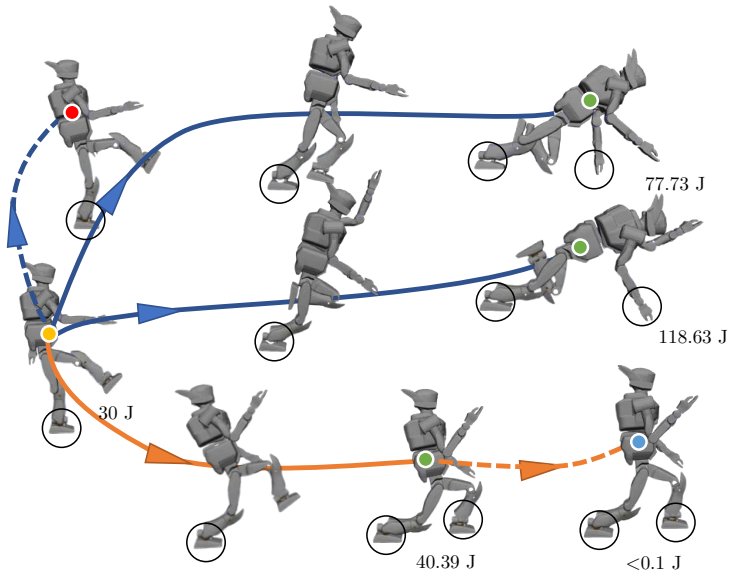


FIGURE 3.5: Protective stepping with initial kinetic energy 30 J.

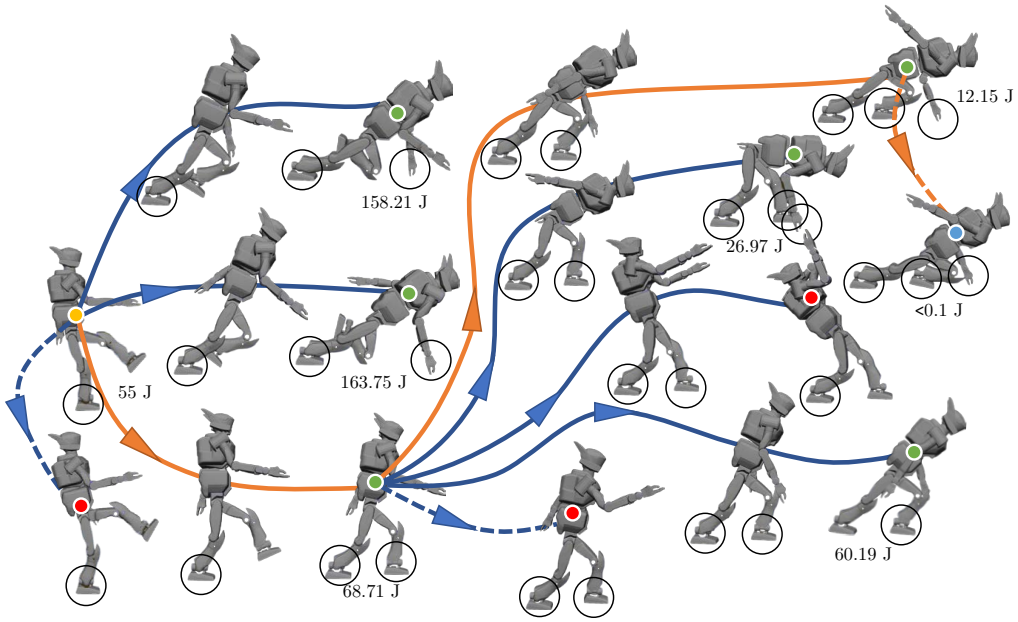


FIGURE 3.6: Protective stepping, hand contact and inertial shaping with initial kinetic energy 55 J.

3. $E_k(\boldsymbol{x_0}) = 60\,\text{J}, 70\,\text{J}, 80\,\text{J}$: Neither inertial shaping nor protective stepping are sufficient. For these cases, our algorithm explores until depth 2, where hand contact enables successful stabilization. The contact transition trees for these cases are similar to Fig. 3.6.

Fig. 3.7 lists the kinetic energy trajectories for all eight cases. All KE at ending time have been reduced to zero.

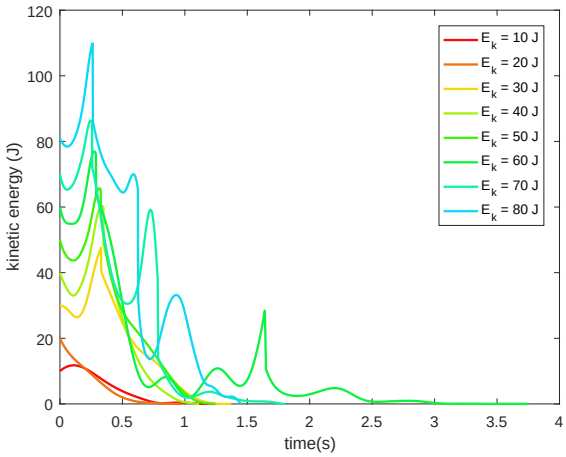


FIGURE 3.7: Kinetic energy trajectories with initial KE varying from 10 J to 80 J

In addition, we further test the proposed algorithm in one extreme case where the initial kinetic energy is greater than 150 J. When the initial $E_k(\boldsymbol{x_0})$ is extremely large, the planning of optimal contact sequence will fail to plan the stabilization strategies. This failure is due to constraints on the joint torque bounds and contact force feasibility. When the joint torque is not larger enough to maintain the contact holonomic constraints on position and velocity, the supportive normal contact force will need to be negative to drag the contact point on the contact surface. This negative contact force violates its feasibility constraints, thus preventing the optimal solution being computed.

3.5 Summary

This chapter introduces a generalized motion planner for humanoid fall recovery which existing strategies, such as inertial shaping, protective stepping, and hand contact strategies, will be automatically synthesized for fall stabilization. The planner optimizes both the contact sequence and the robot state trajectories using a contact transition tree search. A greedy minimization of kinetic energy tends to find solutions with few contact changes and very little backtracking. An efficient method to generate initial seeds for trajectory optimization facilitates convergence. Experiments demonstrate show that our proposed algorithm can generate complex stabilization strategies for a simulated humanoid under varying initial pushes and environment shapes.

Despite the fact that each self or transition optimization takes approximate $3 \sim 5$ min provided promising initial seeds, much more training data can still be gathered. Then we would like to resort to a machine learning approach to learn the control policy which should be generalized enough for arbitrary fall mitigation.

4

Online Calibration for Autonomous Vehicle's Longitudinal Dynamical System

This chapter presents an efficient online calibration system for longitudinal vehicle dynamics of driverless cars. Instead of modeling vehicle's longitudinal dynamical system analytically, a data-driven method is employed to generate an "end-to-end" numerical model with a look-up table which saves vehicle's velocity, control command, and acceleration. This reference table should be calibrated to account for variations of vehicle's hardware status over time. To reduce the expensive labor in calibration process, an effective algorithm is proposed to update this reference look-up table with a Gaussian model approach. A 2-D Gaussian distribution is introduced to model acceleration error between interpolated one from look-up table and actual one from vehicle sensors.Gaussian model's standard deviations is estimated with a "three-sigma rule" heuristic and its height is calculated with a backtracking method such that monotonicity constraint between acceleration and control command is strictly satisfied in the updated table. The effectiveness of the proposed

system is verified in real-world road tests with Lincoln MKZ [1].

4.1 Introduction

Autonomous vehicles have received ever-increasing attention in recent years because of their potentials to decrease traffic congestion and fatal accidents, improve productivity and efficiency of the driving time, and serve as transportation tools accessible to anyone at anytime [SAMR18]. These expectations correspond to level 4 or higher driving automation from Society of Automotive Engineers (SAE) and many companies and research institutes have been studying and developing strategies to elevate levels of autonomy of existing vehicles [SAE18]. A fully autonomous vehicle should typically be equipped with two main sub-systems: (1). perception system for state estimation, obstacle detection and traffic-sign recognition (2). decision-making system for motion planning, path selection and vehicle control in lateral and longitudinal directions [PCY+16, BGC+19]. These intelligent systems enable self-driving cars to navigate themselves in complex environments with static and moving obstacles, and reason in a human-level reliability to deal with unpredictable traffic situations. Though promising results have been demonstrated in recent works [FSRea13, TKZS16, URR+17, FCAF17], the ability to generalize to complex traffic scenarios with many neighboring vehicles are yet to be achieved. Among all the challenges remained to be solved, this paper contributes an effective algorithm to vehicle's decision-making system by increasing the low-level control accuracy of vehicle's longitudinal dynamical system.

Given a standard hierarchical planning and control framework, a reference state trajectory $x_{\text{ref}}(t)$ is calculated after planning for route, path, and vehicle motion. This

[1]This chapter is reproduced from Shihao Wang, Canqiang Deng, Qingjie Qi and Tongyi Cao, "Efficient Online Calibration for Autonomous Vehicle's Longitudinal Dynamical System: A Gaussian Model Approach", submitted, *IEEE International Conference on Robotics and Automation*, Oct 2020

reference denotes vehicle's desired state in time horizon and stabilizing controller will be synthesized for trajectory tracking. Existing approaches have used feedback linearization [DLOS98], model predictive control [BGR04, FBA+07], nonlinear control [HTMT07], feedback-feedforward control [KG15] to regulate vehicle's motion to its reference path. However, the gap between the kinematic or dynamic model used in these methods and vehicle's actual dynamical system makes it difficult to accurately convert high-level control signal into low-level control command (throttle/brake). Furthermore, the fact that conditions in both vehicle and road vary over time requires frequent system identification and calibration to be conducted where each task takes considerable manual labour and time [SRMMK13, DPP15]. To facilitate the conversion from high-level control policy to low-level throttle/brake command and maintain an efficient lifelong model calibration without valuable human labor, we first employ an offline data-driven approach to implicitly model vehicle's longitudinal dynamical system with a look-up table. This table connects vehicle's velocity, control command to its corresponding acceleration and enables a bijective mapping between control command and acceleration under fixed speed. Then an online calibration algorithm is proposed to update this reference table according to vehicle's actual information at runtime. We model as a 2-D Gaussian model acceleration error between interpolated one from look-up table and actual one from vehicle's on-board sensors. Model's standard deviations are estimated with a "three-sigma rule" and its height is calculated with a backtracking method such that monotonicity constraint between control command and acceleration remains satisfied in the updated look-up table. By incorporating this calibration algorithm on-the-fly, autonomous vehicle is able to adjust its reference look-up table using its actual dynamical information and experiments in real-world road-tests demonstrate that our proposed online calibration module effectively decreases vehicle's longitudinal position error by 40% after few laps on test road.

4.2 Related Work

Our proposed system focuses vehicle system identification and calibration. Many related strategies have been proposed to address these topics.

Vehicle model plays an essential role in decision-making system of self-driving cars and model accuracy dramatically influences the performance of vehicle's control system. While kinematic model is generally adopted for vehicle in relatively low-speed scenarios, dynamic model should be considered to describe vehicle motions at high-speed [YWZ+09, YLL13]. Depending on the principles used to identify vehicle's dynamical system, existing approaches can be categorized into two classes : *physics-based modeling* and *data-driven-based modeling*. The first class of modeling methods embraces classical mechanics as their fundamental rules and derives explicit equations of motion of vehicle's translational and rotational dynamics with Newton–Euler equations [Raj06, Jaz09, Pac12]. This type of models captures vehicle's dynamical behaviors with deterministic parameters and provides convenience to model-based controller design and vehicle simulation due to their closed-form expressions. Even though these modeling techniques have obvious advantages resulting from their simplicity and conciseness, they suffer from the difficulty in determining values of physical parameters. A full dynamics model requires a number of parameters to be predetermined to take into consideration of tire-road contact, vehicle powertrain, aerodynamics, engine dynamics, etc. However, some of them are not directly measurable and the estimation of their values needs introducing additional nonlinear regression models, which makes the symbolic dynamics expression no longer simple [Ray95, JRA02, Bes10]. To circumvent the above limitations, researchers have proposed data-driven-based approaches for system modeling and identification of automated vehicles [SSA08]. Strategies such as extended Kalman-filter method [LW07], prediction error method [AUKC08], predictor-based subspace identification [GBZ15],

feed-forward neural networks [SBK+19], and deep-learning based on Koopman operator method [XZX+20] have demonstrated their powerful abilities in approximating highly nonlinear system dynamics. However, their poor interpretability, difficulty in generalization and unknown model sensitivity make them challenging to be employed as industry-level approaches. To address these issues, we propose a specialized data-driven method to generate an "end-to-end" vehicle model with an interpretable look-up table. This table saves vehicle's velocity, control command, and acceleration and can be visualized as a set of 3-D points which can be straightforwardly explained.

The non-deterministic characteristics of vehicle's parameters suggest model calibration to be conducted for error correction. To decrease the expensive human labor in vehicle calibration process, online model adjustment techniques have been proposed to address several relevant aspects. Tafner presents a robust parameter estimation method for a simplified roll dynamics with sliding mode concepts [TRH14]. Seegmiller proposes an automatic calibration method to model the perturbative vehicle behaviors on arbitrary terrains [SRMMK11]. The most relevant work to our online calibration method is Baidu Apollo's auto-calibration system where an offline look-up table is initially generated with a three-layer feedforward neural network and then this whole reference table is adjusted with a customized damping approach [ZMX+18]. Though sharing a similar system structure, our technique has the following advantages:

- **calibration region**: Instead of a "global" calibration, our method divides the reference look-up table into two sub-regions to separate accelerations effected by throttle and brake, and adaptively calibrates a "local" area of the sub-region based on a customized percentage parameter. This transition from "global" to "local" significantly reduces the computational time and makes our approach sufficient for real-time implementation.

- **updated rule**: Apollo introduces a customized similarity evaluation function to calculate the updated value of every point in look-up table where a number of parameters should be carefully tuned. However, our method adopts a 2-D Gaussian model to describe error distribution of acceleration with only few parameters to be determined.

- **constraint satisfaction**: Our method adopts a backtracking strategy to iteratively search feasible solutions for constraint satisfaction while Apollo's approach lacks the ability to deal with constraint violation.

4.3 Offline Look-up Table Generation

We adopt a data-driven-based approach to build an "end-to-end" numerical model of vehicle's longitudinal dynamical system. This numerical model is constructed into a 3-D look-up table whose dimensionalities are vehicle's control command, speed, and acceleration. The reason behind this choice is twofold: (1). Among all the variables that influence vehicle's acceleration, control command and velocity are no doubt the dominant ones [Raj06]. (2). By limiting the number of features to be 2, a mapping function $f(c, v) \rightarrow a$ will be constructed where c, v and a denote vehicle's longitudinal control command, speed, and acceleration, respectively. Enumerations along c and v generate a 3-D surface whose shape is intuitive and interpretable (Please refer to Fig. 4.6b).

Procedures for look-up table construction are as follows:

1. Feasible range estimation of control command and speed

    - Command range is estimated with two steps: Firstly, vehicle's control command range is normalized to be $c \in (-1, 1)$ where $c > 0$ and $c < 0$ stand for throttle and brake, respectively. Secondly, throttle and brake's

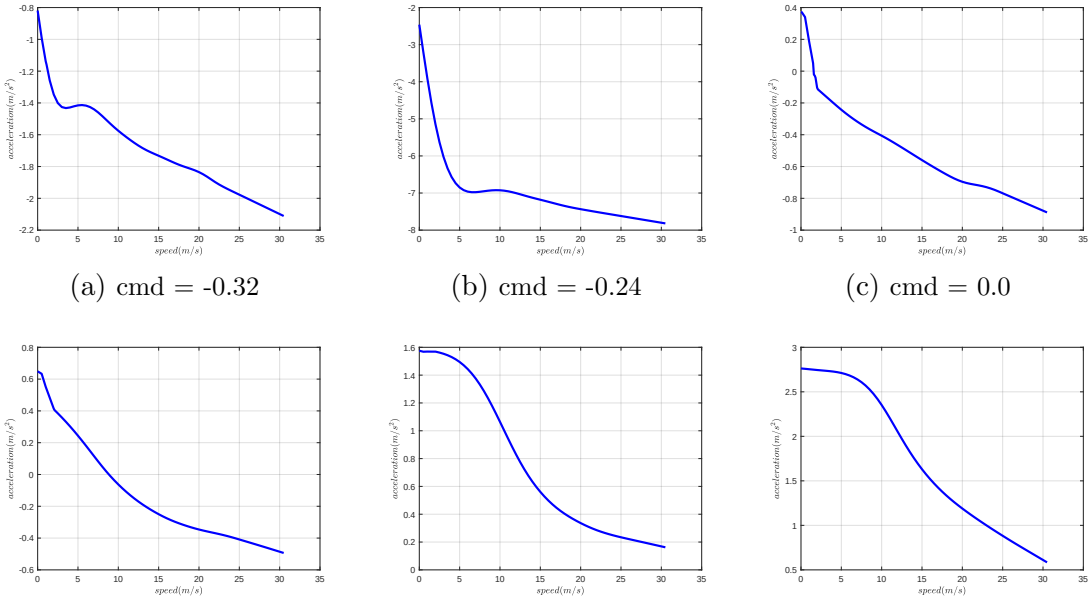
deadzones are measured by human engineers and command's feasible range is then divided to be $c \in (-1, c_{\text{brake}}) \cup (c_{\text{throttle}}, 1)$ where vehicle is non-responsive to applied pedal if $c \in [c_{\text{brake}}, c_{\text{throttle}}]$.

- Speed range is estimated according to vehicle's speed limit from its usual work zone $v \in [0, v_{\max}]$.

2. Data collection of vehicle's speed and acceleration

   After a uniform-grid discretization of command range to be of size N_{brake} and N_{throttle} separately, we collect data of vehicle's velocity and acceleration trajectories under constant control command:

   - For sampled $c \in (-1, c_{\text{brake}})$, we first accelerate vehicle's speed to be $v_{\max}$ and then switch control command to be c to record v and a until vehicle stops.



(a) cmd = -0.32 (b) cmd = -0.24 (c) cmd = 0.0

(d) cmd = 0.25 (e) cmd = 0.5 (f) cmd = 0.75

FIGURE 4.1: 2-D slices of 3-D look-up table along **command** dimension where x axis is speed and y axis is acceleration. Negative and positive cmd denote brake and throttle, respectively and zero cmd (Fig. 4.1c) is the behavior of vehicle's natural dynamics.

- For sampled $c \in (c_{\text{throttle}}, 1)$, vehicle is initially set to be static and then accelerates with c until $v \geqslant v_{\max}$.

3. Acceleration correction with pitch angle

   Due to the existence of unevenness of the ground, vehicle's pitch angle θ will vary during the data collection process. This pitch angle adds ground projection of gravitational acceleration to longitudinal acceleration and its effect should be cancelled from the measured acceleration: $a = a^* - g\sin(\theta)$ where a^* is the measured acceleration and $g = 9.81m/s^2$. Illustrative $v - a$ trajectories are presented in Fig. 4.1.

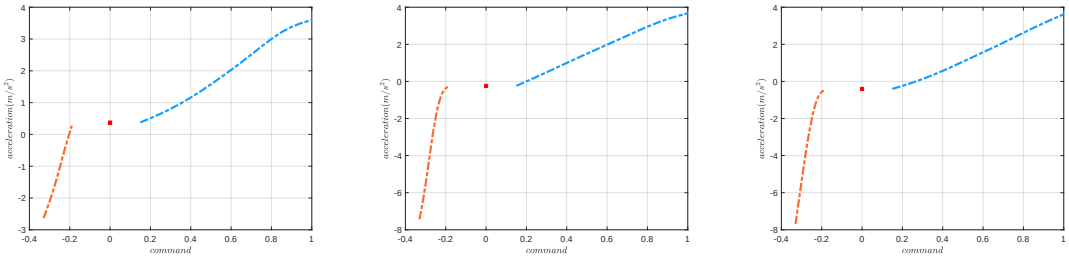
4. Reference table generation with slicing velocities

   After collecting vehicle's velocity and corrected acceleration trajectories under fixed control commands, we evenly discretize speed range to be N_{speed} points and generate reference look-up table by slicing trajectories along speed grid. Fig. 4.2 illustrates several velocity slices from look-up table where a strictly monotonic relationship is satisfied between control command and acceleration.

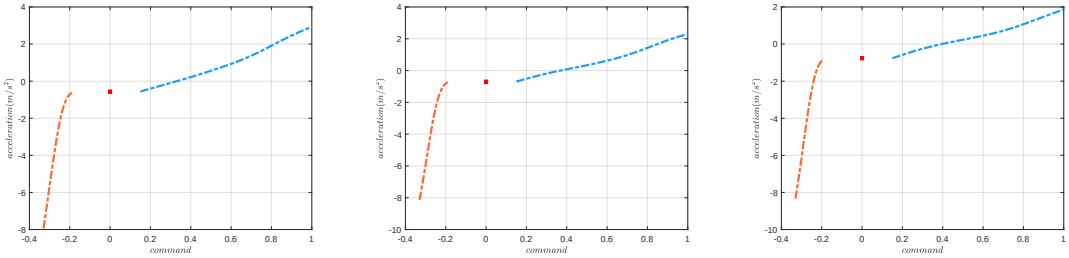
This monotonic relationship between c and a allows a bijective mapping between low-level control command and acceleration and this property makes our look-up table approach suitable for both offline simulation $f(c, v) \rightarrow a$ and online control $f^{-1}(a, v) \rightarrow c$.

4.4 Online Look-up Table Calibration

The reference look-up table describes vehicle's dynamical relationships between its speed, control command and acceleration and should be calibrated to account for variations of vehicle conditions, such as changes in car engine's performance or different vehicles with similar specifications. With actual measurements available from



(a) speed = $0m/s$ (b) speed = $5m/s$ (c) speed = $10m/s$



(d) speed = $15m/s$ (e) speed = $20m/s$ (f) speed = $25m/s$

FIGURE 4.2: 2-D slices of 3-D look-up table along **speed** dimension where x and y denote control command and acceleration, respectively. Red curve is brake (cmd < 0) and blue curve (cmd > 0) is throttle. The gap between these two curves is the effect from deadzones whose acceleration is marked with red dot (cmd $= 0$).

car's on-board sensors, this reference table should be adjusted to minimize acceleration error between table's interpolated value and actual value from sensor reading. For this purpose, we introduce a 2-D Gaussian model to describe distribution of acceleration error on $c \times v$ plane and adopt a backtracking method to calculate model's height A such that acceleration error is reduced and the monotonic constraint is satisfied in the updated table. Note that we employ a "three-sigma-rule" to calculate model's standard deviations σ_c and σ_v in a local region whose size is chosen by area percentage $\gamma \in (0, 1]$.

We define a list of notations to facilitate the following discussion of the proposed online calibration algorithm:

- round to nearest integer operator $[\cdot] : \mathbb{R} \to \mathbb{Z}$
- $c(\cdot) : \mathbb{Z} \to \mathbb{R}$ and $v(\cdot) : \mathbb{Z} \to \mathbb{R}$ evaluate control command and speed at certain

Algorithm 2: Online Calibration Algorithm

Input : Reference look-up table $f(\cdot,\cdot)$, actual measurement $\boldsymbol{x}^* = (c^*, v^*, a^*)$, area percentage γ, learning rate η

Output: Updated look-up table $f^*(\cdot,\cdot)$

1 **if** $c^* < 0$ **then**
2 | $n_{\text{cmd}} \leftarrow [\gamma N_{\text{brake}}]$, c's range is $(-1, c_{\text{brake}})$
3 **else**
4 | $n_{\text{cmd}} \leftarrow [\gamma N_{\text{throttle}}]$, c's range is $(c_{\text{throttle}}, 1)$
5 **end**
6 $n_{\text{speed}} \leftarrow [\gamma N_{\text{speed}}]$, v's range is $[0, v_{\max}]$
7 $f_{\text{local}}(\cdot,\cdot)$, $\mathbb{C}$, $\mathbb{V} \leftarrow$ **updated_region_sel**$(n_{\text{cmd}}, n_{\text{speed}}, c^*, v^*, f(\cdot,\cdot))$
8 $A \leftarrow \eta \cdot (a^* - f(c^*, v^*))$
9 $\sigma_c, \sigma_v \leftarrow$ **standard_deviation_cal**$(\mathbb{C}, \mathbb{V}, c^*, v^*, f(\cdot,\cdot))$
10 $A^* \leftarrow$ **backtracking**$(A, \sigma_c, \sigma_v, \mathbb{C}, \mathbb{V}, f_{\text{local}}(\cdot,\cdot))$
11 **if** $A^* \neq 0$ **then**
12 | $f^*(\cdot,\cdot) \leftarrow$ **table_update**$(A^*, \sigma_c, \sigma_v, \mathbb{C}, \mathbb{V}, f(\cdot,\cdot))$
13 | **return** $f^*(\cdot,\cdot)$
14 **end**
15 **return** $f(\cdot,\cdot)$

grid index, respectively.

- $f(\cdot,\cdot)$: look-up table struct whose members include c_{brake}, c_{throttle}, $v_{\max}$, N_{brake}, N_{throttle} and N_{speed}, and methods contain $c(\cdot)$ and $v(\cdot)$.

- Within $c \times v$ region to be updated, set of control command indices is denoted as $\mathbb{C}$ and set of speed indices set is denoted as $\mathbb{V}$.

Alg. 2 illustrates the details of calibration algoirthm:

- Lines 1–6 distinguishes control command from actual measurement to be **throttle** or **brake** and computes sizes of command grid and speed grid to be updated according to area percentage parameter γ. The introduction of γ allows a local region of trust to be customized where a bigger γ updates a larger area which takes longer computational time while a smaller γ takes less elements for value adjustment but can cause table's overall geometric shape to be not smooth resulting from local "bump"s.

Algorithm 3: Standard Deviation Calculation

Input : Sets of command index $\mathbb{C}$ and speed index $\mathbb{V}$, actual command c^* and speed v^*, and reference look-up table $f(\cdot, \cdot)$

Output: Standard deviations σ_c and σ_v

1 $d_c \leftarrow 0, d_v \leftarrow 0$
2 **for** $i \in \mathbb{C}$ and $j \in \mathbb{V}$ **do**
3 $\quad c \leftarrow c(i)$, and $d_c \leftarrow |c - c^*|$ if $d_c \leqslant |c - c^*|$
4 $\quad v \leftarrow v(j)$, and $d_v \leftarrow |v - v^*|$ if $d_v \leqslant |v - v^*|$
5 **end**
6 $\sigma_c \leftarrow \frac{d_c}{3}, \sigma_v \leftarrow \frac{d_v}{3}$
7 **return** σ_c, σ_v

- Line 7 adaptively selects the region to be updated from actual measurement and grid number n_{cmd} and n_{speed}. The center of this rectangle is located at (c^*, v^*) and its width and length in index level should be n_{cmd} and n_{speed}. We denote this local table as $f_{\text{local}}(\cdot, \cdot)$ and it occupies a closed rectangular space on $c \times v$ plane. This rectangular area will not always be in the interior of feasible ranges of control command and speed, and its width or length must be truncated to match the overlapped region if it is noninclusive to feasible ranges of command and speed.

- Line 8 initializes Gaussian model's height A with a scaled acceleration difference between actual acceleration a^* and interpolated acceleration $f(c^*, v^*)$.

- Line 9 calculates standard deviations used for 2-D Gaussian model with "three-sigma-rule" heuristic [Puk94]. Details of this subroutine are illustrated in Alg. 3.

- Line 10 conducts backtracking to calculate 2-D Gaussian model's height A^* given its initial value A and standard deviations σ_c and σ_v. We denote Gaussian function's dependence on height in superscript and express its formula to be

$$G^A(c, v) = A \exp(-(\frac{(c - c^*)^2}{2\sigma_c^2} + \frac{(v - v^*)^2}{2\sigma_v^2})) \tag{4.1}$$

where $c = c(i), v = v(j), \forall i \in \mathbb{C}$ and $\forall j \in \mathbb{V}$. To assert that acceleration remains a monotonic function of control command after adding this Gaussian offset function to $f_{\text{local}}(\cdot,\cdot)$, an iterative procedure will be conducted to backtrack A's value by multiplying a scale coefficient $s \in (0,1)$. With the increment of iterations, Gaussian model's height will decrease exponentially. Since original $f_{\text{local}}(\cdot,\cdot)$ is monotonic between control command and acceleration, our algorithm is guaranteed to produce a feasible updated table (Alg. 4).

- Line 11-14 calibrates original look-up table by adding this Gaussian distribution to the local area of $f(\cdot,\cdot)$ where **table_update**'s detail is similar to Alg. 4's Line 4-10.

Algorithm 4: Backtracking Algorithm

Input : Initial height A, standard deviations σ_c and σ_v, sets of command index $\mathbb{C}$ and speed index $\mathbb{V}$ and local look-up table $f_{\text{local}}(\cdot,\cdot)$

Output: Calibrated model height A^*

1 Iter $\leftarrow 0, A^* \leftarrow A$
2 **while** Iter $<$ IterMax **do**
3 $\quad f_{\text{local}}^{\text{temp}}(\cdot,\cdot) \leftarrow f_{\text{local}}(\cdot,\cdot)$
4 $\quad$ **for** $i \in \mathbb{C}$ **do**
5 $\quad\quad c \leftarrow c(i)$
6 $\quad\quad$ **for** $j \in \mathbb{V}$ **do**
7 $\quad\quad\quad v \leftarrow v(j)$
8 $\quad\quad\quad f_{\text{local}}^{\text{temp}}(c,v)$ += $G^{A^*}(c,v)$ (4.1)
9 $\quad\quad$ **end**
10 $\quad$ **end**
11 $\quad$ **if** $f_{\text{local}}^{\text{temp}}(\cdot,j)$ is entry-wise monotonic $\forall j \in \mathbb{V}$ **then**
12 $\quad\quad$ **return** A^*
13 $\quad$ **else**
14 $\quad\quad A^* \leftarrow s \cdot A^*$
15 $\quad\quad$ Iter += 1
16 $\quad$ **end**
17 **end**
18 **return** 0

4.5 Sensor Data Processing

Our calibration algorithm (Alg. 2) takes vehicle's runtime measurement $\boldsymbol{x}^* = (c^*, v^*, a^*)$ as nominal value and adapts reference look-up table in compliance with differences between a^* and $f(c^*, v^*)$. $\boldsymbol{x}^*$ has 3 elements and their values can be easily measured from car sensors. However, these raw sensor readings are not directly applicable for real-time calibration because of sensor noise and signal's time delay. As a result, sensor data processing is applied for data refinement before using x^* for table calibration.

4.5.1 Noise Filtering

Control command c^*, speed v^* and longitudinal acceleration a^* are measured from controller area network (CAN bus), global navigation satellite system (GNSS), and inertial measurement unit (IMU), respectively. To filter the high frequency noise from these measurements, low-pass butterworth filter is implemented for signal smoothing. We denote filter's order with $\boldsymbol{n}$ and cut-off frequency with $\boldsymbol{f}$, and introduce filters for throttle, brake, speed and acceleration whose specifications are $(\boldsymbol{n}_{\text{throttle}}, \boldsymbol{f}_{\text{throttle}})$, $(\boldsymbol{n}_{\text{brake}}, \boldsymbol{f}_{\text{brake}})$, $(\boldsymbol{n}_{\text{speed}}, \boldsymbol{f}_{\text{speed}})$, $(\boldsymbol{n}_{\text{acc}}, \boldsymbol{f}_{\text{acc}})$ and $(\boldsymbol{n}_{\text{pitch}}, \boldsymbol{f}_{\text{pitch}})$. One existing issue of signal processing with non-predictive filters is "phase shift" where a fixed period of time is needed to determine what filtered signal should be. A practical strategy to resolve this issue is to first store these trajectories for a certain amount of time and then conduct filtering in both forward and backward directions to result in zero-phase shift trajectories.

4.5.2 Time Shifting

We define response delay δ_{throttle} and δ_{brake} to be the time that vehicle takes to accelerate or decelerate in response to driver's control command in throttle or brake [AK06]. This delay causes misalignment between control command and acceleration

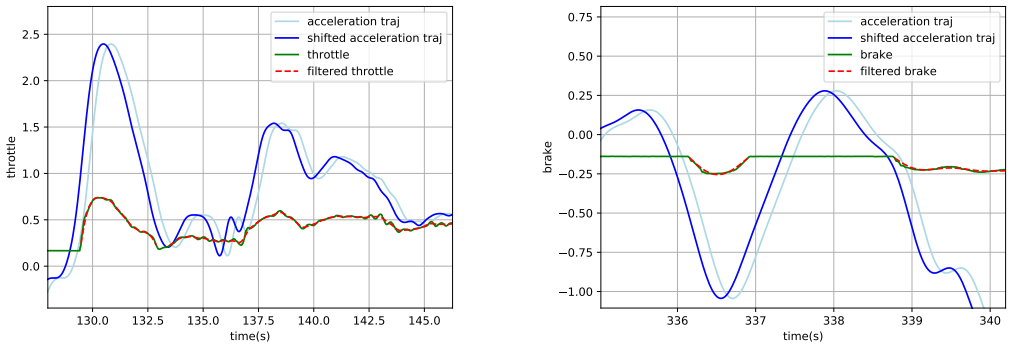
because acceleration trajectory should be shifted forward in time to match its corresponding control command. To calculate this response delay, a cross-correlation problem is formulated. Take throttle for example:

$$\max_{\delta_{\text{throttle}}} ||a(t+\delta_{\text{throttle}}) \cdot c_{\text{throttle}}(t)||_2^2$$
$$0 \leqslant \delta_{\text{throttle}} \leqslant \Delta \tag{4.2}$$

where $a(t)$ and $c_{\text{throttle}}(t)$ are prerecorded trajectories and Δ is an upper bound of time delay. Since response delay is vehicle's inherent characteristic and can be considered to be time-invariant, this allows us to pre-compute it offline and then use it for online time shifting. By discretizing δ_{throttle}'s range into a number of points, a brute force method is used to find the argument of the maximum. Visualizations of time shifting in acceleration trajectories for throttle and brake are illustrated in Fig. 4.3.

4.6 Examples

We evaluate the effectiveness and robustness of the proposed method with two main sets of experiments. The first set of experimentation shows that our method ef-



(a) (b)

FIGURE 4.3: Plots of response delay in throttle (a) and brake (b) where acceleration trajectories are shifted forward to align with command trajectories. It is obvious that this time adjustment improves cross-correlation between acceleration and command trajectories.

Table 4.1: Experimental Parameters

Algorithm Parameters				Filter Coeffs			
c_{throttle}	0.15	N_{throttle}	18	$\boldsymbol{n}_{\text{throttle}}$	2	$\boldsymbol{f}_{\text{throttle}}$	10 Hz
c_{brake}	-0.19	N_{brake}	15	$\boldsymbol{n}_{\text{brake}}$	3	$\boldsymbol{f}_{\text{brake}}$	10 Hz
v_{max}	30.5 m/s	N_{speed}	306	$\boldsymbol{n}_{\text{speed}}$	2	$\boldsymbol{f}_{\text{speed}}$	25 Hz
γ	0.5	η	0.001	$\boldsymbol{n}_{\text{acc}}$	2	$\boldsymbol{f}_{\text{acc}}$	25 Hz
δ_{throttle}	0.35 s	δ_{brake}	0.15 s	$\boldsymbol{n}_{\text{pitch}}$	2	$\boldsymbol{f}_{\text{pitch}}$	10 Hz
IterMax	25	s	0.1				

fectively increases longitudinal control accuracy with real-world road tests and the second sets of experiment demonstrates that our calibration algorithm robustly reduces acceleration discrepancies even though intentionally incorrect reference tables are given. We collect vehicle data with Lincoln MKZ vehicle and implement our approach on a 64-bit Intel 12-Core i7 2.60GHz laptop with 16GB RAM. Experimental parameters are listed in Tab. 4.1 and online calibration procedure takes less than 0.5 ms.

4.6.1 Longitudinal Control Accuracy Evaluation

This subsection demonstrates the capability of our proposed method to effectively improve vehicle's longitudinal control accuracy. Given road test path as shown in

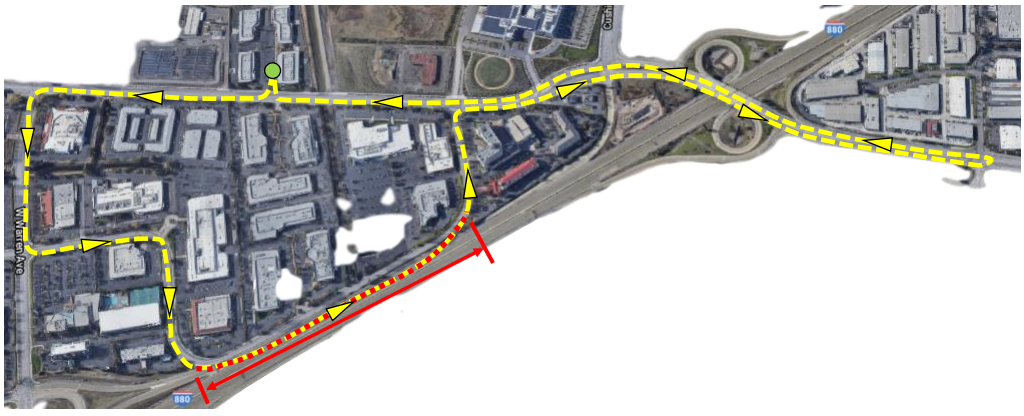


FIGURE 4.4: Representative road-test path which includes traffic lights, stop signs, sharp turns, U-turn, unprotected turn, and highway. Red line denotes the road where position error comparison is conducted. Picture source: Map data ©2020 Google

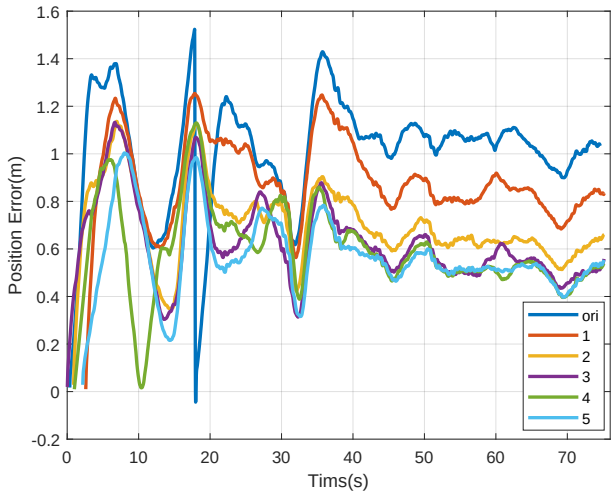


FIGURE 4.5: Trajectories of longitudinal position errors from road tests where their origins are shifted manually to ensure the alignment of geometric shapes.

Fig. 4.4 and a reference look-up table whose calibration was conducted several months ago, we evaluate the improvement of control accuracy with analyzing vehicle's longitudinal position error. This position error $e_p(t) = p^*(t) - p(t)$ is measured from a comparison between vehicle's planned position $p^*(t)$ with its current position $p(t)$.

Firstly, original position error trajectory is collected along the whole path with the given reference look-up table. Then, we activate the online calibration module and save position error trajectories for a total of 5 laps around this path. We compare position error along a segment (red line in Fig. 4.4 whose traffic situations remain steady during these 6 continuous tests. Fig. 4.5 illustrates trajectories of longitudinal position errors from our road tests and its performance analysis in terms of average position error (APE) and error percentage decrease (EPD) is shown in Tab. 4.2.

Table 4.2: Longitudinal Position Error Results

	original	1	2	3	4	5
APE(m)	1.013	0.891	0.703	0.624	0.598	0.581
EPD(%)		12.0	30.6	38.4	40.9	42.6

It is obvious that vehicle's longitudinal error decreases consistently until it reaches a lower bound with our online calibration module and a more than 40% accuracy improvement is achieved after 5 laps' calibration.

4.6.2 Reference Table Acceleration Error Evaluation

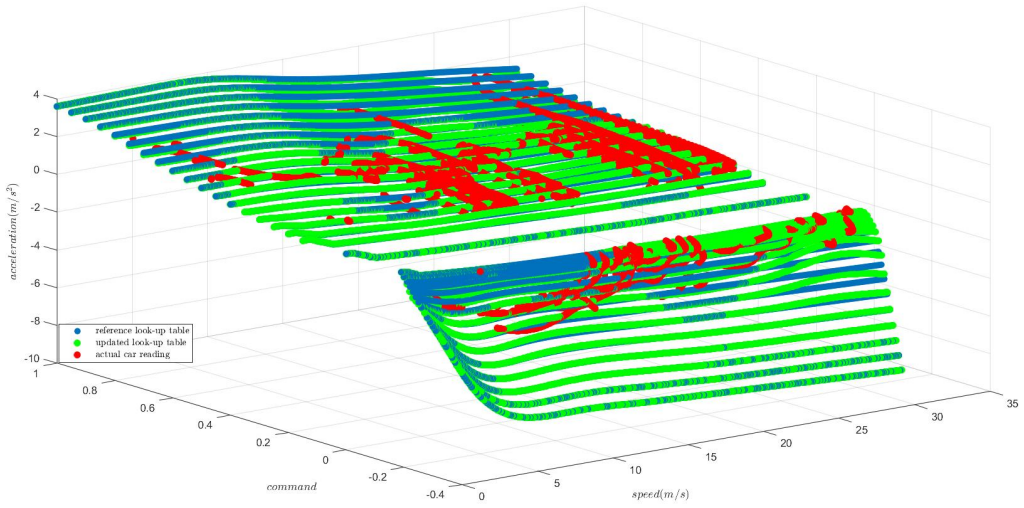
In addition to online adjustment, our method can also be adopted in an "offline" fashion. Given vehicle's data collected from actual road tests, our proposed algorithm robustly minimizes acceleration errors even though intentionally incorrect reference tables are given

Given 10 road test data and 3 reference look-up tables, we compare results of without/with calibration algorithm. Each road test data is collected along paths similar to Fig. 4.4. Among these 3 tables, Table 1 is a standard reference look-up table and we then intentionally add $+/-1.0\,m/s^2$ to Table 1's acceleration dimension to generated offsetted Table 2 and Table 3.

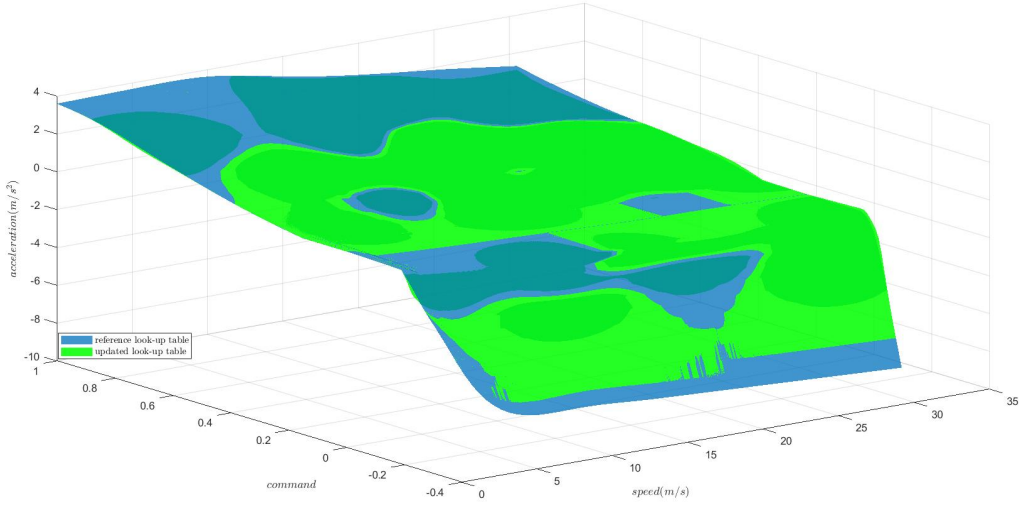
For each road-test data, we randomly allocate 75% for calibration and 25% for test, and quantitative results using mean absolute error (MAE) are illustrated in

Table 4.3: Reference Table Acceleration Error Results

Case	Table1	Table1*	Table2	Table2*	Table3	Table3*
1	0.112	0.086	0.933	0.610	1.085	0.608
2	0.127	0.084	0.925	0.495	1.119	0.807
3	0.107	0.091	0.932	0.477	1.063	0.698
4	0.114	0.093	0.960	0.611	1.056	0.586
5	0.133	0.080	0.922	0.546	1.096	0.643
6	0.123	0.112	0.911	0.701	1.095	0.698
7	0.117	0.087	0.953	0.610	1.081	0.564
8	0.120	0.075	0.945	0.572	1.052	0.869
9	0.112	0.098	0.968	0.641	1.025	0.736
10	0.144	0.132	0.966	0.584	1.000	0.635
Avg	0.119	0.093	0.942	0.585	1.067	0.684
EPD(%)		21.8		37.9		35.9



(a) Visualization of reference look-up table (blue dots), actual car readings (red dots) and look-up table after online calibration (green dots).



(b) Visualization of reference look-up table (blue surface) and updated look-up table (green surface).

FIGURE 4.6: A representative comparison between reference look-up table and updated look-up table after calibration. Here, x, y axes are speed and command, respectively and z axis is acceleration. Given vehicle's actual readings in red dots, our algorithm adjusts table's "shape" to reduce acceleration discrepancies, thus increasing vehicle's longitudinal control accuracy.

Tab. 4.3 where the superscript $*$ denotes the calibrated table and value's unit is m/s^2. The result shows that for all 10 test cases, our method is capable of minimizing acceleration errors given different reference tables and these 2 intentionally offsetted reference table are robustly adjusted with $> 35\%$ improvement in matching actual accelerations.

4.7 Summary

This paper presents an efficient online calibration system for longitudinal vehicle dynamics of autonomous vehicles. We employ a data-driven approach to generate an "end-to-end" numerical vehicle model with a look-up table which saves vehicle's velocity, control command, and acceleration. This reference table helps connect high-level control objective to vehicle's low-level throttle or brake command and should be calibrated to account for variations of vehicle's hardware status over time. For the sake of reducing expensive labor in manual calibration process, we propose an effective online calibration algorithm to update this reference look-up table with a Gaussian model approach. We introduce a 2-D Gaussian distribution to model acceleration errors between interpolated one from look-up table and actual vehicle acceleration. This model's standard deviations are estimated with a "three-sigma rule" heuristic and its height is calculated with a backtracking method to guarantee monotonicity constraint between acceleration and control command. We demonstrate the effectiveness and robustness of our proposed method with two sets of experiments where sufficient performance improvement is achieved using vehicle data collected from Lincoln MKZ.

During the implementation of the proposed algorithm for real-world road tests, we have observed that response delay plays a significant role in calibrating vehicle's acceleration to its corresponding control command. Currently, we employ an offline approach to compute vehicle's delay time (described in Sec. 4.5.2). We plan in the near future to implement an online algorithm to calculate these delays from throttle and brake.

5

Conclusions

This dissertation presents two optimization-based motion planners for humanoid fall recovery. The first motion planner is an online approach which aims to recover humanoid from falling by making hand contact with walls or other surfaces in the cluttered environment. This planner simplifies the difficulty in solving a general multi-modal planning problem (Sec.1.3.3) with the following techniques:

- Contact mode sequence is pre-specified to be: Foot Contact + Foot Contact + Hand Contact. Thus, the computational effort to explore other possible contact mode sequences is saved.
- Full-body humanoid model is approximated with a low-dimensional three-link model to reduce state dimensionality.
- A direct single shooting method is used to handle the system dynamics constraints, thus resulting in a small number of optimization variable to be solved.
- Pre-computation of computationally expensive terms is conducted to save online computation time.

With these techniques, this planner can efficiently plan a hand contact reference in a standard PC with less than $0.1\,s$. This approach extends humanoid's balancing capability to cluttered environment with making hand contact and the idea to make contact with environmental objects provides a novel solution for fall recovery in cluttered environment. The second motion planner aims to generalize humanoid fall recovery to both open and cluttered environment with simultaneously optimizing discrete contact mode sequence and continuous robot trajectories. This planner directly addresses the general multi-modal planning problem (Eqn. 1.3.3) with a hybrid planning scheme where a high-level tree-search algorithm is adopted for exploring contact mode sequence and a low-level trajectory optimization is solved with direct collocation method for state transition and robot stabilization. This approach unifies existing recovery strategies, such as inertial shaping, protective stepping, and hand contact, and automatically plans one strategy or a combination of strategies to regain robot's balance based on its disturbed state and nearby environment features. By enabling humanoid to reason how to regain balance on its own, this algorithm makes a significant contribution to the improvement of humanoid's sustainability in arbitrary environment.

5.1 Summary of Thesis Work

My contributions are as follows,

- I develop a "real-time" humanoid fall recovery motion planning algorithm which enables robot to make use of hand contact for fall protection in cluttered environment. This planning algorithm efficiently solves a nonlinear programming problem where several techniques are proposed to achieved fast computational speed. I implement this algorithm in a standard PC and this method outperforms state-of-the-art optimization-based fall recovery method by orders

of magnitude (Chapter 2).

- I integrate the proposed hand contact motion planner into a self-contain fall recovery system including fall detection, fall stabilization and push-up recovery. This system is realized with less expensive sensors and microcomputer and a number of issues like limited computational power and time delays from communication are addressed during the implementation (Sec. 2.5.1).

- I develop of a generalized humanoid fall recovery planning that unifies inertial shaping, protective stepping, and hand contact strategies in arbitrary environment. This algorithm unifies existing recovery strategies, such as inertial shaping, protective stepping, and hand contact, and automatically plans one strategy or a combination of strategies to regain robot's balance based on its disturbed state and nearby environment features (Chapter 3). I also propose an effective algorithm to generate promising initial seeds for trajectory optimization (Sec. 3.3.4)

- I development and implement of an efficient online calibration algorithm for autonomous vehicle's longitudinal dynamical system and experiments in real-world road-tests demonstrate that our proposed online calibration module effectively decreases vehicle's longitudinal position error by 40% after few laps on test road (Chapter 4).

5.2 Suggested Future Research

We would like to address some limitations of the proposed methods in future research.

Hand Contact in 3-D: The proposed hand contact planner protects humanoid robot from falling by making contact with a point on the falling plane of the robot. Although it allows the selection of a certain contact points from the environment,

it neglects other promising contact points out of the falling plane. This negligence can cause robot to choose a contact point where a large friction force is required to keep the contact fixed. To address this limitation, the proposed three-link model can be adapted to a 3-D model by enabling the "shoulder" to "hand" link make 3-D contact (revolute joint to spherical joint). With this extension, robot now has a larger space to search for hand contact point and can recover itself from falling in 3-D environment.

Fast Planning using Machine Learning: This dissertation focuses on motion planning with optimization-based approach. For dynamical systems with high DOFs, this optimization method can take a considerable amount of time, which makes it unsuitable for real-time use. To address this time issue, we are interested in firstly applying the proposed method to gather a large database of robot trajectories and then using a supervised learning approach to rapidly generate a whole-body control reference for fall recovery.

Grasping for Stabilization: The algorithms proposed in this dissertation plan protective contact point using robot extremities for fall recovery. Though hand contact recovery can be generated in the proposed motion planner, this hand contact behavior does not include robot's capability of grasping. For example, when the robot is pushed towards a door, our planner attempts to explore optimal contact point on the surface of this door, but will not consider grasping door lever or knob to stabilize itself from falling. Grasping enables several contact points to be used from robot hand and can produce a large set of contact wrenches to regain robot's balance. As a result, an interesting direction is to take grasping into the exploration of protective contact points.

Bibliography

[Aa10] A. H. Adiwahono, , and and. Humanoid robot push recovery through walking phase modification. In *2010 IEEE Conference on Robotics, Automation and Mechatronics*, pages 569–574, June 2010.

[ABD+98] Nancy M. Amato, O. Burchan Bayazit, Lucia K. Dale, Christopher Jones, and Daniel Vallejo. Obprm: An obstacle-based prm for 3d workspaces. In *Proceedings of the Third Workshop on the Algorithmic Foundations of Robotics on Robotics: The Algorithmic Perspective*, WAFR '98, page 155–168, USA, 1998. A. K. Peters, Ltd.

[ACMD+18] Bernardo Aceituno-Cabezas, Carlos Mastalli, Hongkai Dai, Michele Focchi, Andreea Radulescu, Darwin G Caldwell, José Cappelletto, Juan C Grieco, Gerardo Fernández-López, and Claudio Semini. Simultaneous contact, gait, and motion planning for robust multilegged locomotion via mixed-integer convex optimization. *IEEE Robotics and Automation Letters*, 3(3):2531–2538, 2018.

[AF97] T. Arakawa and T. Fukuda. Natural motion generation of biped locomotion robot using hierarchical trajectory generation method consisting of ga, ep layers. In *Proceedings of International Conference on Robotics and Automation*, volume 1, pages 211–216 vol.1, April 1997.

[AK06] T. Aono and T. Kowatari. Throttle-control algorithm for improving engine response based on air-intake model and throttle-response model. *IEEE Transactions on Industrial Electronics*, 53(3):915–921, 2006.

[AOI17] Yuki Asano, Kei Okada, and Masayuki Inaba. Design principles of a human mimetic humanoid: Humanoid platform to study human intelligence and internal body system. *Science Robotics*, 2(13), 2017.

[ARW12] Z. Aftab, T. Robert, and P. Wieber. Ankle, hip and stepping strategies for humanoid balance recovery with a single model predictive control scheme. In *2012 12th IEEE-RAS International Conference on Humanoid Robots (Humanoids 2012)*, pages 159–164, Nov 2012.

[AS99] J A. Sethian. Level set methods and fast marching methods. evolving interfaces in computational geometry, fluid mechanics, computer vision, and materials science. *vol. 3. Cambridge university press, 1999.*, 3, 04 1999.

[AT15] Oktay Arslan and Panagiotis Tsiotras. Machine learning guided exploration for sampling-based motion planning algorithms. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2646–2652. IEEE, 2015.

[Aub90] J. Aubin. A survey of viability theory. *SIAM Journal on Control and Optimization*, 28(4):749–788, 1990.

[AUKC08] Kutluk Bilge Arikan, Y. Samim Unlusoy, I. Korkmaz, and Okay Celebi. Identification of linear handling models for road vehicles. *Vehicle System Dynamics - VEH SYST DYN*, 46:621–645, 07 2008.

[BCS17] Mohak Bhardwaj, Sanjiban Choudhury, and Sebastian Scherer. Learning heuristic search via imitation. *arXiv preprint arXiv:1707.03034*, 2017.

[BDLS00] H. G. Bock, M. M. Diehl, D. B. Leineweber, and J. P. Schlöder. A direct multiple shooting method for real-time optimization of nonlinear dae processes. In Frank Allgöwer and Alex Zheng, editors, *Nonlinear Model Predictive Control*, pages 245–267, Basel, 2000. Birkhäuser Basel.

[Bel03] Richard Ernest Bellman. *Dynamic Programming.* Dover Publications, Inc., USA, 2003.

[Bel10] Richard Bellman. *Dynamic Programming.* Princeton University Press, Princeton, NJ, USA, 2010.

[Bes10] Matthew C. Best. Identifying tyre models directly from vehicle test data using an extended kalman filter. *Vehicle System Dynamics*, 48(2):171–187, 2010.

[Bet98] John T. Betts. Survey of numerical methods for trajectory optimization. *Journal of Guidance, Control, and Dynamics*, 21(2):193–207, 1998.

[Bet09] John T. Betts. *Practical Methods for Optimal Control and Estimation Using Nonlinear Programming.* Cambridge University Press, New York, NY, USA, 2nd edition, 2009.

[BG07] P. Bhattacharya and M. L. Gavrilova. Voronoi diagram in optimal path planning. In *4th International Symposium on Voronoi Diagrams in Science and Engineering (ISVD 2007)*, pages 38–47, 2007.

[BGC+19] Claudine Badue, Rânik Guidolini, Raphael Vivacqua Carneiro, Pedro Azevedo, Vinicius Brito Cardoso, Avelino Forechi, Luan Jesus, Rodrigo Berriel, Thiago Paixão, Filipe Mutz, Lucas Veronese, Thiago Oliveira-Santos, and Alberto Ferreira De Souza. Self-driving cars: A survey, 2019.

[BGR04] V. L. Bageshwar, W. L. Garrard, and R. Rajamani. Model predictive control of transitional maneuvers for adaptive cruise control vehicles. *IEEE Transactions on Vehicular Technology*, 53(5):1573–1585, 2004.

[BK00] R. Bohlin and L. E. Kavraki. Path planning using lazy prm. In *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)*, volume 1, pages 521–528 vol.1, 2000.

[BK12] Karim Bouyarmane and Abderrahmane Kheddar. Humanoid robot locomotion and manipulation step planning. *Advanced Robotics*, 26(10):1099–1126, 2012.

[BL08] T. Bretl and S. Lall. Testing static equilibrium for legged robots. *IEEE Transactions on Robotics*, 24(4):794–807, Aug 2008.

[Bre06] Timothy Bretl. Motion planning of multi-limbed robots subject to equilibrium constraints: The free-climbing robot problem. *The International Journal of Robotics Research*, 25(4):317–342, 2006.

[BSFK09] D. Berenson, S. S. Srinivasa, D. Ferguson, and J. J. Kuffner. Manipulation planning on constraint manifolds. In *2009 IEEE International Conference on Robotics and Automation*, pages 625–632, 2009.

[BSK11] Dmitry Berenson, Siddhartha Srinivasa, and James Kuffner. Task space regions: A framework for pose-constrained manipulation planning. *The International Journal of Robotics Research*, 30(12):1435–1460, 2011.

[Car16] Stephane Caron. Friction cones, 2016.

[CFS06] J. Carsten, D. Ferguson, and A. Stentz. 3d field d: Improved path planning and replanning in three dimensions. In *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3381–3386, 2006.

[CPN15] S. Caron, Q. Pham, and Y. Nakamura. Stability of surface contacts for humanoid robots: Closed-form formulae of the contact wrench cone for rectangular support areas. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5107–5112, May 2015.

[CPN17] S. Caron, Q. Pham, and Y. Nakamura. Zmp support areas for multi-contact mobility under frictional constraints. *IEEE Transactions on Robotics*, 33(1):67–80, Feb 2017.

[CTD+10] J. Chestnutty, Y. Takaokaz, M. Doiz, K. Sugaz, and S. Kagamiy. Safe adjustment regions for legged locomotion paths. In *2010 10th IEEE-RAS International Conference on Humanoid Robots*, pages 224–229, Dec 2010.

[Dep69] General Electric Ordinance Dept. *1969 – GE Walking Truck – Ralph Mosher (American)*, 1969. `http://cyberneticzoo.com/walking-machines/1969-ge-walking-truck-ralph-mosher-american/` [Online; accessed October 1, 2020].

[Dij59] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numer. Math.*, 1(1):269–271, December 1959.

[DJF+17] C. Dario Bellicoso, F. Jenelten, P. Fankhauser, C. Gehring, J. Hwangbo, and M. Hutter. Dynamic locomotion and whole-body control for quadrupedal robots. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3359–3365, 2017.

[dLMH10] Martin de Lasa, Igor Mordatch, and Aaron Hertzmann. Feature-based locomotion controllers. In *ACM SIGGRAPH 2010 Papers*, SIGGRAPH '10, New York, NY, USA, 2010. Association for Computing Machinery.

[DLOS98] A. De Luca, G. Oriolo, and C. Samson. *Feedback control of a non-holonomic car-like robot*, pages 171–253. Springer Berlin Heidelberg, Berlin, Heidelberg, 1998.

[DMN+14] A. Del Prete, N. Mansard, F. Nori, G. Metta, and L. Natale. Partial force control of constrained floating-base robots. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3227–3232, 2014.

[DPP15] J. E. A. Dias, G. A. S. Pereira, and R. M. Palhares. Longitudinal model identification and velocity control of an autonomous car. *IEEE*

Transactions on Intelligent Transportation Systems, 16(2):776–786, 2015.

[dSEGA05] P. Gonzalez de Santos, J. Estremera, E. Garcia, and M. Armada. Including joint torques and power consumption in the stability margin of walking robots. *Autonomous Robots*, 18(1):43–57, Jan 2005.

[DTM18] A. Del Prete, S. Tonneau, and N. Mansard. Zero step capturability for legged robots in multicontact. *IEEE Transactions on Robotics*, 34(4):1021–1034, Aug 2018.

[DVT14] H. Dai, A. Valenzuela, and R. Tedrake. Whole-body motion planning with centroidal dynamics and full kinematics. In *2014 IEEE-RAS International Conference on Humanoid Robots*, pages 295–302, Nov 2014.

[Dyn20] Boston Dynamics. *Your Guide to the World of Robotics: Spot*, 2020. `https://robots.ieee.org/robots/spotmini/` [Online; accessed October 1, 2020].

[EFRS20] Peter Englert, Isabel M. Rayas Fernández, Ragesh K. Ramachandran, and Gaurav S. Sukhatme. Sampling-based motion planning on manifold sequences, 2020.

[EK09] K. Erbatur and O. Kurt. Natural zmp trajectories for biped robot reference generation. *IEEE Transactions on Industrial Electronics*, 56(3):835–845, March 2009.

[EMW14] Adrien Escande, Nicolas Mansard, and Pierre-Brice Wieber. Hierarchical quadratic programming: Fast online humanoid-robot motion generation. *The International Journal of Robotics Research*, 33(7):1006–1028, 2014.

[EOA13] J. Englsberger, C. Ott, and A. Albu-Schäffer. Three-dimensional bipedal walking control using divergent component of motion. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2600–2607, Nov 2013.

[EWO+14] J. Englsberger, A. Werner, C. Ott, B. Henze, M. A. Roa, G. Garofalo, R. Burger, A. Beyer, O. Eiberger, K. Schmid, and A. Albu-Schäffer. Overview of the torque-controlled humanoid robot toro. In *2014 IEEE-RAS International Conference on Humanoid Robots*, pages 916–923, 2014.

[FBA+07] P. Falcone, F. Borrelli, J. Asgari, H. E. Tseng, and D. Hrovat. Predictive active steering control for autonomous vehicle systems. *IEEE Transactions on Control Systems Technology*, 15(3):566–580, 2007.

[FCAF17] Lucas E. R. Fernandes, Vinicius Custodio, Gleifer V. Alves, and Michael Fisher. A rational agent controlling an autonomous vehicle: Implementation and formal verification. In Lukas Bulwahn, Maryam Kamali, and Sven Linker, editors, *FVAV@iFM*, volume 257 of *EPTCS*, pages 35–42, 2017.

[Fea07] Roy Featherstone. *Rigid Body Dynamics Algorithms*. Springer-Verlag, Berlin, Heidelberg, 2007.

[FJS+17] Farbod Farshidian, Edo Jelavic, Asutosh Satapathy, Markus Giftthaler, and Jonas Buchli. Real-time motion planning of legged robots: A model predictive control approach. In *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*, pages 577–584. IEEE, 2017.

[FKH+06] K. Fujiwara, S. Kajita, K. Harada, K. Kaneko, M. Morisawa, F. Kanehiro, S. Nakaoka, and H. Hirukawa. Towards an optimal falling motion for a humanoid robot. In *2006 6th IEEE-RAS International Conference on Humanoid Robots*, pages 524–529, Dec 2006.

[FKH+07] K. Fujiwara, S. Kajita, K. Harada, K. Kaneko, M. Morisawa, F. Kanehiro, S. Nakaoka, and H. Hirukawa. An optimal planning of falling motions of a humanoid robot. In *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 456–462, Oct 2007.

[FKK+02] K. Fujiwara, F. Kanehiro, S. Kajita, K. Kaneko, K. Yokoi, and H. Hirukawa. Ukemi: falling motion control to minimize damage to biped humanoid robot. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 3, pages 2521–2526 vol.3, Sep. 2002.

[FS06] Dave Ferguson and Anthony Stentz. Using interpolation to improve path planning: The field d* algorithm. *Journal of Field Robotics*, 23(2):79–101, 2006.

[FS17] U.S. Fish and Wildlife Service. *Bighorn sheep can frequently be seen on or near the Refuge Road during the winter months.*, 2017. `https://www.fws.gov/nwrs/threecolumn.aspx?id=2147608767` [Online; accessed October 1, 2020].

[FSRea13] P. Furgale, U. Schwesinger, M. Rufli, and et al. Toward automated driving in cities using close-to-market sensors: An overview of the v-charge project. In *2013 IEEE Intelligent Vehicles Symposium (IV)*, pages 809–816, 2013.

[FWXA15] Siyuan Feng, Eric Whitman, X. Xinjilefu, and Christopher G. Atkeson. Optimization-based full body control for the darpa robotics challenge. *Journal of Field Robotics*, 32(2):293–312, 2015.

[GA15a] Erico Guizzo and Evan Ackerman. *DARPA Robotics Challenge: A Compilation of Robots Falling Down*, 2015. `https://spectrum.ieee.org/automaton/robotics/humanoids/darpa-robotics-challenge-robots-falling` [Online; accessed October 1, 2020].

[GA15b] Erico Guizzo and Evan Ackerman. *DARPA Robotics Challenge Finals: Know Your Robots*, 2015. `https://spectrum.ieee.org/automaton/robotics/humanoids/darpa-robotics-challenge-drc-finals-know-your-robots` [Online; accessed October 1, 2020].

[GBZ15] Xiqiang Guan, Tengyue Ba, and Jianwu Zhang. Vehicle handling dynamics modelling by a data-driven identification method. *Vehicle System Dynamics*, 53(11):1580–1598, 2015.

[GHW81] R. E. Goddard, H. Hemami, and F. C. Weimer. Biped side step in the frontal plane. In *1981 20th IEEE Conference on Decision and Control including the Symposium on Adaptive Processes*, pages 147–155, Dec 1981.

[GK04] A. Goswami and V. Kallem. Rate of change of angular momentum and balance maintenance of biped robots. In *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA '04. 2004*, volume 4, pages 3785–3790 Vol.4, April 2004.

[GMAM06] S. Garrido, L. Moreno, M. Abderrahim, and F. Martin. Path planning for mobile robot navigation using voronoi diagram and fast marching. In *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2376–2381, 2006.

[GMB06] S. Garrido, L. Moreno, and D. Blanco. Voronoi diagram and fast marching applied to path planning. In *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, pages 3049–3054, 2006.

[GMS02] Philip E. Gill, Walter Murray, and Michael A. Saunders. Snopt: An sqp algorithm for large-scale constrained optimization. *SIAM J. on Optimization*, 12(4):979–1006, April 2002.

[Gos99] Ambarish Goswami. Postural stability of biped robots and the foot-rotation indicator (fri) point. *I. J. Robotics Res.*, 18:523–533, 1999.

[Gre03] Donald T. Greenwood. *Advanced Dynamics*. Cambridge University Press, 2003.

[HC76] H. Hemami and P. Camana. Nonlinear feedback in simple locomotion systems. *IEEE Transactions on Automatic Control*, 21(6):855–860, December 1976.

[HDO16] B. Henze, A. Dietrich, and C. Ott. An approach to combine balancing with hierarchical whole-body control for legged humanoid robots. *IEEE Robotics and Automation Letters*, 1(2):700–707, 2016.

[HDW+10] Andrei Herdt, Holger Diedam, Pierre-Brice Wieber, Dimitar Dimitrov, Katja Mombaur, and Moritz Diehl. Online walking motion generation with automatic footstep placement. *Advanced Robotics*, 24(5-6):719–737, 2010.

[HG77] H. Hemami and C.L. Golliday. The inverted pendulum and biped stability. *Mathematical Biosciences*, 34(1):95 – 110, 1977.

[HG09a] O. Höhn and W. Gerth. Probabilistic balance monitoring for bipedal robots. *The International Journal of Robotics Research*, 28(2):245–256, 2009.

[HG09b] O. Höhn and W. Gerth. Probabilistic balance monitoring for bipedal robots. *The International Journal of Robotics Research*, 28(2):245–256, 2009.

[HHH+06] H. Hirukawa, S. Hattori, K. Harada, S. Kajita, K. Kaneko, F. Kanehiro, K. Fujiwara, and M. Morisawa. A universal stability criterion of the foot contact of legged robots - adios zmp. In *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, pages 1976–1983, May 2006.

[HHHT98] K. Hirai, M. Hirose, Y. Haikawa, and T. Takenaka. The development of honda humanoid robot. In *Proceedings. 1998 IEEE International Conference on Robotics and Automation (Cat. No.98CH36146)*, volume 2, pages 1321–1326 vol.2, May 1998.

[HHL14] M. A. Hopkins, D. W. Hong, and A. Leonessa. Humanoid locomotion on uneven terrain using the time-varying divergent component of motion. In *2014 IEEE-RAS International Conference on Humanoid Robots*, pages 266–272, Nov 2014.

[HKKH03] K. Harada, S. Kajita, K. Kaneko, and H. Hirukawa. Zmp analysis for arm/leg coordination. In *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003) (Cat. No.03CH37453)*, volume 1, pages 75–81 vol.1, Oct 2003.

[HKL+98] David Hsu, Lydia E. Kavraki, Jean-Claude Latombe, Rajeev Motwani, and Stephen Sorkin. On finding narrow passages with probabilistic roadmap planners. In *Proceedings of the Third Workshop on the Algorithmic Foundations of Robotics on Robotics: The Algorithmic Perspective*, WAFR '98, page 141–153, USA, 1998. A. K. Peters, Ltd.

[HL10] Kris Hauser and Jean-Claude Latombe. Multi-modal motion planning in non-expansive spaces. *The International Journal of Robotics Research*, 29(7):897–915, 2010.

[HL15] Sehoon Ha and C. K. Liu. Multiple contact planning for minimizing damage of humanoid falls. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2761–2767, Sep. 2015.

[HM94] Yildirim Hurmuzlu and Dan B. Marghitu. Rigid body collisions of planar kinematic chains with multiple contact points. *The International Journal of Robotics Research*, 13(1):82–92, 1994.

[HNR68] P. E. Hart, N. J. Nilsson, and B. Raphael. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, 4(2):100–107, 1968.

[HNTH11] Kris Hauser and Victor Ng-Thow-Hing. Randomized multi-modal motion planning for a humanoid robot manipulation task. *The International Journal of Robotics Research*, 30(6):678–698, 2011.

[HPTC13] E. M. Hoffman, N. Perrin, N. G. Tsagarakis, and D. G. Caldwell. Upper limb compliant strategy exploiting external physical constraints for humanoid fall avoidance. In *13th IEEE-RAS International Conference on Humanoid Robots*, Oct 2013.

[HRG+14] A. Herzog, L. Righetti, F. Grimminger, P. Pastor, and S. Schaal. Balancing experiments on a torque-controlled humanoid with hierarchical inverse dynamics. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 981–988, 2014.

[HS04] Han-Pang Huang and Shu-Yun Chung. Dynamic visibility graph for path planning. In *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE Cat. No.04CH37566)*, volume 3, pages 2813–2818 vol.3, 2004.

[HTMT07] G. M. Hoffmann, C. J. Tomlin, M. Montemerlo, and S. Thrun. Autonomous automobile trajectory tracking for off-road driving: Controller design, experimental validation and racing. In *2007 American Control Conference*, pages 2296–2301, 2007.

[HTY01] S. Hirose, H. Tsukagoshi, and K. Yoneda. Normalized energy stability margin and its contour of walking vehicles on rough terrain. In *Proceedings 2001 ICRA. IEEE International Conference on Robotics and Automation (Cat. No.01CH37164)*, volume 1, pages 181–186 vol.1, May 2001.

[INC20] ROBOTIS INC. *ROBOTIS MINI[INTL]*, 2020. `http://www.robotis.us/robotis-mini-int` [Online; accessed October 1, 2020].

[Jaz09] Reza N. Jazar. *Vehicle Dynamics: Theory and Application*. Springer, Boston, MA, 2009.

[JLK95] J. A. Janet, R. C. Luo, and M. G. Kay. The essential visibility graph: an approach to global motion planning for autonomous mobile robots. In *Proceedings of 1995 IEEE International Conference on Robotics and Automation*, volume 2, pages 1958–1963 vol.2, 1995.

[JP13] L. Jaillet and J.M. Porta. Efficient asymptotically-optimal path planning on manifolds. *Robotics and Autonomous Systems*, 61(8):797 – 807, 2013.

[JRA02] Jin-Oh Hahn, R. Rajamani, and L. Alexander. Gps-based real-time identification of tire-road friction coefficient. *IEEE Transactions on Control Systems Technology*, 10(3):331–343, 2002.

[JSB+15] Matthew Johnson, Brandon Shrewsbury, Sylvain Bertrand, Tingfan Wu, Daniel Duran, Marshall Floyd, Peter Abeles, Douglas Stephen, Nathan Mertins, Alex Lesman, John Carff, William Rifenburgh, Pushyami Kaveti, Wessel Straatman, Jesper Smith, Maarten Griffioen, Brooke Layton, Tomas de Boer, Twan Koolen, Peter Neuhaus, and Jerry Pratt. Team ihmc's lessons learned from the darpa robotics challenge trials. *Journal of Field Robotics*, 32(2):192–208, 2015.

[JV08] Jinsu Liu and M. Veloso. Online zmp sampling search for biped walking planning. In *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 185–190, 2008.

[JWS12] Jiuguang Wang, E. C. Whitman, and M. Stilman. Whole-body trajectory optimization for humanoid falling. In *American Control Conference (ACC)*. IEEE, jun 2012.

[JX01] X. Ji and J. Xiao. Planning motions compliant to complex contact states. *International Journal of Robotics Research*, 2001.

[KdBR+12] Twan Koolen, Tomas de Boer, John Rebula, Ambarish Goswami, and Jerry Pratt. Capturability-based analysis and control of legged

locomotion, part 1: Theory and application to three simple gait models. *The International Journal of Robotics Research*, 31(9):1094–1113, 2012.

[Kel17] M. Kelly. An introduction to trajectory optimization: How to do your own direct collocation. *SIAM Rev.*, 59:849–904, 2017.

[KF10] S. Karaman and E. Frazzoli. Optimal kinodynamic motion planning using incremental sampling-based methods. In *49th IEEE Conference on Decision and Control (CDC)*, pages 7681–7687, 2010.

[KF11] Sertac Karaman and Emilio Frazzoli. Sampling-based algorithms for optimal motion planning. *The International Journal of Robotics Research*, 30(7):846–894, 2011.

[KG11] SHIVARAM KALYANAKRISHNAN and AMBARISH GOSWAMI. Learning to predict humanoid fall. *International Journal of Humanoid Robotics*, 08(02):245–273, 2011.

[KG15] Nitin R. Kapania and J. Christian Gerdes. Design of a feedback-feedforward steering controller for accurate path tracking and stability at the limits of handling. *Vehicle System Dynamics*, 53(12):1687–1704, 2015.

[Kha87] O. Khatib. A unified approach for motion and force control of robot manipulators: The operational space formulation. *IEEE Journal on Robotics and Automation*, 3(1):43–53, 1987.

[KHHY14] Shuuji Kajita, Hirohisa Hirukawa, Kensuke Harada, and Kazuhito Yokoi. *Introduction to Humanoid Robotics*. Springer Publishing Company, Incorporated, 2014.

[KHJ+17] Eric Krotkov, Douglas Hackett, Larry Jackel, Michael Perschbacher, James Pippine, Jesse Strauss, Gill Pratt, and Christopher Orlowski. The darpa robotics challenge finals: Results and perspectives. *Journal of Field Robotics*, 34(2):229–240, 2017.

[KHK+08] K. Kaneko, K. Harada, F. Kanehiro, G. Miyamori, and K. Akachi. Humanoid robot hrp-3. In *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2471–2478, 2008.

[KHL17] V. C. V. Kumar, S. Ha, and C. K. Liu. Learning a unified control policy for safe falling. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3940–3947, Sep. 2017.

[Kin09] Davis E. King. Dlib-ml: A machine learning toolkit. *Journal of Machine Learning Research*, 10:1755–1758, 2009.

[KKK+01] S. Kajita, F. Kanehiro, K. Kaneko, K. Yokoi, and H. Hirukawa. The 3d linear inverted pendulum mode: a simple modeling for a biped walking pattern generation. In *Proceedings 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 1, pages 239–246 vol.1, Oct 2001.

[KKK+03a] S. Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, K. Harada, K. Yokoi, and H. Hirukawa. Biped walking pattern generation by using preview control of zero-moment point. In *2003 IEEE International Conference on Robotics and Automation (Cat. No.03CH37422)*, volume 2, pages 1620–1626 vol.2, Sep. 2003.

[KKK+03b] S. Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, K. Harada, K. Yokoi, and H. Hirukawa. Resolved momentum control: humanoid motion planning based on the linear and angular momentum. In *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003)*, volume 2, pages 1644–1650 vol.2, Oct 2003.

[KKK+17] T. Kamioka, H. Kaneko, M. Kuroda, C. Tanaka, S. Shirokura, M. Takeda, and T. Yoshiike. Dynamic gait transition between walking, running and hopping for push recovery. In *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*, pages 1–8, Nov 2017.

[KKM+11] K. Kaneko, F. Kanehiro, M. Morisawa, K. Akachi, G. Miyamori, A. Hayashi, and N. Kanehira. Humanoid robot hrp-4 - humanoid robotics platform with lightweight and slim body. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4400–4407, 2011.

[KKN+02] James J. Kuffner, Satoshi Kagami, Koichi Nishiwaki, Masayuki Inaba, and Hirochika Inoue. Dynamically-stable motion planning for humanoid robots. *Autonomous Robots*, 12(1):105–118, Jan 2002.

[KKS+19] K. Kaneko, H. Kaminaga, T. Sakaguchi, S. Kajita, M. Morisawa, I. Kumagai, and F. Kanehiro. Humanoid robot hrp-5p: An electrically actuated humanoid robot with high-power and wide-range joints. *IEEE Robotics and Automation Letters*, 4(2):1431–1438, 2019.

[KL00] J. J. Kuffner and S. M. LaValle. Rrt-connect: An efficient approach to single-query path planning. In *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and*

Automation. Symposia Proceedings (Cat. No.00CH37065), volume 2, pages 995–1001 vol.2, 2000.

[KMM+10] S. Kajita, M. Morisawa, K. Miura, S. Nakaoka, K. Harada, K. Kaneko, F. Kanehiro, and K. Yokoi. Biped walking stabilization based on linear inverted pendulum tracking. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4489–4496, Oct 2010.

[KOK+03] T. Kito, J. Ota, R. Katsuki, T. Mizuta, T. Arai, T. Ueyama, and T. Nishiyama. Smooth path planning by using visibility graph-like method. In *2003 IEEE International Conference on Robotics and Automation (Cat. No.03CH37422)*, volume 3, pages 3770–3775 vol.3, 2003.

[KSLO96] L. E. Kavraki, P. Svestka, J. . Latombe, and M. H. Overmars. Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation*, 12(4):566–580, Aug 1996.

[KSO+16] T. Kaneko, M. Sekiya, K. Ogata, S. Sakaino, and T. Tsuji. Force control of a jumping musculoskeletal robot with pneumatic artificial muscles. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5813–5818, Oct 2016.

[KT91] S. Kajita and K. Tani. Study of dynamic biped locomotion on rugged terrain-derivation and application of the linear inverted pendulum mode. In *Proceedings. 1991 IEEE International Conference on Robotics and Automation*, 1991.

[KUSP16] Beobkyoon Kim, Terry Taewoong Um, Chansu Suh, and F. C. Park. Tangent bundle rrt: A randomized algorithm for constrained motion planning. *Robotica*, 34(1):202–225, 2016.

[KW08] J.G. Karssen and M. Wisse. Fall detection in walking robots by multi-way principal component analysis. *Robotica, 27 (2009)*, 2008.

[KW09] J. G. Daniel Karssen and Martijn Wisse. Fall detection in walking robots by multi-way principal component analysis. *Robotica*, 27:249–257, 2009.

[KYK92] S. Kajita, T. Yamaura, and A. Kobayashi. Dynamic walking control of a biped robot along a potential energy conserving orbit. *IEEE Transactions on Robotics and Automation*, 8(4):431–438, Aug 1992.

[LDHW16] Dingsheng Luo, Yian Deng, Xiaoqiang Han, and Xihong Wu. Biped robot falling motion control with human-inspired active compliance. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, October 2016.

[LG07] S. Lee and A. Goswami. Reaction mass pendulum (rmp): An explicit model for centroidal angular momentum of humanoid robots. In *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pages 4667–4672, 2007.

[LG11] Sung-Hee Lee and Ambarish Goswami. Fall on backpack: Damage minimizing humanoid fall on targeted body segment using momentum control. In *8th International Conference on Multibody Systems, Nonlinear Dynamics, and Control, Parts A and B*. ASME, January 2011.

[Lin01] Nicholas P. Linthorne. Analysis of standing vertical jumps using a force platform. *American Journal of Physics*, 69(11):1198–1204, 2001.

[LJJK01] Steven M. LaValle and Jr. James J. Kuffner. Randomized kinodynamic planning. *The International Journal of Robotics Research*, 20(5):378–400, 2001.

[LO88] W. . Lee and D. E. Orin. The kinematics of motion planning for multilegged vehicles over uneven terrain. *IEEE Journal on Robotics and Automation*, 4(2):204–212, 1988.

[LPW79] Tomás Lozano-Pérez and Michael A. Wesley. An algorithm for planning collision-free paths among polyhedral obstacles. *Commun. ACM*, 22(10):560–570, October 1979.

[LS93] B. . Lin and S. . Song. Dynamic modeling, stability and energy efficiency of a quadrupedal walking machine. In *[1993] Proceedings IEEE International Conference on Robotics and Automation*, pages 367–373 vol.3, May 1993.

[LT04] Weiwei Li and Emanuel Todorov. Iterative linear quadratic regulator design for nonlinear biological movement systems. In *ICINCO (1)*, pages 222–229, 2004.

[LTK92] Q. Li, A. Takanishi, and I. Kato. Learning control of compensative trunk motion for biped walking robot based on zmp stability criterion. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 1, pages 597–603, July 1992.

[LTK93] Q. Li, A. Takanishi, and I. Kato. Learning control for a biped walking robot with a trunk. In *Proceedings of 1993 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '93)*, volume 3, pages 1771–1777 vol.3, July 1993.

[LW07] C. B. Low and D. Wang. Integrated estimation for wheeled mobile robot posture, velocities, and wheel skidding perturbations. In *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pages 2355–2360, 2007.

[LWP80] J. Luh, M. Walker, and R. Paul. Resolved-acceleration control of mechanical manipulators. *IEEE Transactions on Automatic Control*, 25(3):468–474, June 1980.

[LZC+15] Z. Li, C. Zhou, J. Castano, , F. Negrello, N. G. Tsagarakis, and D. G. Caldwell. Fall prediction of legged robots based on energy state and its implication of balance augmentation: A study on the humanoid. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5094–5100, May 2015.

[MAN04] Ellips Masehian and M. R. Amin-Naseri. A voronoi diagram-visibility graph-potential field compound algorithm for robot path planning. *Journal of Robotic Systems*, 21(6):275–300, 2004.

[May66] David Mayne. A second-order gradient method for determining optimal trajectories of non-linear discrete-time systems. *International Journal of Control*, 3(1):85–95, 1966.

[MB11] M. Missura and S. Behnke. Lateral capture steps for bipedal walking. In *2011 11th IEEE-RAS International Conference on Humanoid Robots*, pages 401–408, Oct 2011.

[MB13] Marcell Missura and Sven Behnke. Omnidirectional capture steps for bipedal walking. In *13th IEEE-RAS International Conference on Humanoid Robots (Humanoids)*. IEEE, oct 2013.

[MBJA15] C. Mandery, J. Borras, M. Jochner, and T. Asfour. Analyzing whole-body pose transitions in multi-contact motions. In *IEEE-RAS 15th International Conference on Humanoid Robots*, Nov 2015.

[McG88] T McGeer. Stability and control of two-dimensional biped walking. In *Technical Report 1.*, Center for Systems Science, Simon Fraser University, Burnaby, B.C., Canada, 1988.

[MDG+17a] Tobia Marcucci, Robin Deits, Marco Gabiccini, Antonio Bicchi, and Russ Tedrake. Approximate hybrid model predictive control

for multi-contact push recovery in complex environments. In *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*, pages 31–38. IEEE, 2017.

[MDG+17b] Tobia Marcucci, Robin Deits, Marco Gabiccini, Antonio Bicchi, and Russ Tedrake. Approximate hybrid model predictive control for multi-contact push recovery in complex environments. In *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*, pages 31–38. IEEE, 2017.

[Meh92] S. Mehrotra. On the implementation of a primal-dual interior point method. *SIAM Journal on Optimization*, 2(4):575–601, 1992.

[MES85] DOMINIC A. MESSURI. *OPTIMIZATION OF THE LOCOMOTION OF A LEGGED VEHICLE WITH RESPECT TO MANEUVERABILITY*. PhD thesis, The Ohio State University, 1985.

[MF68] R.B. McGhee and A.A. Frank. On the stability properties of quadruped creeping gaits. *Mathematical Biosciences*, 3:331 – 351, 1968.

[MI79] R. B. Mcghee and G. I. Iswandhi. Adaptive locomotion of a multilegged robot over rough terrain. *IEEE Transactions on Systems, Man, and Cybernetics*, 9(4):176–182, April 1979.

[MML11] D. Mansour, A. Micaelli, and P. Lemerle. A computational approach for push recovery in case of multiple noncoplanar contacts. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3213–3220, Sep. 2011.

[MTP12] Igor Mordatch, Emanuel Todorov, and Zoran Popović. Discovery of complex behaviors through contact-invariant optimization. *ACM Trans. Graph.*, 31(4):43:1–43:8, July 2012.

[MW89] S Mahalingam and W.L. Whittaker. Terrain adaptive gaits for walkers with completely overlapping leg workspaces. *Robots*, 13:1–14, May 1989.

[MZS09] Adriano Macchietto, Victor Zordan, and Christian R. Shelton. Momentum control for balance. In *ACM SIGGRAPH 2009 Papers*, SIGGRAPH '09, pages 80:1–80:8, New York, NY, USA, 2009. ACM.

[Nag91] Peter V. Nagy. *An investigation of walker/terrain interaction*. PhD thesis, Carnegie Mellon University, 1991.

[NCM+08] Jun Nakanishi, Rick Cory, Michael Mistry, Jan Peters, and Stefan Schaal. Operational space control: A theoretical and empirical comparison. *The International Journal of Robotics Research*, 27(6):737–757, 2008.

[NDW94] P. V. Nagy, S. Desa, and W. L. Whittaker. Energy-based stability measures for reliable locomotion of statically stable walkers. Theory and application. *International Journal of Robotics Research*, 13(3):272–287, 1994.

[NKT+18] S. Noda, Y. Kakiuchi, H. Takeda, K. Okada, and M. Inaba. Goal-oriented simulation-based motion interpolator for complex contact transition: Experiments on knee-contact behavior. In *2018 IEEE-RAS 18th International Conference on Humanoid Robots (Humanoids)*, pages 1–7, Nov 2018.

[NNY+03] S. Nakaoka, A. Nakazawa, K. Yokoi, H. Hirukawa, and K. Ikeuchi. Generating whole body motions for a biped humanoid robot from captured human dances. In *2003 IEEE International Conference on Robotics and Automation (Cat. No.03CH37422)*, volume 3, pages 3905–3910 vol.3, Sep. 2003.

[NSTJ19] Hanlin Niu, Al Savvaris, Antonios Tsourdos, and Ze Ji. Voronoi-visibility roadmap-based path planning algorithm for unmanned surface vehicles. *Journal of Navigation*, 72(4):850–874, 2019.

[NW06] Jorge Nocedal and Stephen Wright. *Numerical optimization*. Springer Science & Business Media, 2006.

[OGL13] David E. Orin, Ambarish Goswami, and Sung-Hee Lee. Centroidal dynamics of a humanoid robot. *Autonomous Robots*, 35(2):161–176, Oct 2013.

[OK07] K. Ogata and Y. Kuniyoshi. Falling motion control for humanoid robots while walking. In *2007 7th IEEE-RAS International Conference on Humanoid Robots*, pages 306–311, Nov 2007.

[ORI76] DAVID E. ORIN. *INTERACTIVE CONTROL OF A SIX-LEGGED VEHICLE WITH OPTIMIZATION OF BOTH STABILITY AND ENERGY*. PhD thesis, The Ohio State University, 1976. Copyright - Database copyright ProQuest LLC; ProQuest does not claim copyright in the individual underlying works; Last updated - 2016-06-09.

[OTK08] K. Ogata, K. Terada, and Y. Kuniyoshi. Real-time selection and generation of fall damage reduction actions for humanoid robots.

In *Humanoids 2008 - 8th IEEE-RAS International Conference on Humanoid Robots*, pages 233–238, Dec 2008.

[Pac12] Hans B. Pacejka. *Tire and Vehicle Dynamics*. Butterworth-Heinemann, Oxford, 2012.

[PBvdP15] Xue Bin Peng, Glen Berseth, and Michiel van de Panne. Dynamic terrain traversal skills using reinforcement learning. *ACM Trans. Graph.*, 34(4):80:1–80:11, July 2015.

[PBYVDP17a] Xue Bin Peng, Glen Berseth, Kangkang Yin, and Michiel Van De Panne. Deeploco: Dynamic locomotion skills using hierarchical deep reinforcement learning. *ACM Trans. Graph.*, 36(4):41:1–41:13, July 2017.

[PBYvdP17b] Xue Bin Peng, Glen Berseth, KangKang Yin, and Michiel van de Panne. Deeploco: Dynamic locomotion skills using hierarchical deep reinforcement learning. *ACM Transactions on Graphics (Proc. SIGGRAPH 2017)*, 36(4), 2017.

[PCCL12] Mike Phillips, Benjamin J Cohen, Sachin Chitta, and Maxim Likhachev. E-graphs: Bootstrapping planning with experience graphs. In *Robotics: Science and Systems*, volume 5, 2012.

[PCDG06] J. Pratt, J. Carff, S. Drakunov, and A. Goswami. Capture point: A step toward humanoid push recovery. In *2006 6th IEEE-RAS International Conference on Humanoid Robots*, pages 200–207, Dec 2006.

[PCT14] Michael Posa, Cecilia Cantu, and Russ Tedrake. A direct method for trajectory optimization of rigid bodies through contact. *The International Journal of Robotics Research*, 33(1):69–81, 2014.

[PCY+16] B. Paden, M. Cap, S. Z. Yong, D. Yershov, and E. Frazzoli. A survey of motion planning and control techniques for self-driving urban vehicles. *IEEE Transactions on Intelligent Vehicles*, 1(1):33–55, 2016.

[PKT16] M. Posa, S. Kuindersma, and R. Tedrake. Optimization and stabilization of trajectories for constrained dynamical systems. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1366–1373, May 2016.

[PO06] L. R. Palmer and D. E. Orin. Attitude control of a quadruped trot while turning. In *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5743–5749, 2006.

[POE+17] N. Perrin, C. Ott, J. Englsberger, O. Stasse, F. Lamiraux, and D. G. Caldwell. Continuous legged locomotion planning. *IEEE Transactions on Robotics*, 33(1):234–239, Feb 2017.

[PP98] Jerry Pratt and Gill Pratt. Exploiting natural dynamics in the control of a planar bipedal walking robot. In *Proceedings of the Thirty-Sixth Annual Allerton Conference on Communication, Control, and Computing*, 11 1998.

[PR96] Evangelos Papadopoulos and Dariela Rey. New measure of tipover stability margin for mobile manipulators. In *Proceedings - IEEE International Conference on Robotics and Automation*, volume 4, pages 3111 – 3116 vol.4, 05 1996.

[PT06] J.E. Pratt and R. Tedrake. *Velocity-Based Stability Margins for Fast Bipedal Walking*, pages 299–324. Springer Berlin Heidelberg, Berlin, Heidelberg, 2006.

[Puk94] Friedrich Pukelsheim. The three sigma rule. *The American Statistician*, 48(2):88–91, 1994.

[Rai86] Marc H. Raibert. *Legged Robots That Balance*. Massachusetts Institute of Technology, USA, 1986.

[Raj06] R Rajamani. *Vehicle Dynamics and Control*. Springer US, 2006.

[Ray95] L. R. Ray. Nonlinear state and tire force estimation for advanced vehicle control. *IEEE Transactions on Control Systems Technology*, 3(1):117–124, 1995.

[RB06] R. Renner and S. Behnke. Instability detection and fall avoidance for a humanoid using attitude sensors and reflexes. In *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2967–2973, Oct 2006.

[RCPG07] J. Rebula, F. Canas, J. Pratt, and A. Goswami. Learning capture points for humanoid push recovery. In *2007 7th IEEE-RAS International Conference on Humanoid Robots*, pages 65–72, Nov 2007.

[RH15] O. E. Ramos and K. Hauser. Generalizations of the capture point to nonlinear center of mass paths and uneven terrain. In *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*, pages 851–858, Nov 2015.

[RMS14] O. E. Ramos, N. Mansard, and P. Souères. Whole-body motion integrating the capture point in the operational space inverse dynamics

control. In *2014 IEEE-RAS International Conference on Humanoid Robots*, pages 707–712, Nov 2014.

[RSH+15] Nicolaus A. Radford, Philip Strawser, Kimberly Hambuchen, Joshua S. Mehling, William K. Verdeyen, A. Stuart Donnan, James Holley, Jairo Sanchez, Vienny Nguyen, Lyndon Bridgwater, Reginald Berka, Robert Ambrose, Mason Myles Markee, N. J. Fraser-Chanpong, Christopher McQuin, John D. Yamokoski, Stephen Hart, Raymond Guo, Adam Parsons, Brian Wightman, Paul Dinh, Barrett Ames, Charles Blakely, Courtney Edmondson, Brett Sommers, Rochelle Rea, Chad Tobler, Heather Bibby, Brice Howard, Lei Niu, Andrew Lee, Michael Conover, Lily Truong, Ryan Reed, David Chesney, Robert Platt Jr, Gwendolyn Johnson, Chien-Liang Fok, Nicholas Paine, Luis Sentis, Eric Cousineau, Ryan Sinnet, Jordan Lack, Matthew Powell, Benjamin Morris, Aaron Ames, and Jide Akinyode. Valkyrie: Nasa's first bipedal humanoid robot. *Journal of Field Robotics*, 32(3):397–419, 2015.

[Ryg93] L.A. Rygg. Mechanical horse, U.S. Patent 491927A, Feb. 14. 1893.

[RZBS09] N. Ratliff, M. Zucker, J. A. Bagnell, and S. Srinivasa. Chomp: Gradient optimization techniques for efficient motion planning. In *2009 IEEE International Conference on Robotics and Automation*, pages 489–494, 2009.

[SA10a] B. J. Stephens and C. G. Atkeson. Dynamic balance force control for compliant humanoid robots. In *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1248–1255, Oct 2010.

[SA10b] B. J. Stephens and C. G. Atkeson. Push recovery by stepping for humanoid robots with force controlled joints. In *2010 10th IEEE-RAS International Conference on Humanoid Robots*, pages 52–59, Dec 2010.

[SAE18] SAE(2018). Taxonomy and definitions for terms related to driving automation systems for on-road motor vehicles. *Technical Report SAE International*, 2018.

[SAMR18] Wilko Schwarting, Javier Alonso-Mora, and Daniela Rus. Planning and decision-making for autonomous vehicles. *Annual Review of Control, Robotics, and Autonomous Systems*, 1(1):187–210, 2018.

[SBK+19] Nathan Spielberg, Matthew Brown, Nitin Kapania, John Kegelman, and J. Gerdes. Neural network vehicle models for high-performance automated driving. *Science Robotics*, 4, 03 2019.

[SCBK17] V. Samy, S. Caron, K. Bouyarmane, and A. Kheddar. Post-impact adaptive compliance for humanoid falls using predictive control of a reduced model. In *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*, pages 655–660, Nov 2017.

[SDH+14] John Schulman, Yan Duan, Jonathan Ho, Alex Lee, Ibrahim Awwal, Henry Bradlow, Jia Pan, Sachin Patil, Ken Goldberg, and Pieter Abbeel. Motion planning with sequential convex optimization and convex collision checking. *The International Journal of Robotics Research*, 33(9):1251–1270, 2014.

[SK07] Bruno Siciliano and Oussama Khatib. *Springer Handbook of Robotics*. Springer-Verlag, Berlin, Heidelberg, 2007.

[SL15] Sehoon Ha and C. Karen Liu. Multiple contact planning for minimizing damage of humanoid falls. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, sep 2015.

[SLC+90] C. L. Shih, Y. Z. Li, S. Churng, T. T. Lee, and W. A. Gruver. Trajectory synthesis and physical admissibility for a biped robot during the single-support phase. In *Proceedings., IEEE International Conference on Robotics and Automation*, pages 1646–1652 vol.3, May 1990.

[SRMMK11] Neal Seegmiller, Forrest Rogers-Marcovitz, Greg Miller, and Alonzo Kelly. A unified perturbative dynamics approach to online vehicle model identification. In *ISRR*, 2011.

[SRMMK13] Neal Seegmiller, Forrest Rogers-Marcovitz, Greg Miller, and Alonzo Kelly. Vehicle model identification by integrated prediction error minimization. *The International Journal of Robotics Research*, 32(8):912–931, 2013.

[SS20] S. Singh and F. Sahin. Push recovery for humanoid robots based on linearized double inverted pendulum. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2020.

[SSA08] Dimitri Solomatine, Linda See, and R.J. Abrahart. *Data-Driven Modelling: Concepts, Approaches and Experiences*, volume 68, pages 17–30. Springer, Berlin, Heidelberg, 01 2008.

[Ste07] B. Stephens. Humanoid push recovery. In *2007 7th IEEE-RAS International Conference on Humanoid Robots*, pages 589–595, Nov 2007.

[Sti10] M. Stilman. Global manipulation planning in robot joint space with task constraints. *IEEE Transactions on Robotics*, 26(3):576–584, 2010.

[STV+08] C. Semini, N. G. Tsagarakis, B. Vanderborght, Y. Yang, and D. G. Caldwell. Hyq - hydraulically actuated quadruped robot: Hopping leg prototype. In *2008 2nd IEEE RAS EMBS International Conference on Biomedical Robotics and Biomechatronics*, pages 593–599, Oct 2008.

[Sum00] Heikki Summala. Brake reaction times and driver behavior analysis. *Transportation Human Factors*, 2:217–226, 09 2000.

[SY05] S. Sakka and K. Yokoi. Humanoid vertical jumping based on force feedback and inertial forces optimization. In *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, pages 3752–3757, April 2005.

[SYZ97] A. W. Salatian, Keon Young Yi, and Yuan F. Zheng. Reinforcement learning for a biped robot to climb sloping surfaces. *Journal of Robotic Systems*, 14(4):283–296, 1997.

[TET12] Y. Tassa, T. Erez, and E. Todorov. Synthesis and stabilization of complex behaviors through online trajectory optimization. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4906–4913, Oct 2012.

[TIYK85] Atsuo Takanishi, M. Ishida, Y. Yamazaki, and I. Kato. Realization of dynamic walking by the biped walking robot wl-10rd. In *Unknown Host Publication Title*, pages 459–466. Japan Industrial Robot Assoc, 1985.

[TK16] Benjamin Tam and Navinda Kottege. Fall Avoidance and Recovery for Bipedal Robots using Walking Sticks, dec 2016.

[TKZS16] O. S. Tas, F. Kuhnt, J. M. Zöllner, and C. Stiller. Functional system architectures towards fully automated driving. In *2016 IEEE Intelligent Vehicles Symposium (IV)*, pages 304–309, 2016.

[TMT14] Yuval Tassa, Nicolas Mansard, and Emo Todorov. Control-limited differential dynamic programming. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1168–1175. IEEE, 2014.

[TRH14] Robert Tafner, Markus Reichhartinger, and Martin Horn. Robust online roll dynamics identification of a vehicle using sliding mode concepts. *Control Engineering Practice*, 29:235 – 246, 2014.

[TS89] O. Takahashi and R. J. Schilling. Motion planning in a plane using generalized voronoi diagrams. *IEEE Transactions on Robotics and Automation*, 5(2):143–150, 1989.

[TSH18] Gao Tang, Weidong Sun, and Kris Hauser. Learning trajectories for real-time optimal control of quadrotors. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3620–3625. IEEE, 2018.

[URR+17] Simon Ulbrich, Andreas Reschka, Jens Rieken, Susanne Ernst, Gerrit Bagschik, Frank Dierkes, Marcus Nolte, and Markus Maurer. Towards a functional system architecture for automated vehicles, 2017.

[VB04] MIOMIR VUKOBRATOVIĆ and BRANISLAV BOROVAC. Zero-Moment Point — Thirty Five Years of Its Life. *International Journal of Humanoid Robotics*, 01(01):157–173, 2004.

[VKA+16] Joris Vaillant, Abderrahmane Kheddar, Hervé Audren, François Keith, Stanislas Brossette, Adrien Escande, Karim Bouyarmane, Kenji Kaneko, Mitsuharu Morisawa, Pierre Gergondet, Eiichi Yoshida, Suuji Kajita, and Fumio Kanehiro. Multi-contact vertical ladder climbing with an hrp-2 humanoid. *Autonomous Robots*, 40(3):561–580, Mar 2016.

[VS72a] M. Vukobratović and J. Stepanenko. On the stability of anthropomorphic systems. *Mathematical Biosciences*, 15(1):1 – 37, 1972.

[VS72b] M. Vukobratović and J. Stepanenko. On the stability of anthropomorphic systems. *Mathematical Biosciences*, 15(1):1 – 37, 1972.

[VS73] Miomir Vukobratović and Jurij Stepanenko. Mathematical models of general anthropomorphic systems. *Mathematical Biosciences*, 17(3):191 – 242, 1973.

[VWK+15] Nikolaus Vahrenkamp, Mirko Wächter, Manfred Kröhnert, Kai Welke, and Tamim Asfour. The robot software framework armarx. *it - Information Technology*, 57, 01 2015.

[WH17] S. Wang and K. Hauser. Real-time stabilization of a falling humanoid robot using hand contact: An optimal control approach. In *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*, pages 454–460, Nov 2017.

[WH18a] S. Wang and K. Hauser. Realization of a real-time optimal control strategy to stabilize a falling humanoid robot with hand contact. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3092–3098, May 2018.

[WH18b] S. Wang and K. Hauser. Unified multi-contact fall mitigation planning for humanoids via contact transition tree optimization. In *2018*

IEEE-RAS 18th International Conference on Humanoid Robots (Humanoids), pages 1–9, Nov 2018.

[Wie02] Pierre-Brice Wieber. On the stability of walking systems. In *International Workshop on Humanoid and Human Friendly Robotics*, 2002.

[Wie06] P. Wieber. Trajectory free linear model predictive control for stable walking in the presence of strong perturbations. In *2006 6th IEEE-RAS International Conference on Humanoid Robots*, pages 137–142, Dec 2006.

[WWEO82] M W. Walker and D E. Orin. Efficient dynamic computer simulation of robotic mechanisms. *Journal of Dynamic Systems Measurement and Control-transactions of The Asme - J DYN SYST MEAS CONTR*, 104, 09 1982.

[WWS12] J. Wang, E. C. Whitman, and M. Stilman. Whole-body trajectory optimization for humanoid falling. In *2012 American Control Conference (ACC)*, pages 4837–4842, June 2012.

[XLH17] Zhaoming Xie, C Karen Liu, and Kris Hauser. Differential dynamic programming with nonlinear constraints. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 695–702. IEEE, 2017.

[XZX+20] Yongqian Xiao, Xinglong Zhang, Xin Xu, Xueqing Liu, and Jiahang Liu. A deep learning framework based on koopman operator for data-driven modeling of vehicle dynamics, 2020.

[Yap02] Peter Yap. Grid-based path-finding. In *Proceedings of the 15th Conference of the Canadian Society for Computational Studies of Intelligence on Advances in Artificial Intelligence*, AI '02, page 44–55, Berlin, Heidelberg, 2002. Springer-Verlag.

[YAS+06] Yu Ogura, H. Aikawa, K. Shimomura, H. Kondo, A. Morishima, Hun-ok Lim, and A. Takanishi. Development of a new humanoid robot wabian-2. In *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, pages 76–81, 2006.

[YG14a] S. Yun and A. Goswami. Tripod fall: Concept and experiments of a novel approach to humanoid robot fall damage reduction. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2799–2805, May 2014.

[YG14b] S. Yun and A. Goswami. Tripod fall: Concept and experiments of a novel approach to humanoid robot fall damage reduction. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2799–2805, 2014.

[YGS09] S. Yun, A. Goswami, and Y. Sakagami. Safe fall: Humanoid robot fall direction change through intelligent stepping and inertia shaping. In *2009 IEEE International Conference on Robotics and Automation*, pages 781–787, May 2009.

[YLL13] Shaopu Yang, Yongjie Lu, and Shaohua Li. An overview on vehicle dynamics. *International Journal of Dynamics and Control*, 1:385–395, 2013.

[YNY02] Y. Yokokohji, S. Nomoto, and T. Yoshikawa. Static evaluation of humanoid robot postures constrained to the surrounding environment through their limbs. In *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No.02CH37292)*, volume 2, pages 1856–1863 vol.2, May 2002.

[YWZ+09] J. Yi, H. Wang, J. Zhang, D. Song, S. Jayasuriya, and J. Liu. Kinematic modeling and analysis of skid-steered mobile robots with applications to low-cost inertial-measurement-unit-based motion estimation. *IEEE Transactions on Robotics*, 25(5):1087–1097, 2009.

[YZHL13] S. Yi, B. Zhang, D. Hong, and D. D. Lee. Online learning of low dimensional strategies for high-level push recovery in bipedal humanoid robots. In *2013 IEEE International Conference on Robotics and Automation*, pages 1649–1655, May 2013.

[ZMX+18] Fan Zhu, Lin Ma, Xin Xu, Dingfeng Guo, Xiao Cui, and Qi Kong. Baidu apollo auto-calibration system - an industry-level data-driven and learning based vehicle longitude dynamic calibrating algorithm, 2018.

[ZS90] Chang-De Zhang and Shin-Min Song. Stability analysis of wave-crab gaits of a quadruped. *Journal of Robotic Systems*, 7(2):243–276, 1990.

[ZS12] Y. Zhao and L. Sentis. A three dimensional foot placement planner for locomotion in very rough terrains. In *2012 12th IEEE-RAS International Conference on Humanoid Robots (Humanoids 2012)*, pages 726–733, Nov 2012.

Biography

Education

- **Doctor of Philosophy (Ph.D.)** in Mechanical Engineering Dec. 2020
  Duke University, Durham, North Carolina, USA
- **Master of Engineering (M.Eng)** in Mechanical Engineering Jun. 2015
  Cornell University, Ithaca, New York, USA
- **Bachelor of Engineering (B.Eng)** in Mechanical Engineering Jun. 2014
  Beihang University (BUAA), Beijing, China

Research Interest

- **Humanoid Motion Planning**
- **Numerical Optimization**
- **Dynamics and Controls**

Publications

1. **Shihao Wang**, Canqiang Deng, Qingjie Qi and Tongyi Cao, "Efficient Online Calibration for Autonomous Vehicle's Longitudinal Dynamical System: A Gaussian Model Approach", submitted, *IEEE International Conference on Robotics and Automation*, Oct 2020

2. **Shihao Wang**, and Kris Hauser, "Unified Multi-Contact Fall Mitigation Planning for Humanoids via Contact Transition Tree Optimization", **Best paper finalist**, *IEEE-RAS International Conference on Humanoid Robots*, Beijing, China, 2018

3. **Shihao Wang**, and Kris Hauser, "Realization of a Real-Time Optimal Control Strategy to Stabilize a Falling Humanoid Robot with Hand Contact", *IEEE International Conference on Robotics and Automation*, Brisbane, Australia, 2018

4. **Shihao Wang**, and Kris Hauser, "Real-time stabilization of a falling humanoid robot using hand contact: An optimal control approach", *IEEE-RAS International Conference on Humanoid Robots*, Birmingham, UK, 2017

5. Arundathi M. Sharma, **Shihao Wang**, Yu Meng Zhou, Andy Ruina, "Towards a maximally-robust self-balancing bicycle without reaction-moment gyroscopes or reaction wheels", *Proceedings of the 2016 Bicycle and Motorcycle Dynamics Conference*, Milwaukee, US, 2016