Motion Planning and Control for the Locomotion of Humanoid Robot

by

Yangwei You

Submitted to the Department of Informatics, Bioengineering,
Robotics and Systems Engineering
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Bioengineering and Robotics

at the

UNIVERSITA DEGLI STUDI DI GENOVA
and
ISTITUTO ITALIANO DI TECNOLOGIA

February 2018

Author .....
Department of Informatics, Bioengineering, Robotics and Systems Engineering
February 8, 2018

Certified by .....
Nikos G. Tsagarakis
Doctor of Philosophy
Thesis Supervisor

Accepted by .....
Darwin Caldwell
Chairman, Department Committee on Graduate Theses

Motion Planning and Control for the Locomotion of Humanoid Robot

by

Yangwei You

Submitted to the Department of Informatics, Bioengineering, Robotics and Systems Engineering
on February 8, 2018, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Bioengineering and Robotics

Abstract

This thesis aims to contribute on the motion planning and control problem of the locomotion of humanoid robots. For the motion planning, various methods were proposed in different levels of model dependence. First, a model free approach was proposed which utilizes linear regression to estimate the relationship between foot placement and moving velocity. The data-based feature makes it quite robust to handle modeling error and external disturbance. As a generic control philosophy, it can be applied to various robots with different gaits. To reduce the risk of collecting experimental data of model-free method, based on the simplified linear inverted pendulum model, the classic planning method of model predictive control was explored to optimize CoM trajectory with predefined foot placements or optimize them two together with respect to the ZMP constraint. Along with elaborately designed re-planning algorithm and sparse discretization of trajectories, it is fast enough to run in real time and robust enough to resist external disturbance. Thereafter, nonlinear models are utilized for motion planning by performing forward simulation iteratively following the multiple shooting method. A walking pattern is predefined to fix most of the degrees of the robot, and only one decision variable, foot placement, is left in one motion plane and therefore able to be solved in milliseconds which is sufficient to run in real time. In order to track the planned trajectories and prevent the robot from falling over, diverse control strategies were proposed according to the types of joint actuators. CoM stabilizer was designed for the robots with position-controlled joints while quasi-static Cartesian impedance control and optimization-based full body torque control were implemented for the robots with torque-controlled joints. Various scenarios were set up to demonstrate the feasibility and robustness of the proposed approaches, like walking on uneven terrain, walking with narrow feet or straight leg, push recovery and so on.

Thesis Supervisor: Nikos G. Tsagarakis
Title: Doctor of Philosophy

Acknowledgments

First of all, I would like to thank my tutor Nikos G Tsagarakis. He gave me the opportunity to participate in the WALK-MAN project and continue my research on the locomotion of legged robots. As a tutor, Nikos is very kind to his PhD students. I have a lot of freedom to conduct the research and focus on what I am interested in. Nikos has rich experience on the development of humanoid robot system. He pointed out the main research direction for me at the beginning and shared more inspiring and valuable ideas with me thereafter.

I also really appropriate the help from our locomotion group, especially Zhibin Li. Zhibin gave me detailed and comprehensive instructions during the first two years of my PhD period before he moved to Edinburg. We discussed a lot about the problems I met in the research and he is very willing to share his experience on locomotion control and paper writing. From the other guys in our group, Chengxu Zhou, Przemyslaw Kryczka, Juan Castano and Songyan Xin, I have also learned a lot. We cooperated on the project and paper publication, and really enjoyed the time we spent together. I will always keep this happy time in my mind.

In addition, I also want to thank other people in my lab, like Luca Muratore, Arturo Laurenzi, Jörn Malzahn *et al.* They taught me how to use the whole control framework implemented for the WALK-MAN robot and introduced me more details about the hardware. Without their help, it is difficult for me to advance in research.

Last but not least, I want to thank the support from my family. Their understanding allows me to chase my interest without hesitation and enjoy the three years research life fully.

Contents

1 Introduction **15**

1.1 Motivations and Objectives 16

1.2 State of the Art 16

1.3 Thesis Contributions 20

1.4 Organization of Chapters 21

2 Model Free Planning: Foot Placement Control **23**

2.1 Foot Placement Control Based On Online Linear Regression 24

2.2 One-legged Hopping 28

2.3 Bipedal Running and Walking with Point Foot 31

2.4 Natural Bipedal Walking with Sole 35

2.5 Summary 40

3 Simplified Model Planning: Model Predictive Control **41**

3.1 Simplified Model 41

3.2 Model Predictive Control 43

3.2.1 Predefined Foot Placement 43

3.2.2 Optimized Foot Placement 45

3.2.3 Consider CoM Height Change 47

3.2.4 Details in Re-planning 48

3.3 Simulation 52

3.3.1 Walk on Uneven Terrain 52

3.3.2 Walk with Narrow Feet 55

3.3.3 Push Recovery 56

3.4 Summary 58

4 Nonlinear Model Planning: Multiple Shooting Method **61**

4.1 Nonlinear models . . . 62

4.1.1 Nonlinear Model for Sagittal Dynamics . . . 62

4.1.2 Nonlinear Model for Lateral Dynamics . . . 64

4.1.3 From Nonlinear Models to the Robot . . . 65

4.2 Forward Simulation for Control . . . 65

4.2.1 Accurate Foot Placement Control . . . 66

4.2.2 Mapping From Virtual Legs to Revolute Knees . . . 68

4.3 Simulation . . . 69

4.3.1 Traverse A Stair Blindly . . . 69

4.3.2 Sagittal Push Recovery . . . 71

4.3.3 Lateral Push Recovery . . . 72

4.4 Summary . . . 75

5 Low Level Controller for Stabilization and Motion Tracking **77**

5.1 Position Control and Stabilizer . . . 77

5.2 Quasi-static Cartesian Impedance Control . . . 78

5.2.1 Formulation . . . 79

5.2.2 Walk Like A Inverted Pendulum . . . 79

5.3 Optimization-based Full Body Torque Control . . . 85

5.3.1 Formulation . . . 85

5.3.2 Balance with Arms . . . 89

5.3.3 Walk with Straight Leg . . . 90

5.4 Summary . . . 94

6 Conclusion and Future Work **101**

6.1 Full Body Motion Planning . . . 101

6.2 Stable Torque Control for Motion Tracking . . . 102

6.3 Conclusion . . . 103

A WALK-MAN Robot **107**

B Publication **109**

List of Figures

1-1 Typical bipedal or humanoid robots (From left to right: biped of MIT leg lab, Cassie of Agility company, HRP-2 of AIST, Atlas of Boston Dynamics). 17

2-1 Generic foot placement control from one-legged hopping to bipedal running and walking. 24

2-2 Simplified model of legged robots. 25

2-3 Correlation between forward velocity and foot placement. 26

2-4 Control of one-legged hopping. 28

2-5 Forward velocity comparison of two methods. 30

2-6 Online coefficient adaptation of one-legged hopper. 30

2-7 Adaptive control of one-legged hopping with unknown mass offset. . . 31

2-8 Forward velocity control with mass offset. 31

2-9 Online coefficient adaptation of one-legged hopper with mass offset. . 32

2-10 Comparison between different numbers of sampled data. 32

2-11 Snapshots of bipedal walking. 34

2-12 Forward velocity of bipedal walking. 35

2-13 Online coefficient adaptation of bipedal walking. 35

2-14 Snapshots of bipedal running. 36

2-15 Forward velocity of bipedal running. 36

2-16 Online coefficient adaptation of bipedal running. 36

2-17 Planar humanoid robot with sole and its walking pattern. 37

2-18 Forward velocity tracking and online coefficient adaption of natural walking with sole. 38
2-19 Simulation of walking across uneven terrain. 39

3-1 Simplified model. 42
3-2 Foot prints during turning. 44
3-3 The lower body of WALK-MAN turning in place (time interval 2 seconds). 46
3-4 Push Recovery based on MPC with optimized foot placements(Above: 3D movement; Bottom: ground projection). 48
3-5 The lower body of WALK-MAN walking upwards a stair (time interval 1.5 seconds). 49
3-6 Tanh and deadband function(tanh function: k = 0.6, 0.8, 1.0; dead zone function: threshold = 1.0, 1.5, 2.0). 51
3-7 Trajectories of connecting measured or desired current state with re-planned next state(Above: CoM trajectories, Bottom: ZMP trajectories). 52
3-8 WALK-MAN walking through uneven terrain (interval: 0.8 second). . 53
3-9 Global and local trajectories of WALK-MAN walking through uneven terrain in the forward direction (blue line: CoM, red line: left foot, green line: right foot, solid line: measured, dash line: desired). 54
3-10 Snapshots of WALK-MAN walking with narrow feet (interval: 0.2 second). 55
3-12 Push recovery of WALK-MAN lower body based on MPC re-planning. 57
3-13 Lateral CoM trajectory of WALK-MAN under push recovery. 57
3-11 CoM and foot trajectories of WALK-MAN in the world frame (blue line: CoM, red line: left foot, green line: right foot, solid line: measured, dash line: desired). 59

4-1 Two nonlinear models derived from a full robot model. 63
4-2 Relationship between foot placement and end velocity. 67
4-3 Two robot configurations for touch-down. 67

4-4 Snapshots of crossing a stair. 72
4-5 Angular velocity of virtual support leg when crossing a step. 72
4-6 Upper body posture when crossing a step. 72
4-7 Snapshots of push recovery. 73
4-8 Angular velocity of virtual support leg in push recovery. 73
4-9 Upper body posture in push recovery 73
4-10 COMAN pushed by a lateral force. 74
4-14 Snapshots of dynamic recovery from a series of lateral pushes in ODE (interval 0.25s). 74
4-11 Angular velocity of left virtual leg. 76
4-12 Angular velocity of right virtual leg. 76
4-13 Time consuming of controller. 76

5-1 Snapshots of CENTAURO walking (interval: 0.2 second). 80
5-2 Different motion patterns with narrow feet in lateral plane. 80
5-3 CoM displacement at different feet distances and CoM heights. 82
5-4 CoM impact velocity at different feet distances and CoM heights. . . 83
5-5 State machine of CENTAURO walking pattern. 84
5-8 WALK-MAN used full bodies to keep balancing when a push was exerted on the waist from behind(interval 0.8s). 90
5-6 Foot trajectories of CENTAURO in the local frame of pelvis (red: left foot, green: right foot, purple: contact state, solid: measured, dash-dot: desired). 95
5-7 CoM and foot trajectories of CENTAURO upper body in the world frame (blue: CoM, red: left foot, green: right foot). 96
5-9 Simplified model and ZMP deviation in sagittal plane. 97
5-10 Simplified model and ZMP deviation in lateral plane. 97
5-11 WALK-MAN walking with straight leg (interval: 0.5 second). 97
5-12 CoM and ZMP trajectories when walking straight (blue solid line is measured and red dash-dot line is desired). 98

5-13 CoM trajectory projected in front y-z plane. 99
5-14 Knee position and torque data (blue solid and dash-dot lines are for left and right straight legs,red solid and dash-dot lines are for left and right bent legs). 99

A-1 WALK-MAN body size specifications (all dimensions are in mm) . . . 108

List of Tables

2.1 Parameters of Planar One-legged Robot. 29
2.2 Initial Data for Linear Regression Analysis of One-legged Hopping. . 29
2.3 Parameters of Planar Bipedal Robot. 33
2.4 Initial Data for Linear Regression Analysis of Walking with Sole. . . . 38

4.1 Parameters of Planar Bipedal Robot. 70

Chapter 1

Introduction

Inspired by the high versatility and adaptability of animals on rough terrains, a lot of research has been carried out to develop legged robots which are expected to walk and run in a natural environment[1, 17, 21, 35, 38, 49]. As a typical example of legged robots, humanoid robot is expected to replace people to perform repetitive or hazardous works, who has the potential of good compatibility with already existing artificial tools and human-living environment compared with other robots. Whereas, one principal foundation to achieve these is to realize agile, efficient, robust locomotion, which has been considered as a challenging problem for a long time.

Hence, this thesis is trying to contribute on the humanoid locomotion problem, focusing on the motion planning and control based on the WALK-MAN project, which is a 4 years integrated project funded by the European Commission through the call FP7-ICT-2013-10. The project has the goal to develop a robotic platform (of an anthropomorphic form) which can operate outside the laboratory space in unstructured environments and work spaces as a result of natural and man-made disasters. Different planning technologies were explored, including model free method based on linear regression, model predictive control (MPC) based on simplified model, and multiple shooting method based on nonlinear model. According to the types of joint actuators, position controlled stabilizer, impedance control, and optimization-based full body torque control are implemented to realize stable tracking of the planned trajectories. Various scenarios are set up to verify the proposed approaches, like walking

on uneven terrain, walking with narrow feet or straight leg, push recovery and so on.

1.1 Motivations and Objectives

According to the requirements of WALK-MAN project, a humanoid robot platform is expected to be developed and able to operate outside the laboratory space in unstructured environments. With respect to this target, the research about bipedal locomotion is carried out in this thesis and aims to achieve human-like, robust, efficient and agile walking.

The primary objective of this thesis is to achieve robust walking, which means the robot should be able to automatically handle external disturbances during walking. Typical examples are that it should be capable of making full use of ankle torques to overcome uneven terrain and adjusting foot placements online to resist external pushes.

The further goal is to realize efficient walking. This target can be achieved from different directions, such as delicately designed actuators [30], smart structure design [20] *etc.* Here this thesis endeavors on the realization of the straight-leg walking gait, which is much more efficient than the bent-leg walking gait dominating the locomotion of current humanoid robots nowadays.

This thesis focused on the motion planning and control of dynamic walking of humanoid robots while more agile locomotion types, such as running and jumping, are left for future work.

1.2 State of the Art

To date, many humanoid robots have been built all over the world with impressive progresses in hardware and control[11, 27, 34, 35, 46]. Bipedal gait generation and control for humanoid robots has been extensively studied in recent decades and some remarkable control frameworks have been proposed and widely implemented on different bipedal robotic platforms[8, 14, 18, 25, 31, 51].

(a) Raibert's biped (b) Cassie (c) HRP-2 (d) Atlas

Figure 1-1: Typical bipedal or humanoid robots (From left to right: biped of MIT leg lab, Cassie of Agility company, HRP-2 of AIST, Atlas of Boston Dynamics).

Inspired by biological research that the movement of legged animals can be represented very well by a Spring Loaded Inverted Pendulum (SLIP) model [16], Raibert developed some extremely dynamic legged robots, including one-legged hopper, biped (see Fig. 1-1 (a)) and quadruped [51]. Those robots shared a simple yet effective control philosophy, named three-part controller, in which the pelvis attitude is stabilized by the torque applied on the hip joint and the height is regulated by the thrust energy of support leg injected into the system during the stance phase, and the forward velocity is modulated by adjusting the foot placement during the flight phase. The foot placement is decided by a simple linear equation according to the current and desired forward velocities of these robots. Recently, Hubicki *et al.* successfully developed tether-free 3D-capable bipedal robots ATRIAS and Cassie (see Fig. 1-1 (b)) based on similar ideas [20]. One advantage of this type of heuristic methods is that it does not rely on model which gives it more freedom to design locomotion gaits. The price for this benefit is that it usually involves a lot of manual parameter tunning.

Another main direction in locomotion research of humanoid robots is to perform motion planning based on models. One of the most popular models for bipedal walking is the Linear Inverted Pendulum Model (LIPM) [26], which can provide analytic solution by constraining the movement of center of mass (CoM) in a plane and therefore needs very little computational resource. It usually integrates with preview control or model predictive control to optimize CoM trajectory with respect

to dynamic constraints, such as Zero Moment Point (ZMP) should be always inside the support polygon. This method is fast enough to run in real time and has been implemented on many humanoid robots to follow the operator's walking command on the fly. Typical examples are the HRP-2 robot (Fig. 1-1 (c)) located in the CNRS lab [2] and the Atlas robots (Fig. 1-1 (d)) used by CMU, MIT and IHMC teams in the DARPA challenge [12, 23, 33].

Although LIPM as a simplified model has been popular for a long time due to its easiness of implementation and fast computation, some important dynamic characteristics are missing there and different complements haven been developed to improve it further. For example, the linear inverted pendulum plus flywheel model was proposed to consider the angular momentum of the whole system [48]. And the assumption of CoM staying in a plane also constrains the potential movement ability that the robots can achieve and results unnatural and inefficient bent leg walking. To address this, vast studies have taken CoM height variation into consideration for planning in order to generate more human-like walking motion. Howerver, introducing CoM vertical motion leads to the nonlinear ZMP constraints. Different approaches have been proposed to resolve the nonlinearity. One way is to define the vertical motion beforehand, then the ZMP constraints will be still linear and could be solved via linear approaches. Limiting CoM to a sculptured surface, CoM trajectory can be uniquely defined along the surface satisfying the ZMP constraint[42]. Given CoM vertical oscillation, analytic solution is proposed to cooperate the vertical motion with horizontal ones [59]. Li *et al.* [36] proposed virtual spring-damper model to generate the vertical CoM motion which is independent from the horizontal motions. Englsberger *et al.* realized 3D walking based on the divergent component of motion[9]. Particularly, the humanoid WABIAN [43] could perform knee stretching walking by predefining the trajectory of support-leg's knee joint, and it could also realize heel-contact and toe-off motions with specially designed passive toe joints.

To handle complicated tasks involving multiple bodies of the robot inherently, some researchers have turned to nonlinear optimization for motion planning. Sequential Quadratic Programming is exploited to naturally resolve contact constraint forces

while simultaneously optimizing a trajectory that satisfies the complementarity constraints [47]. Recently, Dai *et al.* transformed the whole body motion planning of humanoid robot to a nonlinear optimization problem and obtained impressive results where the robot can complete really complex tasks such as passing monkey bars [6]. Nevertheless, these nonlinear numerical techniques are usually computer-intensive and have to plan the trajectories off-line as well.

Another crucial advance in dynamic locomotion of legged robots is the availability of powerful and accurate torque actuators for the joints and much faster computers which allow full body optimization to run in real-time. Before the emergence of torque-controlled humanoid robots, various stabilizers are proposed to improve the walking stability for position-controlled bipedal robots. The stabilization approach in [28] applied the full state feedback control to track the COM (position, velocity, acceleration) based on the Linear Inverted Pendulum Model (LIPM), where the gains were designed using the best COM/ZMP regulator [57]. Fukuda et. al explored the genetic algorithms to train the recurrent neural networks for generating the references for actuators based on the Center of Pressure (CoP) feedback in soles [15]. These methods can improve the walking stability a bit, however the stiff joints of position-controlled robots will cause big impact at landing and make them very sensitive to the modeling error of the ground. This situation forces researchers to develop compliant torque controlled robots.

For torque controlled robots, walking is usually considered as a multi-task motion. It involves Cartesian trajectory tracking, body posture regulation and dynamic stability maintaining. While dealing with multiple tasks, traditional null-space projection based techniques could be applied to solve the problem in a hierarchical manner [55] [54]. But these analytical techniques can not properly handle inequality constraints, such as torque limit and friction cone limit. Researchers turn to numerical method which is better at considering different constraints. Although detailed formulations differ, most of approaches formulate the floating base inverse dynamics as a quadratic programming problem with equality and inequality constraints [7, 10, 19, 22, 62]. There are mainly three types of structures to formulate the quadratic programming

problem. One is to do trade-off between different tasks by assigning corresponding weight to each task in the objective function while another one is to utilize priority to clarify the importance of these tasks, named cascade structure. The third one is called hierarchical structure, a combination of the two structures which set tasks into distinct layers through priorities and use weight to balance the tasks in the same layer. Compared with using strict priorities, weighting the tasks is softer and therefore numerically robust during optimization [13]. Hierarchical quadratic programming is indeed more natural to specify conflict tasks in a prioritized hierarchy, however, holds the possibility of chatter as tasks switch between being marginally feasible and infeasible [31].

1.3 Thesis Contributions

This thesis aims to address the problem of locomotion of humanoid robots. With regard to the core issues lying inside the humanoid locomotion, like motion planning, control and some challenging applications, the main contributions of this thesis can be concluded as below:

- Extend Raibert's three-part controller to make full use of the benefit of model-free feature while alleviating the parameter tunning issue and improving its adaptiveness to model error and external disturbance at the same time.

- Implement the model predictive control on the CoM and foot trajectory planning for the humanoid robots and drive its potential to some challenging applications like narrow feet walking and push recovery.

- Explore the possibility of utilizing nonlinear model on motion planning in real-time. Two nonlinear models are proposed individually for the sagittal and lateral planes, and robust walking is demonstrated by applying the multiple shooting method to these models.

- Different control approaches are investigated and discussed for the planned trajectory tracking and stabilization according to the specific types of joint actu-

ators. Optimization-based full body torque control framework is built to deal with multiple tasks following their importance simultaneously.

1.4 Organization of Chapters

Following the introduction, the content of the thesis can be divided into two main parts. The first part gives the detailed description of the motion planning technologies implemented in this thesis, ranging from Chapter 2 to Chapter 4.

In Chapter 2, a novel foot placement control approach was proposed based on the online linear regression analysis as an extension of Raibert's three-part controller, which is model free, and has the capability of adjusting the control coefficients automatically and responding to external disturbances promptly. In addition, it is insensitive to modeling error and able to track the desired forward velocity accurately. As a general approach, it is successfully implemented on different robots with different gaits, from one-legged hopping to bipedal running and walking.

Chapter 3 simplified the whole robot to a point-mass model and formulated the motion planning of this simplified model as a quadratic programming problem through model predictive control. The method is able to optimize CoM trajectory given predefined foot placements or optimize the CoM trajectory and foot placements together. The optimization can be solved fast enough to run in real time with sparse discretization on the trajectories while good enough to guarantee walking stability. In order to resist external disturbance, the details of online re-planning is studied, such as the triggering conditions of re-planning and the trajectory transition after re-planning. The feasibility and robustness of the MPC planner is demonstrated by the simulations performed on the lower body of WALK-MAN robot.

The forward model concept inspired by the sensorimotor control realm is explored in Chapter 4 and applied to the control of robust and dynamic bipedal walking. In this chapter, good candidates of nonlinear models are studied for sagittal and lateral planes to achieve minimum computation without downgrading much of the accuracy. Multiple shooting method is adopted to best exploit forward simulation

and automatically search for precise foot placement via gradient descent. With the proposed method, a simulated COMAN sized robot was able to blindly travel across a stair and withstand external push or continuous ball impact attacks without falling over.

The second main part of this thesis is Chapter 5, focusing on the motion tracking and stabilization, which can be considered as low level control part compared with the first planning part. Different control approaches are investigated and discussed for the planned trajectory tracking and stabilization according to the specific types of joint actuators. CoM stabilizer is designed for the robots with position-controlled joints while impedance control is explored to achieve compliant contact with the ground for torque-controlled robots. Optimization-based full body torque control framework is built to deal with multiple tasks following their importance at the same time, which is supported by a typical example, where the WALK-MAN robot made full use of its full body, especially the arms, to keep balancing.

In Chapter 6, the whole thesis ends with a conclusion summarizing all the accomplished work and some challenging problems of humanoid robots left for further investigation in the future.

Chapter 2

Model Free Planning: Foot Placement Control

One of the most important control actions for the locomotion of legged robots is to know where to place the foot [61], which affects the stability of the locomotion remarkably. Some works have been done to study the relationship between the foot placement and the system stability [63],[61], in which foot placement indicator and estimator are proposed as a measurement of balance. In very early age, Raibert has developed some amazing legged robots, including one leg hopper, bipedal and quadruped, by regulating the foot placement[51]. Similar methods have been applied to many other robots successfully [50], [37]. To further improve the performance of this simple yet effective method, tabular control [52] and optimal control [58] have been tried to combine with the existing method to deal with more complicated situations. Whereas, both of them need large quantities of experimental data to train the tabular or neural network insider their controller, which results slow response to the change of internal system or external environment.

To address this problem, a novel foot placement control approach was proposed here based on the online linear regression analysis, which requires much less empirical data and has the capability to response to external disturbances promptly. In addition, it is able to track the desired forward velocity accurately, insensitive to modeling error like incorrect CoM position, and compatible with different motion patterns, no

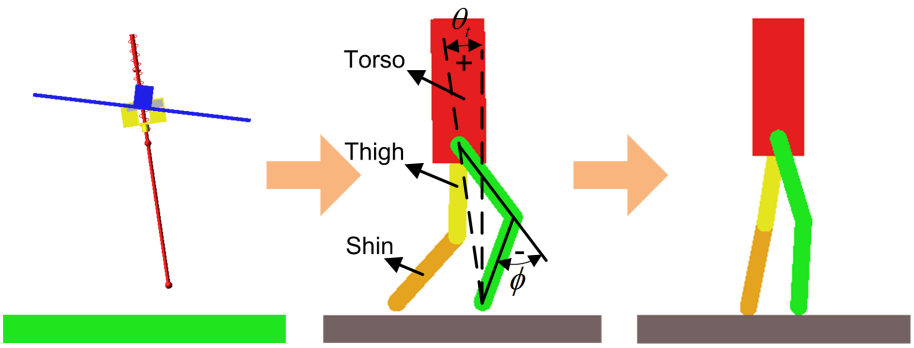


Figure 2-1: Generic foot placement control from one-legged hopping to bipedal running and walking.

matter LIPM, SLIP or others. High adaptation is demonstrated by the successful implementation from one-legged hopping to bipedal running and walking as shown in Fig. 2-1.

This Chapter is organized as follows. In Section 2.1, the proposed foot placement control algorithm is introduced. Thereafter, in Section 2.2, the method is applied to a one-legged hopper in simulation, and compares with Raibert's method. And then it is extended to a bipedal robot for running and walking in Section 2.3. The chapter ends with a summary.

2.1 Foot Placement Control Based On Online Linear Regression

One of the most successful control methods for legged robots is the three-part controller proposed by Raibert, in which the body attitude is controlled by the torque applied on the hip joint during the stance, the body height is regulated by the thrust energy injected into the system, and the forward velocity is modulated by adjusting the foot placement [51]. To warrant the overall performance of these controllers, it is very critical to design the foot placement control algorithm carefully. To investigate the relationship between the foot placement and the forward velocity, a spring loaded inverted pendulum model (SLIP) can be utilized for primary analysis which closely

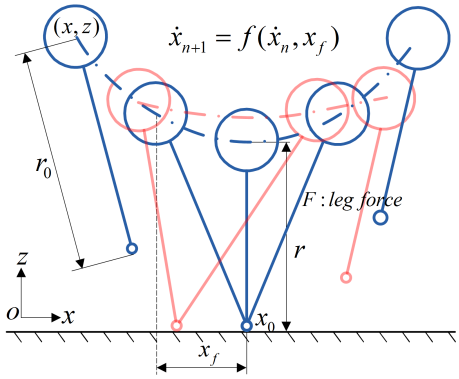


Figure 2-2: Simplified model of legged robots.

resembles the locomotion of legged animals.

The SLIP model contains a point mass located on a spring as Fig. 2-2. In the figure, the spring is replaced by a force, which is more generic. Its dynamic equation can be written as (2.1), where m, k, r_0, r and g are separately the mass, spring stiffness, free length and compression length of spring and gravity constant. x_f is the distance between the end of leg and hip joint along the horizontal direction at the first instant of the touch-down, which represents the foot placement. x, and z are the position of the point mass along the horizontal and vertical directions while x_0 and z_0 are the ones of the end of leg.

$$\begin{aligned} &\text{Flight}: \ \ddot{x} = 0; \ddot{z} = -g; \\ &\text{Stance}: \ \ddot{x} = k(x - x_0)(r_0 - r)/(mr); \\ &\qquad\qquad \ddot{z} = k(z - z_0)(r_0 - r)/(mr) - g; \\ &\qquad\qquad r = \sqrt{(x - x_0)^2 + (z - z_0)^2}. \end{aligned} \tag{2.1}$$

Suppose the total mechanical energy is conserved, let the SLIP model hop one step with different initial forward velocities $\dot{x}$ and foot placements x_f. The relationship of foot placement, forward velocity and its change after one step can be figured out through the multiple numerical simulations as shown in Fig. 2-3.

By observing the symmetry of running and the influence of foot placement on the forward velocity, Raibert came up with a simple algorithm to control forward velocity

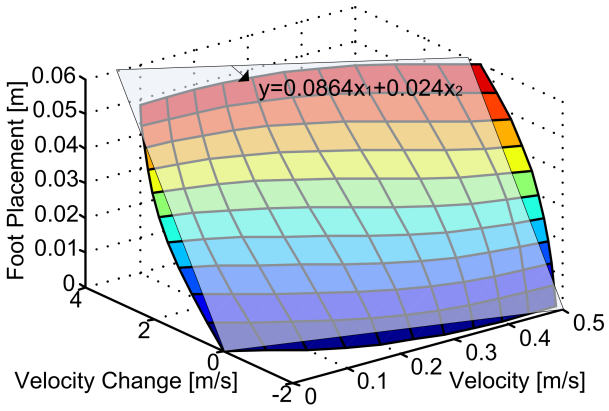


Figure 2-3: Correlation between forward velocity and foot placement.

by the foot placement:

$$x_f = \frac{\dot{x}T_s}{2} + k_{\dot{x}}(\dot{x} - \dot{x}_d), \tag{2.2}$$

where x_f, T_s, $\dot{x}$ and $\dot{x}_d$ are separately the foot placement, duration of stance phase, current and desired forward velocity. The item $\frac{\dot{x}T_s}{2}$ is the neutral point where the movement is symmetric and forward speed remains unchanged after one step. The item $k_{\dot{x}}(\dot{x} - \dot{x}_d)$ is a feedback to displace the foot from the neutral point to stabilize the forward speed against errors and external disturbances, and to change from one forward speed to another. More details can be found in the work of Raibert [51].

From another perspective, Raibert's equation (2.2) can be considered as a flat plane representing the relationship of foot placement, forward velocity and its change as shown in Fig. 2-3. By choosing appropriate coefficients, i.e., the duration of the stance phase T_s and the gain $k_{\dot{x}}$, the plane can fit well into the almost linear surface of Fig. 2-3. But the manual tunning required by Raibert's method is quite time consuming and hard to guarantee a high control accuracy for all different speeds.

To improve this control paradigm, here the states of robots are recorded to form a flat plane by linear regress to represent the local region of the nonlinear surface instead of the entire surface. Obviously, it promises more accurate and robust control performance by updating the local flat plane continuously. In other words, the nonlinear surface can be represented better by a set of local flat plane than one plane

with a constant global linearization.

Treating the foot placement as dependent variable, forward velocity and its change as independent variables, the linear regression aiming to figure out their correlation can be formulated as:

$$
\begin{aligned}
x_f &= k_0 + k_1\dot{x} + k_2(\dot{x} - \dot{x}_d) = k_0 + k_1\dot{x} + k_2\Delta\dot{x} \\
\begin{bmatrix} k_0 \\ k_1 \\ k_2 \end{bmatrix} &= [\ \mathbf{I} \quad \dot{\mathbf{X}} \quad \Delta\dot{\mathbf{X}}\ ]^{\dagger}\mathbf{X}_f
\end{aligned}
\tag{2.3}
$$

The linear equation (2.3) with three coefficients (k_0, k_1, k_2) is adopted to describe the relationship and decide the foot placement x_f according to current velocity $\dot{x}$ and desired velocity $\dot{x}_d$. Compared with Raibert's equation (2.2), the coefficient k_0 is added to handle possible modeling errors such as CoM offset or others which would result in asymmetric foot placements. To perform linear regression, a set of corresponding data measured from robot states is collected to fill into the current velocity vector $\dot{\mathbf{X}}$, velocity change vector $\Delta\dot{\mathbf{X}}$ and foot placement vector $\mathbf{X}_f$.

The structure of vectors $\dot{\mathbf{X}}$, $\Delta\dot{\mathbf{X}}$ and $\mathbf{X}_f$ is actually a queue with fix size so that the latest data can always come in to refresh the coefficients of current local plane. On the other side, the data stored in the queue cannot be too close to each other, otherwise the plane would degrade to a point resulting in the loss of stability and sensitivity due to other factors like body height and attitude regulations and ground disturbance.

To address this issue, a filtering algorithm is developed that the latest data would add into the queue only when its distance from all the existing data in the queue is bigger than one predefined threshold. The requirement can be formulated as below:

$$
\begin{aligned}
|\dot{x}^{new} - \dot{x}| &> \dot{x}^{thd}, \quad \forall x \in \dot{\mathbf{X}} \\
|x_f^{new} - x_f| &> x_f^{thd}, \quad \forall x_f \in \mathbf{X}_f \\
|\Delta\dot{x}^{new} - \Delta\dot{x}| &> \Delta\dot{x}^{thd}, \quad \forall \Delta\dot{x} \in \Delta\dot{\mathbf{X}}
\end{aligned}
\tag{2.4}
$$

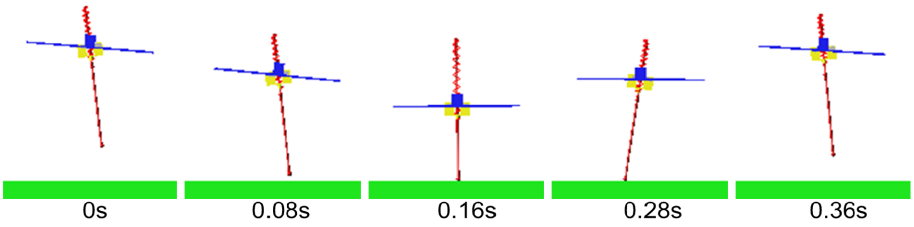


Figure 2-4: Control of one-legged hopping.

where $\dot{x}^{thd}$, x_f^{thd} and $\Delta\dot{x}^{thd}$ are the corresponding threshold. Their value would affect the steady-state accuracy and the robustness of forward velocity tracking. If these thresholds are too big, the data used to estimate the relationship will be too far away from each other to form a precise local linear plane. On the contrary, too small threshold cannot work well to keep the linear relationship stable. Therefore, the thresholds should be chosen carefully to balance between steady-state accuracy and robustness.

In addition, the size n of the queue or the vectors $\dot{\mathbf{X}}$, $\Delta\dot{\mathbf{X}}$, $\mathbf{X}_f$, would also have an influence on the control performance. If n is too big, the estimated plane will refresh very slowly and cannot response promptly to the change of the system. Nevertheless, if n is too small, drastic updating may result in instability.

2.2 One-legged Hopping

To evaluate the performance of the proposed foot placement control, simulations of a planar one-legged robot as Fig. 2-4 are studied in the ADAMS software. The parameters of this robot are listed in TABLE 2.1. It consists of two parts: a body and a telescopic leg. There are one rotary actuator in the hip joint and one spring-damper element in the telescopic leg which can be preloaded. The body attitude is regulated by the hip torque and the hopping height is controlled by the preload in the flight and the release of spring in the stance phase. In this study, the preload is constant and the hopping height will converge to different stable states corresponding to different forward velocities.

The queue size is set as 6 for the linear regression. And the thresholds $\dot{x}^{thd}$,

Table 2.1: Parameters of Planar One-legged Robot.

Parameter	**Value**	**Parameter**	**Value**
Mass of Body	0.4(kg)	Spring Stiffness	80(N/m)
Inertia of Body	1.75e-3(kg m^2)	Spring Damping	0.8(Ns/m)
Mass of Leg	7.8e-3(kg)	Length of Leg	0.3(m)
Inertia of Leg	8.23e-5(kg m^2)	Spring Preload	0.07(m)

Table 2.2: Initial Data for Linear Regression Analysis of One-legged Hopping.

Num.	$\dot{x}$	$\Delta\dot{x}$	x_f	Num.	$\dot{x}$	$\Delta\dot{x}$	x_f
1	0.000	1.000	0.030	4	0.000	1.000	0.030
2	1.000	0.000	0.092	5	1.000	0.000	0.092
3	0.000	0.000	0.000	6	0.000	0.000	0.000

x_f^{thd}, and $\Delta\dot{x}^{thd}$ are 0.05 m/s, 0.03 m, and 0.05 m/s respectively. To compare with Raibert's method, the best control coefficients are tuned for this controller as in (2.5). The initial data chosen for linear regression analysis are listed in TABLE 2.2 to ensure that two methods have the same control coefficients at the beginning.

$$x_f = \frac{\dot{x}T_s}{2} + k_{\dot{x}}(\dot{x} - \dot{x}_d) = 0.092\dot{x} + 0.03(\dot{x} - \dot{x}_d) \tag{2.5}$$

The case study targets at controlling the robot to hop at three speeds, 0.1 m/s, 0.3m/s and 0.5m/s. The simulation results of the two methods are shown in Fig. 2-5. Raibert's method can keep a small stead-state error at the speed of 0.1 m/s while not at another two speeds. The proposed method can update the coefficients as in (2.3) based on the collected data during hopping and keep a very small stead-state error for all the three speeds. Fig. 2-6 shows the update of the coefficients to eliminate the steady-state error. The proposed foot placement control method can guarantee high tracking accuracy and represent better the local relationship of foot placement, forward velocity and its change, compared with the fixed one provided by Raibert's method.

As mentioned in Section 2.1, the coefficient k_0 is added to eliminate the possible modeling error such as unknown mass offset. To validate this, the CoM of body

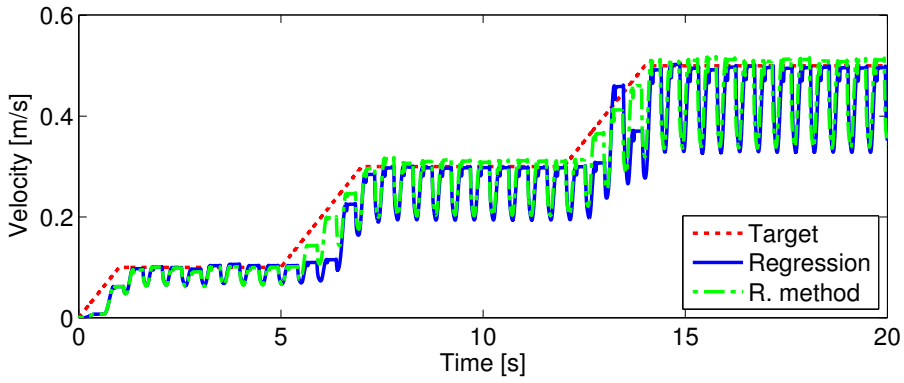


Figure 2-5: Forward velocity comparison of two methods.

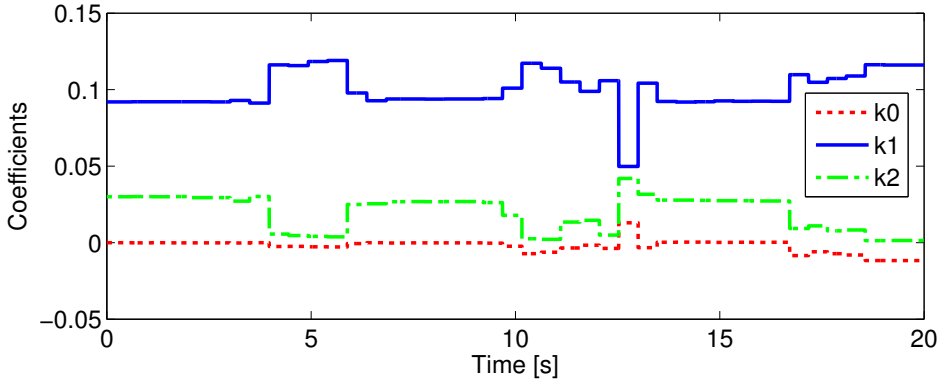


Figure 2-6: Online coefficient adaptation of one-legged hopper.

is offset as shown in Fig. 2-7. The control coefficients and the initial data for the two methods are the same as above. The robot is expected to hop at the speed of 0.5 m/s after 5 seconds of transition. The forward velocity tracking result in Fig. 2-8 indicates that when the unknown mass offset exists, the steady-state error becomes so significant for Raibert's method that the direction of travel is inevitably reversed. However, with the proposed approach, the robot hops first backward for some steps to prevent falling due to the unknown mass offset, then adapts to the foot placement estimation and recovers its hopping direction in the end. The forward velocity converges to the desired one after a while. This process is also shown in Fig. 2-7. The online update of coefficients is shown in Fig. 2-9. The coefficient k_0 is not zero any more when the mass offset exists. The simulation result proves that the new foot placement control algorithm is more robust and adaptive to modeling error.

In Section 2.1, the effect of the queue size n on the online regression is also

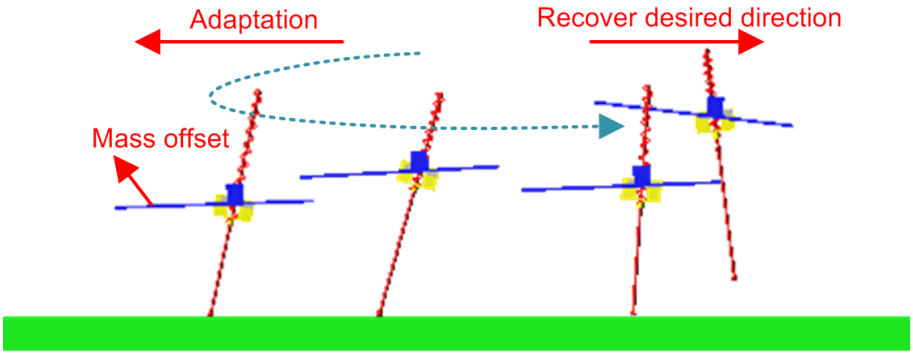


Figure 2-7: Adaptive control of one-legged hopping with unknown mass offset.

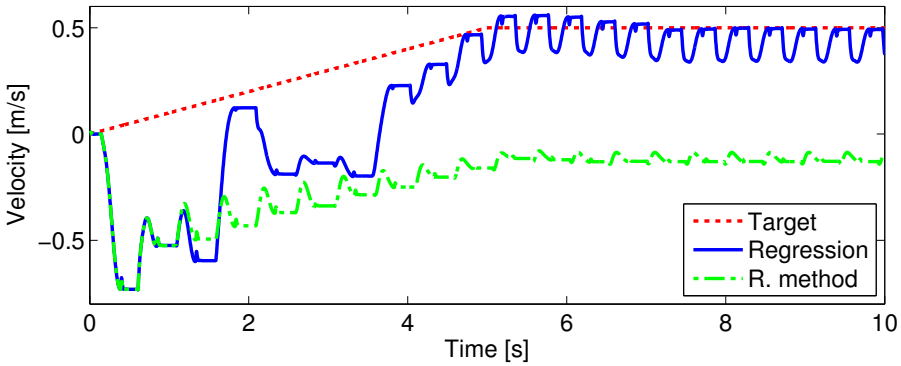


Figure 2-8: Forward velocity control with mass offset.

discussed. To confirm it, two simulations are performed on the mass offset robot using queue sizes 6 and 12 individually. The different responses to the unknown mass offset is shown in Fig. 2-10. Obviously, the result supports well the aforementioned assumption that forward velocity with bigger queue size fluctuates less but takes more time to reach the desired state.

2.3 Bipedal Running and Walking with Point Foot

The key point of the foot placement control with online linear regression is that the body attitude and height are controlled by independent controllers so that the foot placement can have an almost linear relationship with forward velocity and its change which can be represented well by a set of estimated local linear plane. This can provide more freedom to explore the diversity of the leg motion instead of behaving like a

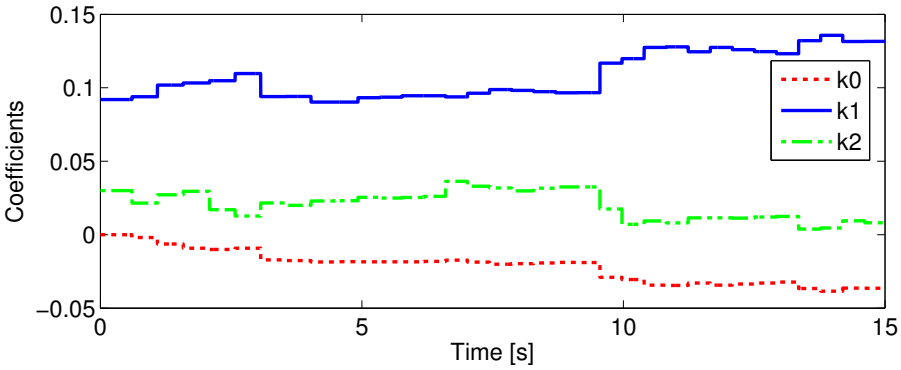


Figure 2-9: Online coefficient adaptation of one-legged hopper with mass offset.

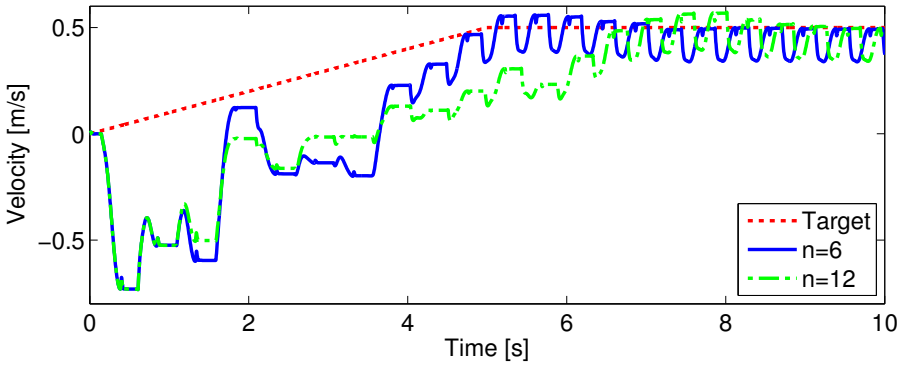


Figure 2-10: Comparison between different numbers of sampled data.

spring-damper. It can be applied to not only one-legged hoppers but also bipedal robots, for both running and walking.

To verify this, a bipedal robot is built in simulation. Its mechanical parameters are listed in TABLE 2.3. ϕ and θ_t marked in Fig. 2-1 are the knee angle and touch-down angle. Touch-down angle θ_t is defined as the angle between the vertical line and the line crossing the hip joint and the end of stance leg at the first moment of touching down. Because the swing leg has less influence on the system than the stance one, the discussion about bipedal controller is focused on the stance leg.

The attitude of torso is controlled by the torque in the hip joint. Instead of using a telescopic leg with a spring-damper element as that of the one-legged robot, the bipedal robot controls its height by the knee joint. A three-order polynomial module (2.6) is used to plan the trajectory of the knee joint for both running and walking. After the end of stance, the leg will retract. The equivalent stiffness and damping of

Table 2.3: Parameters of Planar Bipedal Robot.

Part	**Mass(kg)**	**Inertia(kg*m^2)**	**Length(m)**
Torso	19.8	0.3	0.4
Thigh	2.8	0.02	0.25
Shin	2.5	0.02	0.25

the knee joint are 287 Nm/rad and 5.7 Nms/rad.

$$\begin{aligned}\phi(t) &= \phi(0) + \dot{\phi}(0)t \\ &- \frac{3\phi(0) - 3\phi(T) + 2T\dot{\phi}(0) + T\dot{\phi}(T)}{T^2}t^2 \\ &+ \frac{2\phi(0) - 2\phi(T) + T\dot{\phi}(0) + T\dot{\phi}(T)}{T^3}t^3,\end{aligned} \tag{2.6}$$

where T is the stance time, $\phi(t)$ is the expected angle of knee joint at the time of t after touching down, $\phi(0)$ and $\dot{\phi}(0)$ are the angle and angular speed of knee joint at the first moment of touching down, and $\phi(T)$ and $\dot{\phi}(T)$ are the expected angle and angular speed at the end of the stance phase. For walking, $\phi(T) = 0$ rad, $\dot{\phi}(T) = 0$ rad/s, $T = 0.4$ s; for running, $\phi(T) = -0.4$ rad, $\dot{\phi}(T) = 0$ rad, $T = 0.3$ s. $\phi(0)$ and $\dot{\phi}(0)$ are measured at the first moment of touch-down but $\phi(0)$ will be largely affected by the motion of knee joint when the leg is swinging.

The difference between running and walking for the bipedal robot is the trajectory of the knee joint. In running, $\phi(0)$ is controlled to -1 rad for all speed. But for walking, it is controlled to a value decided by the expected velocity as the equation $\phi(0) = -0.1 - 0.4v_d$. A simple linear equation is used here to make sure that no energy is over injected by the knee motion to prevent the stance leg from lifting off the ground, and there is enough clearance to touch down with an expected touch angle. The stance time of running is a little shorter than walking, and the knee angle of running is changing in a smaller range from -1 rad to -0.4 rad. Both are designed to ensure that the robot can lift off the ground after stance which is necessary for running. The trajectories of knee joint mentioned above are the simple solutions for walking and running, yet effective while working with the proposed foot placement

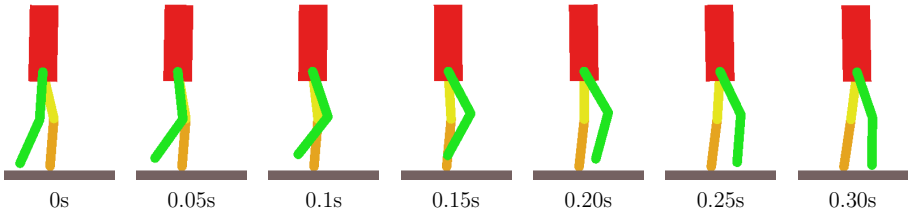


Figure 2-11: Snapshots of bipedal walking.

control algorithm.

For linear regression analysis, the touch-down angle θ_t represents the foot placement as a dependent variable, and forward velocity of hip joint and its change as independent variables. The distance of stance leg projected on the ground also can be used to represent the foot placement as in the one-legged robot. But touch-down angle is easier to calculate for the bipedal robot and it will be shown that our algorithm is not sensitive to which variable is used as long as the nearly linear relationship exists.

Linear regression is used to estimate the equation (2.7), in which θ_t is touch-down angle, $\dot{x}$ is the velocity of hip joint before touch-down and $\dot{x}_d$ is the desired velocity at the first moment of touching down of next step. k_0, k_1, k_2 are the coefficients calculated from a queue with size 6. The thresholds $\dot{x}^{th}$, θ_t^{th}, $\Delta\dot{x}^{th}$ to selectively filter data are 0.05 m/s, 0.02 rad and 0.05 m/s.

$$\theta_t = k_0 + k_1\dot{x} + k_2(\dot{x} - \dot{x}_d) \tag{2.7}$$

First, the bipedal robot is controlled to walk at 0.5 m/s after 5 seconds of transition. Fig. 2-11 is the snapshot of successful walking. The walking velocity is tracked accurately and stably. The coefficients also converge to stable values when the velocity is stable. The walking velocity and online refreshing of the coefficients are shown in the Fig. 2-12 and Fig. 2-13.

Further, the bipedal robot is controlled to run at 1 m/s after 5 seconds of transition. Fig. 2-14 is the snapshots of stable running. With the same control algorithm, the robot can reach the velocity accurately and stably, although there are some ve-

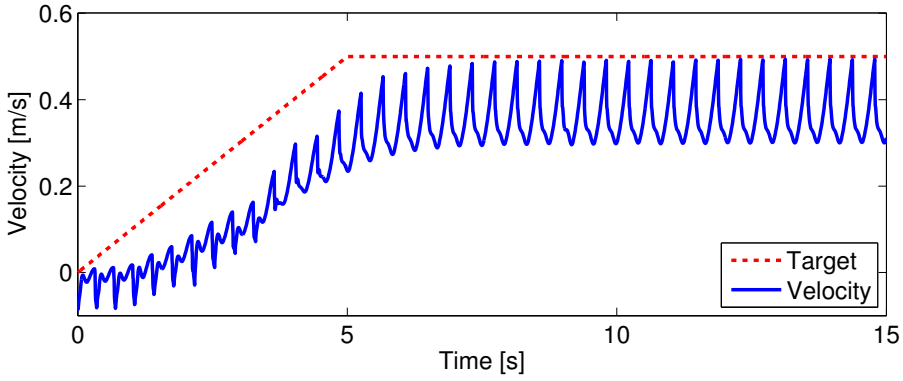


Figure 2-12: Forward velocity of bipedal walking.

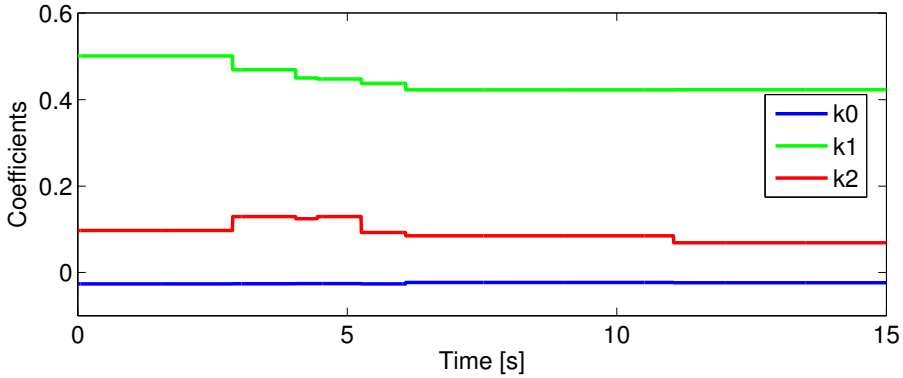


Figure 2-13: Online coefficient adaptation of bipedal walking.

locity spikes in each step due to the large impacts between the leg and the ground during the touch-down. The coefficients also converge to stable values when the velocity stabilizes. The running velocity and the online update of the coefficients can be seen in the Fig. 2-15 and Fig. 2-16.

It should be noted that the leg length of the simulated biped is 0.5m, so the running speed of 1 m/s corresponds to a normalized speed of 2 leg/s. In other words, a speed of 2 m/s for a human adult, which is similar to an average speed of jogging.

2.4 Natural Bipedal Walking with Sole

Following the same idea, to enable more natural walking, the basic patterns of stance and swing legs are designed with straight leg and heel-strike toe-off features first, then the foot placement control can be implemented to adjust the specific trajectories for

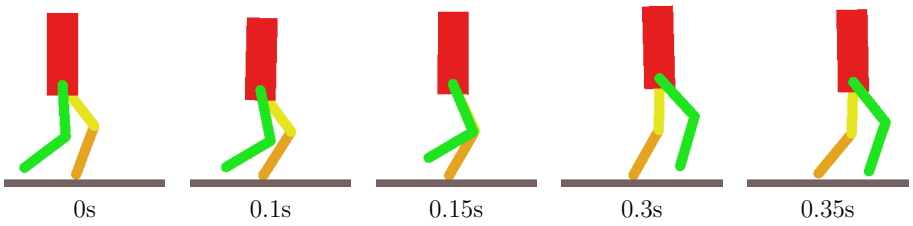


Figure 2-14: Snapshots of bipedal running.

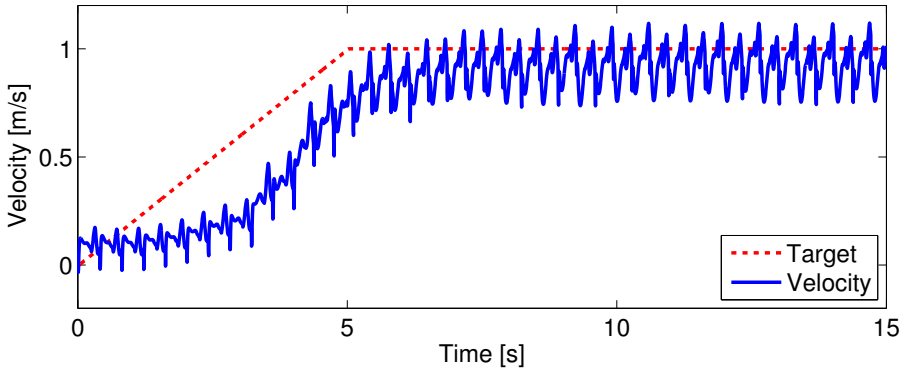


Figure 2-15: Forward velocity of bipedal running.

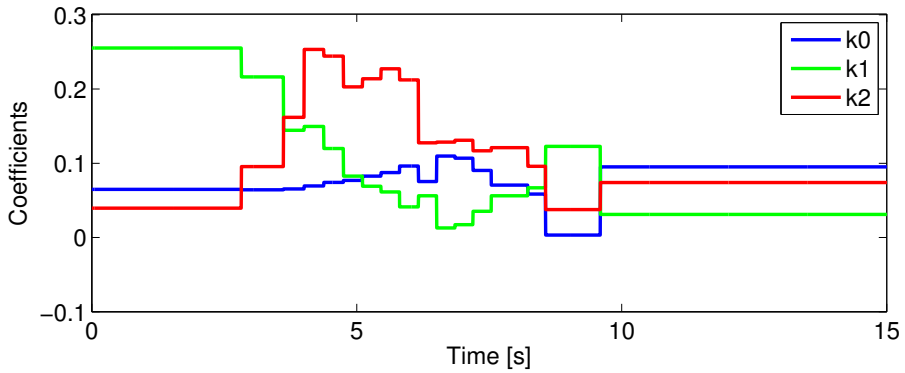


Figure 2-16: Online coefficient adaptation of bipedal running.

each step and keep the walking stability.

The robot studied here is a simulated planar robot comprised of 7 parts, one torso, two thighs, two shins and two soles, as shown in Fig. 2-17(a). The robot is almost the same with the one in previous section, only has extra soles (mass 0.3 kg, inertia 3.7e-4 kgm^2, length 0.075 m) at the end of the legs. Actuators are mounted in the hip, knee and ankle joints. γ is the pitch angle of torso. θ is the swing angle, which is the angle between the vertical line and the line across the hip and ankle joint. θ_t

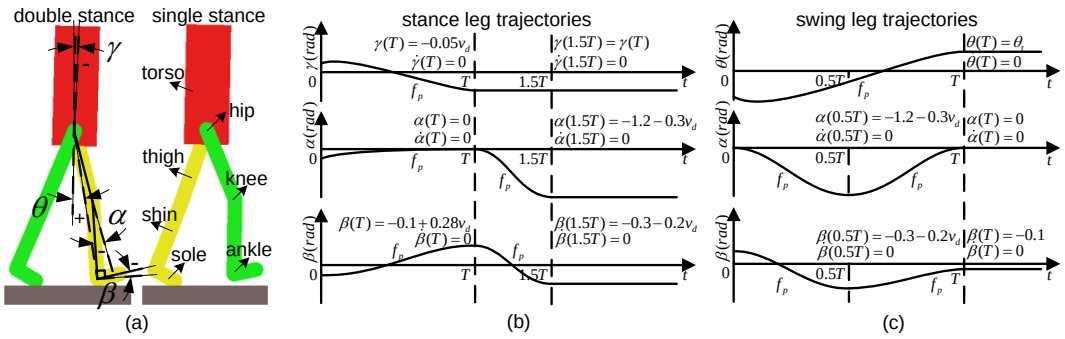


Figure 2-17: Planar humanoid robot with sole and its walking pattern.

is the touch-down angle, i.e., the swing angle at the moment of touch-down. α and β are individually the angles of knee and ankle joints. The direction convention of these angles are defined in Fig. 2-17(a) with '+' and '-' signs. And 3rd-order polynomials are utilized here for trajectory planning.

The touch-down event of walking is triggered when the swing leg touches on the ground and the support leg has stayed for more than half of the expected stance time T. At this moment, the trajectories for the upcoming walking step are planned based on the current state and expected forward velocity, and the support and swing leg swap. Although there is a short period of double support, this transition is not specially planned, but taken into account while planning the trajectories of single support phase. So the trajectory patterns of a leg can be classified according to the walking phases.

During the stance phase, the hip actuator applies torque at the torso to track a planned trajectory of body pitch, which is related to forward velocity in order to imitate the leaning-forward behavior when people are walking. Meanwhile, actuators on the knee and ankle joints are expected to extend the leg to propel. Note that when the actual time of stance phase is longer than desired and the swing leg still does not touch down, the stance leg will actively retract. The expected trajectories of these angles are piecewise three-order polynomials and detailed in Fig. 2-17(b).

During the swing phase, the swing angle is controlled by the hip actuator of swing leg to track the desired touch-down angle. Meanwhile, the leg retracts first to achieve ground clearance, and then extends by the knee and ankle actuators. If the actual

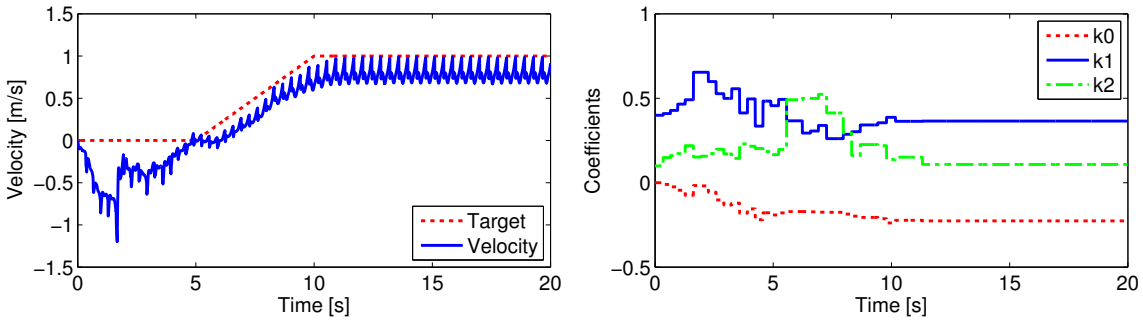


Figure 2-18: Forward velocity tracking and online coefficient adaption of natural walking with sole.

Table 2.4: Initial Data for Linear Regression Analysis of Walking with Sole.

Num.	$\dot{x}$	$\Delta\dot{x}$	θ_t	Num.	$\dot{x}$	$\Delta\dot{x}$	θ_t
1	1.000	0.000	0.400	4	1.000	0.000	0.400
2	0.000	1.000	0.100	5	0.000	1.000	0.100
3	0.000	0.000	0.000	6	0.000	0.000	0.000

swing time exceeds the predefined one and the touch down still does not occur, then knee and ankle joints will remain at the same position until touch-down happens. The swing angle is of particular interest. The swing angle trajectory should lead to the touch-down angle θ_t at the end of this phase, and the touch-down angle θ_t should be updated according to the proposed foot placement control algorithm in Section 2.3. The resulting swing leg trajectories can be seen in Fig. 2-17(c).

The queue size and the thresholds used for the linear regression is the same with the robot with point foot in previous section. The initial data for the linear regression are listed in Table 2.4. They are chosen arbitrarily and cannot promise stable walking without online update.

The robot walked in place for 5 s to adjust the coefficients in (2.3), and then walked at 1 m/s within a transition of 5 s. At the beginning, the velocity fluctuated because the initial data for regression analysis was arbitrarily given. When the coefficients were adjusted well, then the walking velocity was tracked accurately. The online update of the coefficients and walking velocity are shown in the Fig. 2-18. Then, the robot was controlled to walk at 1 m/s to cross an unknown step of 5 cm height. Fig. 2-19(a) shows the snapshots of successful walking over the step. The coefficients

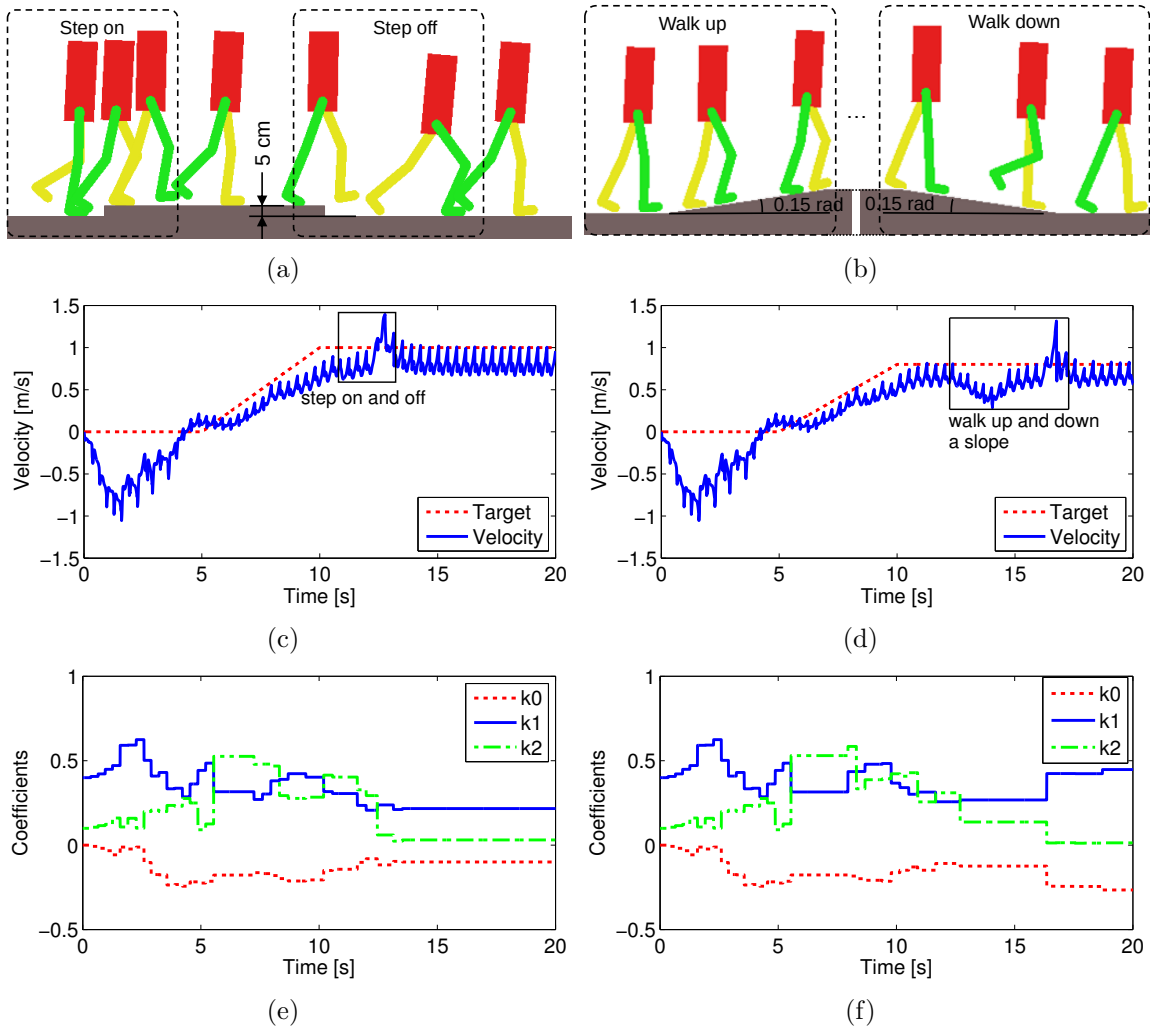


Figure 2-19: Simulation of walking across uneven terrain.

refreshing and walking velocity are shown in the Fig. 2-19(e) and 2-19(c).

Further, the robot was controlled to walk up and down a slope of 0.15 rad at 0.8 m/s. The slope was known for the robot, and the ankle angle $\beta(T)$ was revised according to the slope to ensure a good contact between the sole and slope. In practice, the slope can also be calculated according to the orientation of sole when it was fully in touch with the ground. Figure 2-19(b) is the snapshots of successful walking over the slope. The coefficients refreshing and walking velocity are shown in the Fig. 2-19(f) and 2-19(d).

2.5 Summary

In this chapter, a unified and adaptive foot placement control algorithm is proposed for legged robots based on the online linear regression. The essential is that if the body attitude and height are properly controlled, the foot placement then has a nearly linear relationship with forward velocity and its change, which can be locally represented well by an estimated linear plane.

This control strategy permits more versatile control for the stance leg as long as the nearly linear relationship preserves. With small quantity of measured data, the proposed algorithm is capable of adjusting the control parameters automatically and responding quickly to external disturbances. Its good adaptability and higher control accuracy also outperform the empirical tuning approach. The very same controller is able to produce stable hopping with accurate forward velocity tracking even with unknown mass offset, as well as stable bipedal running and walking. For a bipedal robot with sole, the controller is able to produce very natural walking with straight leg and heel-strke toe-off features with very little revise.

Chapter 3

Simplified Model Planning: Model Predictive Control

Considering the poor generalization ability of model-free methods that new experimental data should be collected for training a new robot which could by risky, Model Predictive Control (MPC) based on a simplified model is exploited in this chapter. MPC is an advanced method of process control that has been in use in the process industries in chemical plants and oil refineries since the 1980s. Recently, more and more applications in the area of robotics motion planning have been observed. The main advantage of MPC is the fact that it allows the current timeslot to be optimized, while keeping future timeslots in account. It is a good approach for locomotion to consider the stability in the future and then decide what to do at the current moment. In MPC, the dynamic model of system is utilized to allow the system state to evolve into future. The more complicated the model is, the more computation time is needed. To meet the real-time requirement, simplified models, especially linear models, are preferred compared with the full body nonlinear model.

3.1 Simplified Model

To describe the over-all dynamics of the whole robot without considering the detailed configuration of each joint, a mass-point model of centroidal dynamics is extensively

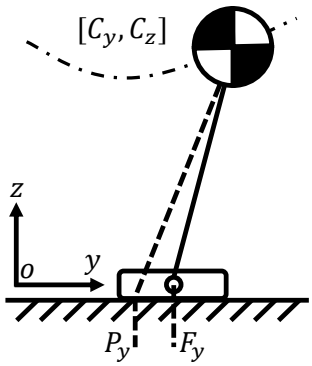


Figure 3-1: Simplified model.

utilized on locomotion planning. Taking the motion in the Y-Z plane (shown in Fig. 3-1) as example, the centroidal dynamics of humanoid robot can be described as:

$$C_y - \frac{mC_z\ddot{C}_y - \dot{L}_x}{m(\ddot{C}_z + g)} = P_y \in conv\{p_i\} \tag{3.1}$$

where C stands for the motion of CoM and P for ZMP, m is the total mass of robot and L is the angular momentum of the whole robot around CoM. g is the gravitational acceleration constant. Those subscripts indicate the motion coordinate. $conv\{p_i\}$ represents the convex hull of contact points.

According to the state of art, there is still no reasonable way to define the desired angular momentum, hence the angular momentum part in the simplified dynamics model is usually not taken into account[45]. Instead, it is preferred to be considered in the low level controller which involves full body dynamics model and will be introduced in Chapter 5. In addition, to make the equation linear, C_z and $\ddot{C}_z$ should be known before hand. And the simplest case is to keep the CoM height constant so that $C_z = h$ and $\ddot{C}_z = 0$. With the assumptions of constant CoM height and no angular momentum, the simplified dynamic model can be rewritten as:

$$C_y - \frac{h}{g}\ddot{C}_y = P_y \in conv\{p_i\} \tag{3.2}$$

The motion in the X-Z plane is the same with the Y-Z plane. So the dynamic

model along all the three directions are be written as below:

$$\begin{bmatrix} \ddot{C}_x \\ \ddot{C}_y \\ \ddot{C}_z \end{bmatrix} = \begin{bmatrix} \frac{g}{h}C_x - \frac{g}{h}P_x \\ \frac{g}{h}C_y - \frac{g}{h}P_y \\ 0 \end{bmatrix}. \tag{3.3}$$

With this simplified linear model, MPC can be implemented in real time and run fast enough to handle external disturbance through online re-planning.

3.2 Model Predictive Control

Based on the simplified model, there are two popular ways to implement model predictive control. One is with foot placements predefined while the other one is to optimize the foot placement and CoM trajectory at the same time. The continuous CoM trajectory is discretized into a sequence of knots to perform optimization. One knot contains the CoM position, velocity and acceleration $[C, \dot{C}, \ddot{C}]$ at one moment. The time period between two adjacent knots is T during which CoM jerk $\dddot{C}$ is constant. So given a starting knot and the optimized jerks along time, the corresponding continuous trajectory can be constructed.

3.2.1 Predefined Foot Placement

For MPC with predefined foot placements, only CoM trajectory needs to be optimized. Since the simplified model assumes the CoM height is constant, then only the movements of X and Y directions need to be concerned:

$$\min_{\dddot{\mathbf{C}}_x^k, \dddot{\mathbf{C}}_y^k} \quad \frac{\alpha}{2}||\dddot{\mathbf{C}}_x^k||^2 + \frac{\alpha}{2}||\dddot{\mathbf{C}}_y^k||^2 + \frac{\gamma}{2}||\mathbf{P}_x^{k+1} - \mathbf{P}_{x\_ref}^{k+1}||^2 + \frac{\gamma}{2}||\mathbf{P}_y^{k+1} - \mathbf{P}_{y\_ref}^{k+1}||^2 \tag{3.4}$$

subject to

$$\begin{aligned} \mathbf{P}_{x\_min}^{k+1} &< \mathbf{P}_x^{k+1} - \mathbf{P}_{x\_ref}^{k+1} < \mathbf{P}_{x\_max}^{k+1} \\ \mathbf{P}_{y\_min}^{k+1} &< \mathbf{P}_y^{k+1} - \mathbf{P}_{y\_ref}^{k+1} < \mathbf{P}_{y\_max}^{k+1} \end{aligned} \tag{3.5}$$

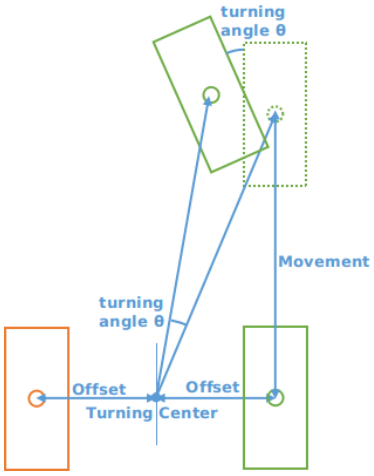


Figure 3-2: Foot prints during turning.

In the above equation, the superscripts k and $k+1$ indicate the knot number, $\dddot{\mathbf{C}}_x^k$ and $\dddot{\mathbf{C}}_y^k$ are the CoM jerks, and $\mathbf{P}_{x\_ref}^{k+1}$, $\mathbf{P}_{y\_ref}^{k+1}$ are the ZMP reference positions lying in the center of the support polygon decided by the predefined foot placements. $\mathbf{P}_{x\_max}^{k+1}$, $\mathbf{P}_{x\_min}^{k+1}\mathbf{P}_{y\_max}^{k+1}$, $\mathbf{P}_{y\_min}^{k+1}$ are the upper and lower limits of support polygon along x and y directions which is also decided by the predefined foot placements. In order to take the future state into consideration in the optimization, the CoM and ZMP variables are vectors containing the current knot and the following knots in a fix-size time window. The superscripts of the CoM and ZMP variables indicate the number of the current knot inside. Additionally, α and γ are the weights to do trade off between the smoothness and stability of movement.

In the optimization problem, the decision variables are the jerks of CoM trajectory along the x and y directions. The objective is to minimize the CoM jerk and the ZMP deviation from the center of polygon. The CoM jerk represents the smoothness of the trajectory. Smaller jerk means a smoother trajectory. And the ZMP deviation indicates the stability of optimized trajectory. The bigger the deviation is, the smaller the stability margin is.

Within this control framework, the turning during walking can also be considered straightforward. To achieve this, the planning of foot prints along X and Y directions should be related to the turning rate as shown in Fig. 3-2.

Take left leg supporting (red foot print) and right leg swinging (right foot print)

as an example. Assuming the robot is expected to turn $\Delta\theta = \theta^{k+1} - \theta^k$ degrees and move M_x^k distance in x direction and M_y^k in y direction, the foot prints of next step can be given:

$$\begin{aligned} \begin{bmatrix} P_{x\_l}^{k+1} \\ P_{y\_l}^{k+1} \end{bmatrix} &= \begin{bmatrix} P_{x\_l}^{k} \\ P_{y\_l}^{k} \end{bmatrix} \\ \begin{bmatrix} P_{x\_r}^{k+1} \\ P_{y\_r}^{k+1} \end{bmatrix} &= \begin{bmatrix} P_{x\_l}^{k} \\ P_{y\_l}^{k} \end{bmatrix} + R(\theta^k) \begin{bmatrix} 0 \\ -L \end{bmatrix} + R(\theta^{k+1})(\begin{bmatrix} M_x^k \\ M_y^k \end{bmatrix} + \begin{bmatrix} 0 \\ -L \end{bmatrix}) \end{aligned} \tag{3.6}$$

where P_{x_l}, P_{y_l} and P_{x_r}, P_{y_r} are the ZMP references or foot prints of left and right legs, and their right superscripts k and $k+1$ indicate the current and next steps. And the two feet are expected to be separated $2L$ from each other. Since the left leg is supporting, its foot print of next step is kept the same with the current step. However, for the right leg, its foot print of next step is determined by the second equation of (3.6). The first two items on the right side can be considered as the turning center while the third item is the turning vector. $R(\theta^k)$ and $R(\theta^{k+1})$ are the planar rotation matrices along z direction. A typical example, the robot turning in place, is shown in the snapshot Fig. 3-3.

3.2.2 Optimized Foot Placement

The potential problem lurking in the MPC given above is that the predefined foot placements are usually given arbitrarily and don't take the dynamics into account which may result infeasible solution. In addition, if the robot is affected by external disturbance, the foot placement defined before hand may not be appropriate any more. In this case, it is better to optimize the foot placement and the CoM trajectory at the same time so that the foot placement can be updated timely and reasonably. Similarly, with foot placements taken as decision variables, the planning can still be

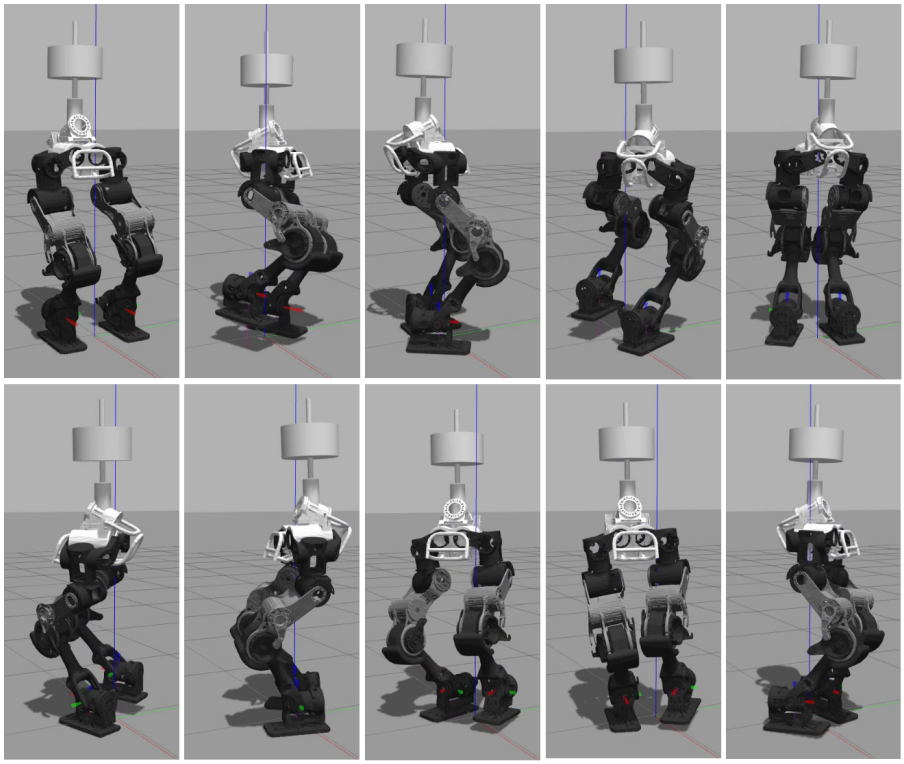


Figure 3-3: The lower body of WALK-MAN turning in place (time interval 2 seconds).

formulated as an optimization problem of MPC type:

$$\begin{aligned}\min_{\dddot{\mathbf{C}}_x^k, \mathbf{F}_x^k, \dddot{\mathbf{C}}_y^k, \mathbf{F}_y^k} \quad & \frac{\alpha}{2}||\dddot{\mathbf{C}}_x^k||^2 + \frac{\alpha}{2}||\dddot{\mathbf{C}}_y^k||^2 + \frac{\beta}{2}||\dot{\mathbf{C}}_x^{k+1} - \dot{\mathbf{C}}_{x\_ref}^{k+1}||^2 + \frac{\beta}{2}||\dot{\mathbf{C}}_y^{k+1} - \dot{\mathbf{C}}_{y\_ref}^{k+1}||^2 \\ & + \frac{\gamma}{2}||\mathbf{P}_x^{k+1} - \mathbf{P}_{x\_ref}^{k+1}||^2 + \frac{\gamma}{2}||\mathbf{P}_y^{k+1} - \mathbf{P}_{y\_ref}^{k+1}||^2 \end{aligned} \tag{3.7}$$

subject to

$$\begin{aligned} \mathbf{P}_{x\_min}^{k+1} < \mathbf{P}_x^{k+1} - \mathbf{P}_{x\_ref}^{k+1} < \mathbf{P}_{x\_max}^{k+1} \\ \mathbf{P}_{y\_min}^{k+1} < \mathbf{P}_y^{k+1} - \mathbf{P}_{y\_ref}^{k+1} < \mathbf{P}_{y\_max}^{k+1} \\ \mathbf{F}_{x\_min}^{k} < \mathbf{F}_x^{k} < \mathbf{F}_{x\_max}^{k} \\ \mathbf{F}_{y\_min}^{k} < \mathbf{F}_y^{k} < \mathbf{F}_{y\_max}^{k} \end{aligned} \tag{3.8}$$

Compared with Equation 3.4, the CoM velocity tracking errors $||\dot{\mathbf{C}}_x^{k+1} - \dot{\mathbf{C}}_{x\_ref}^{k+1}||$ and $||\dot{\mathbf{C}}_y^{k+1} - \dot{\mathbf{C}}_{y\_ref}^{k+1}||$ are also taken as objective to minimize. Since the foot placements $\mathbf{F}_x^k$, $\mathbf{F}_y^k$ are not predefined any more, they are considered as decision variables in addition to CoM jerks $\dddot{\mathbf{C}}_x^k$, $\dddot{\mathbf{C}}_y^k$.

For the constraint, the ZMP should not exceed the boundary of support polygon as said in previous section. Besides, the foot placements $\mathbf{F}_x^k$, $\mathbf{F}_x^k$ should also be inside the reachable area.

With a sparse discretization of the trajectory, like $T = 0.1$ s, the MPC is fast enough to finish all the optimization within 1 ms which should be sufficient to run in real time, so it is able to cope well with the case of push-recovery with the optimized foot placements as Fig. 3-4 shown. The red square is the foot print of left foot, and the blue one is for right foot. The green solid line is CoM trajectory, and ZMP is depicted by orange dot line. The robot is expected to move along x direction with constant height. After push, new foot placements are generated, the CoM trajectory is not divergent, and the ZMP always stays inside the support polygon which means that the robot is able to keep walking with falling .

3.2.3 Consider CoM Height Change

No matter for predefined foot placement or optimized foot placement, the CoM height is always assumed to be constant by now. Actually, it is possible to take predefined CoM height change into account without breaking the linearity. Take a closer look at the equation 3.1. Without considering the angular momentum, a linear equation can still be achieved:

$$C_y - \frac{C_z}{\ddot{C}_z + g}\ddot{C}_y = C_y - K\ddot{C}_y = P_y \in conv\{p_i\} \tag{3.9}$$

where $\frac{C_z}{\ddot{C}_z+g}$ can be treated as a constant K when C_z and $\ddot{C}_z$ are already given from the predefined CoM heihgt. Integrating this into the MPC control framework, it is straightforward to make the robot walk with varying CoM height. Fig. 3-5 shows the snapshot of the lower body of WALK-MAN walking upwards a stair.

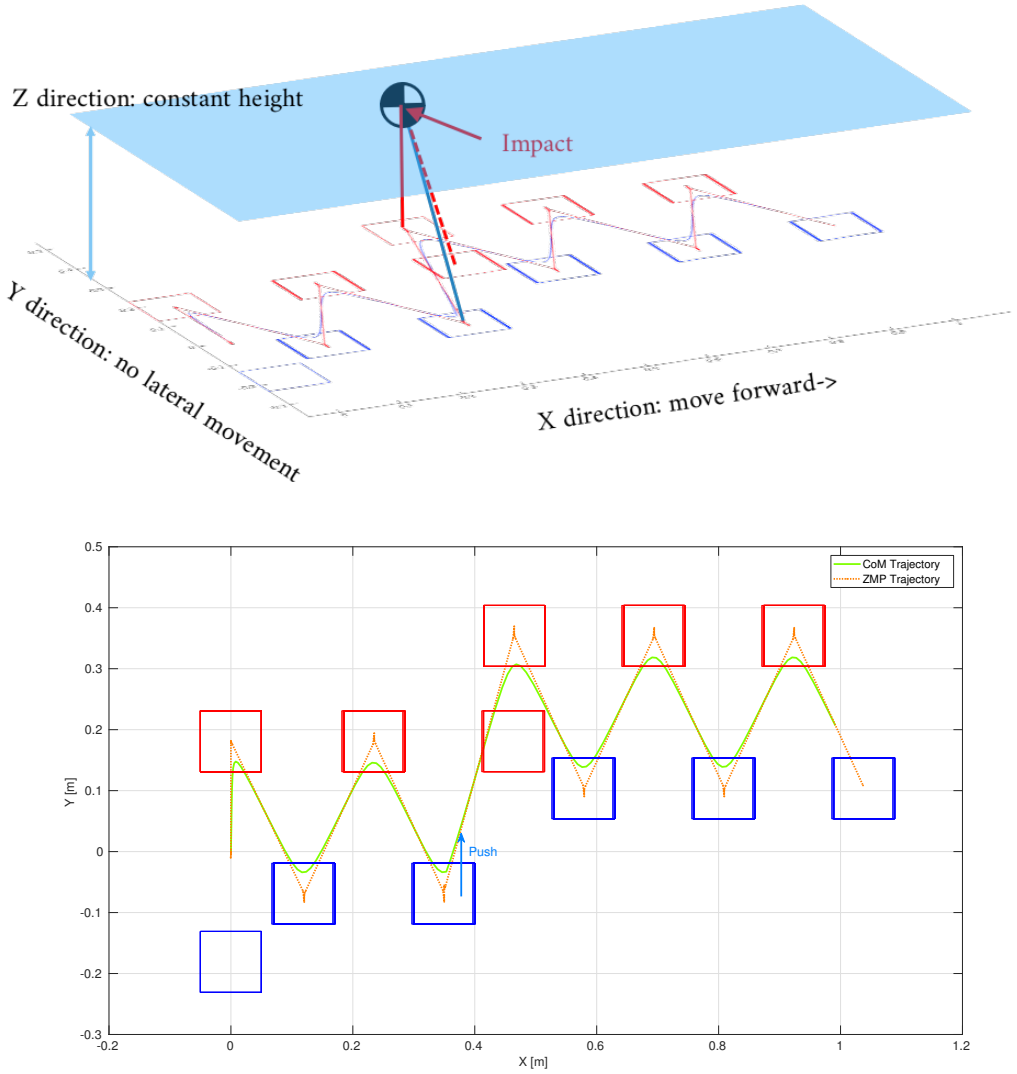


Figure 3-4: Push Recovery based on MPC with optimized foot placements(Above: 3D movement; Bottom: ground projection).

3.2.4 Details in Re-planning

Based on the simplified model, MPC is fast enough to run in real time and therefore gives us the opportunity to re-plan on the fly. Considering the complexity of the real system, to make the re-planning approach practical, there are several tricks which should be carefully though over. The details are given below following two types of re-planning strategies.

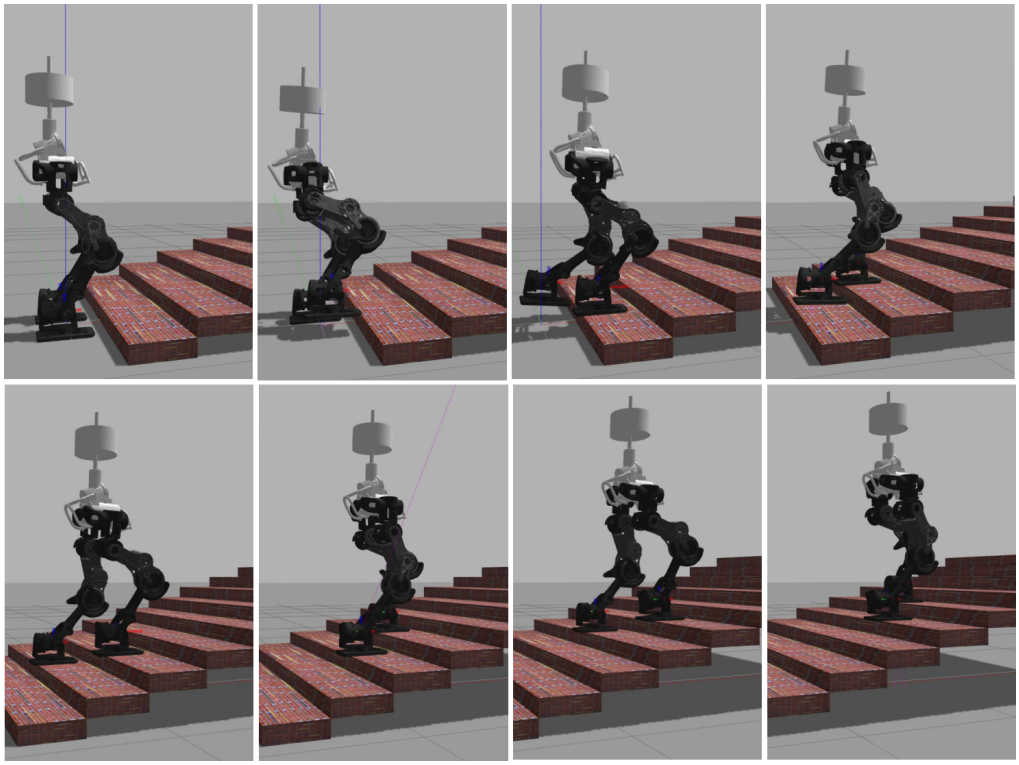


Figure 3-5: The lower body of WALK-MAN walking upwards a stair (time interval 1.5 seconds).

Re-planning at a constant frequency

One strategy to perform re-planning is to keep re-planning at a constant frequency in which no extra artificial parameter is required except the re-planning frequency. But sometimes the frequent re-planning could make the whole system too sensitive and result in loss of stability, because it will take the tracking error into account straightly without cease which may result in a regenerative feedback. To reduce the sensitivity, one way is to reshape the measured state before passing to MPC for re-planning. Take the hyperbolic tangent function as example to reshape the measured CoM position:

$$\begin{aligned} \Delta\mathbf{C} &= \mathbf{C}_{measure} - \mathbf{C}_{ref} \\ \mathbf{C}_{reshape} &= \mathbf{C}_{ref} + S(\Delta\mathbf{C})||\Delta\mathbf{C}||, \end{aligned} \tag{3.10}$$

where $\mathbf{C}_{ref}$, $\mathbf{C}_{measure}$ and $\mathbf{C}_{reshape}$ are individually the reference CoM position generated by the MPC planner, the measured current CoM position, and the reshaped one

passed to the re-planning. The hyperbolic tangent function S is one type of sigmoid function:

$$S(x) = \tanh(kx) = \frac{e^{kx} - e^{-kx}}{e^{kx} + e^{-kx}} \tag{3.11}$$

This method could alleviate the issue of sensitivity a bit. However, it also introduces the parameter k of the hyperbolic tangent function, which is cumbersome to tune.

Re-planning by event triggering

Another way to do re-planning is by event triggering, that is to say, the re-planning will only happen when one predefined event is triggered. Here the triggering event is set to when the tracking error is out of a arbitrarily given threshold. The benefit of this kind of re-planning strategy is that the dead-zone characteristic brought by the threshold will mitigate the regenerative feedback problem a lot. This method can also be considered as a special case of reshaping function of the first strategy, named deadband function here. To make it clear, the hyperbolic tangent (or tanh function) and deadband functions with different parameters are plotted in Fig. 3-6. As the figure shows, the hyperbolic tangent function can be considered as a soft version of the deadband function. For the hyperbolic tangent function, the mapping gets steeper and steeper as the parameter k increases while bigger threshold in the deadband function would enlarger the window of outputing zero.

Another issue worth mentioning is that for re-planning, which state should be chosen to connect with desired next state to generate the trajectory between them, the measured current state or the desired one (the ending state of previous trajectory). Figure 3-7 shows an example to illustrate this issue. Blue line is for the desired trajectory of last knot period. Since the tracking can not be perfect, at the end of this period (0 second in figure), the measured and desired current states are different. In order to react to possible external disturbance, re-planning should take the measured current state into account via the reshaping technologies discussed above. Here assuming the event-triggering strategy is adopted and the tracking error exceeds the

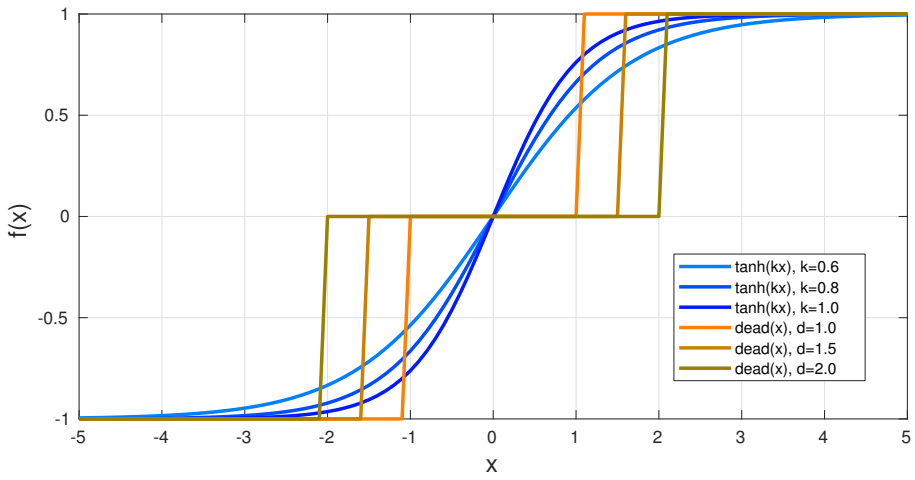
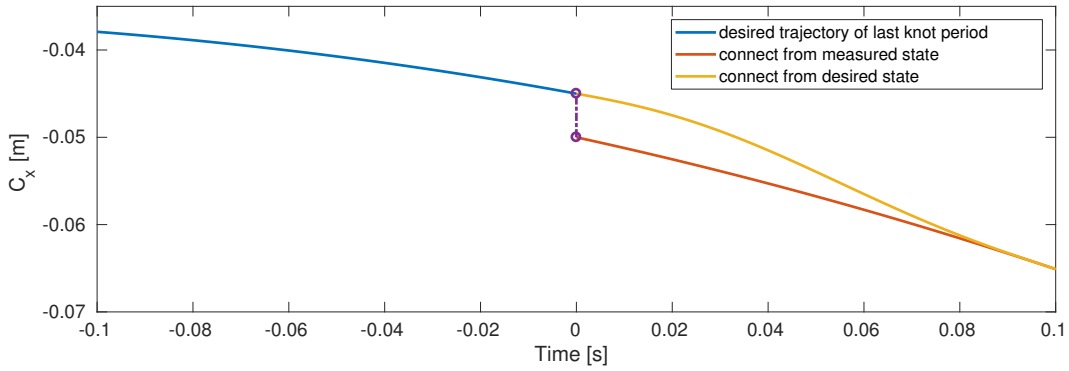


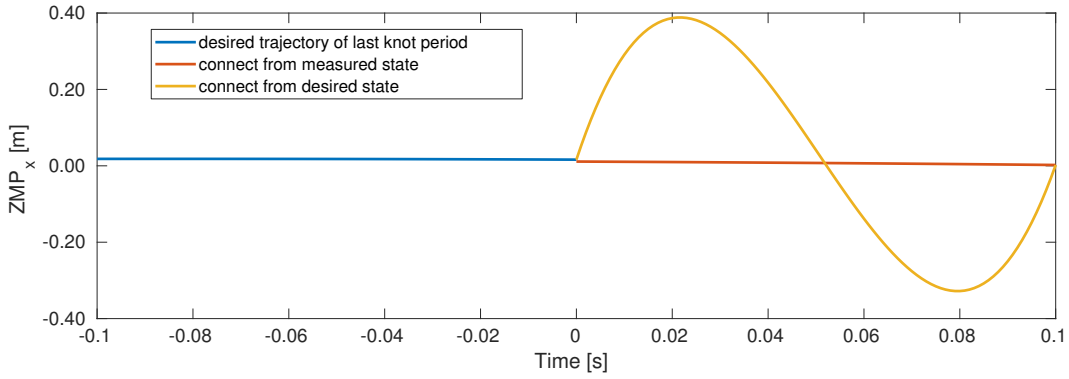
Figure 3-6: Tanh and deadband function(tanh function: k = 0.6, 0.8, 1.0; dead zone function: threshold = 1.0, 1.5, 2.0).

threshold, the re-planning is triggered and generates the desired state of next knot from current measured state based MPC. This is similar for the first constant frequency strategy, and the only litter difference is that the current measured state will be reshaped first and then passed to MPC for re-planning. The yellow line is for the trajectory connecting the current and next desired states via quintic polynomial, while the red is for the one connecting the current measured state and the next desired state. The differences between them are depicted by the CoM trajectory (Fig. 3-7 Above)and ZMP trajectory (Fig. 3-7 Bottom).

Observing the CoM trajectory, it seems better to choose the trajectory starting from the current desired state to make sure the desired trajectory is smooth all the time. However, this may result in a serious problem that ZMP trajectory exceeds the boundary of support polygon and causes the robot to fall, because directly connecting the two state is not the case considered inside the MPC optimization and no guarantee is given with respect to the ZMP constraint. This can be easily observed from the ZMP trajectories (Fig 3-7), where the yellow line oscillates a lot and is much bigger than the red one. Therefore, although the discontinuity of desired CoM trajectory exists, it is still rational to generate the trajectory by connecting the current measured state and next desired state. And the discontinuity can get mitigated by re-planning less frequently or adopting the second re-planning strategy.



(a) CoM trajectories



(b) ZMP trajectories

Figure 3-7: Trajectories of connecting measured or desired current state with re-planned next state(Above: CoM trajectories, Bottom: ZMP trajectories).

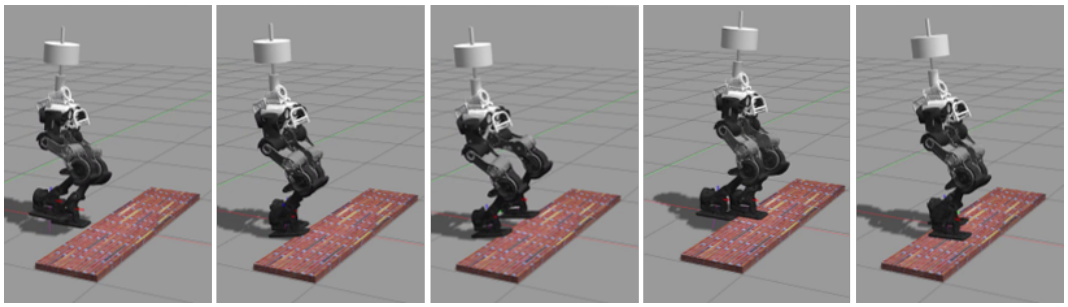
3.3 Simulation

In order to evaluate the performance of MPC, three simulations were designed for the lower body of WALK-MAN robot. The information of WALK-MAN robot is listed in the appendix A. First, the robot was expected to walk across a stair blindly to demonstrate the robustness of tackling the ground disturbance. Thereafter, the foot of the robot was shrunk to a stick and almost passive walking is achieved in the lateral motion plane. At the end, the ability of re-planning on the fly is shown by the case of push recovery. More details about these simulations are given below.

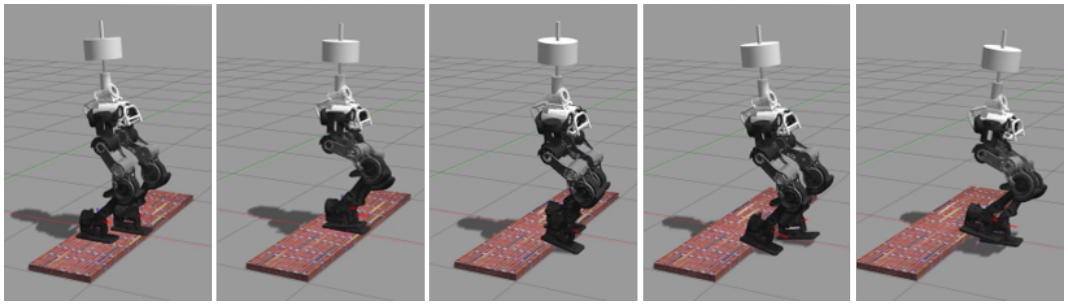
3.3.1 Walk on Uneven Terrain

Based on the MPC planner with predefined foot placements introduced in Section 3.2.1 and the optimization-based full body torque controller given in Chapter 5, the

lower body of WALK-MAN was able to walk through a stair (height 5cm) blindly as shown in Fig. 3-8.



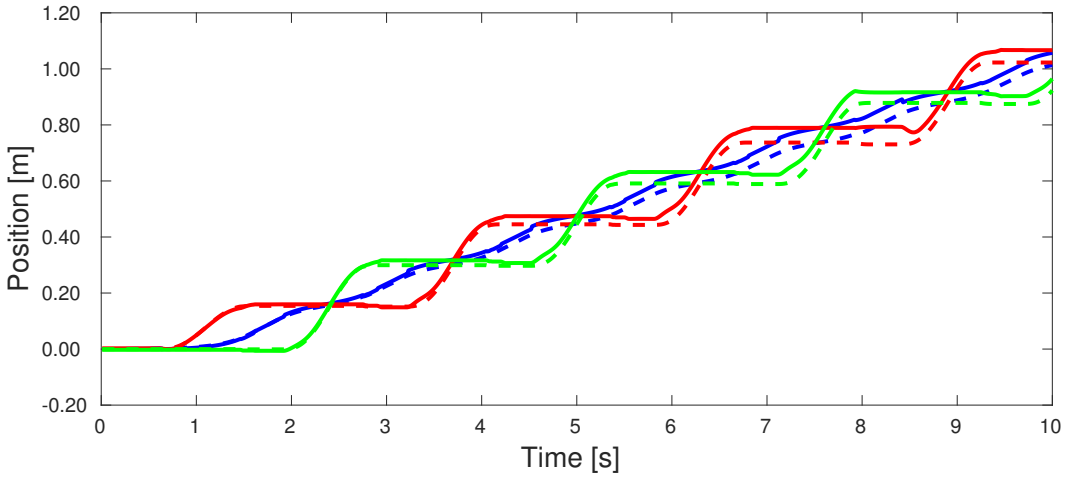
(a) Walk up a stair



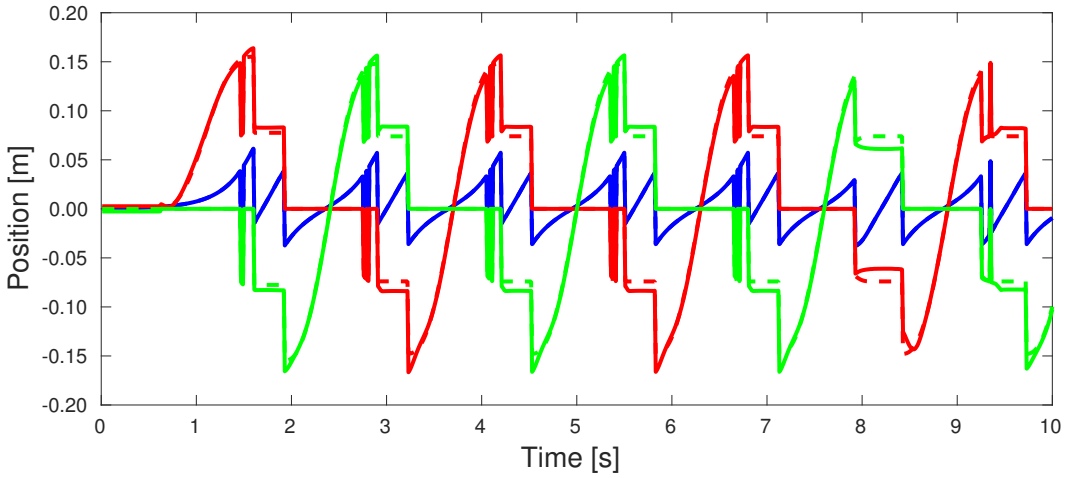
(b) Walk down a stair

Figure 3-8: WALK-MAN walking through uneven terrain (interval: 0.8 second).

To be noted, in this simulation, the high-level MPC planner does not know about the existing of the stair and always assumes the ground is flat. However, thanks to the compliant behavior brought by the full body torque control, the ground impact caused by the imperfect ground is quite small compared with position controlled robots. In addition, although no on-line re-planning is involved in this simulation, the low level controller will always transform the generated global trajectories to the local supporting foot frame and only consider the relative location of CoM and the swinging foot with respect to the supporting foot. In this way, the planned trajectory can work without global localization and adapt to the ground height variance. This can also be observed from the difference between the global and local trajectories of CoM and two feet along the forward direction as shown in Fig. 3-9. For convenience, the forward, lateral and vertical direction would be assigned as X, Y and Z directions respectively in the following.



(a) Global trajectories



(b) Local trajectories

Figure 3-9: Global and local trajectories of WALK-MAN walking through uneven terrain in the forward direction (blue line: CoM, red line: left foot, green line: right foot, solid line: measured, dash line: desired).

In Fig. 3-9, the tracking data of CoM is depicted with blue line, left foot with red line and right foot with green line. The measured and desired trajectories are presented individually with solid and dash lines. The figure above shows that as time goes on, there is a drift getting bigger and bigger in the global trajectories due to the lack of global localization. However, the robot is able to keep walking because the low level tracking only cares about the local trajectories transformed to the support foot according to the current contact state. And the local tracking is quite good as seen in the figure at bottom, where the spiking signal exists because the contact state

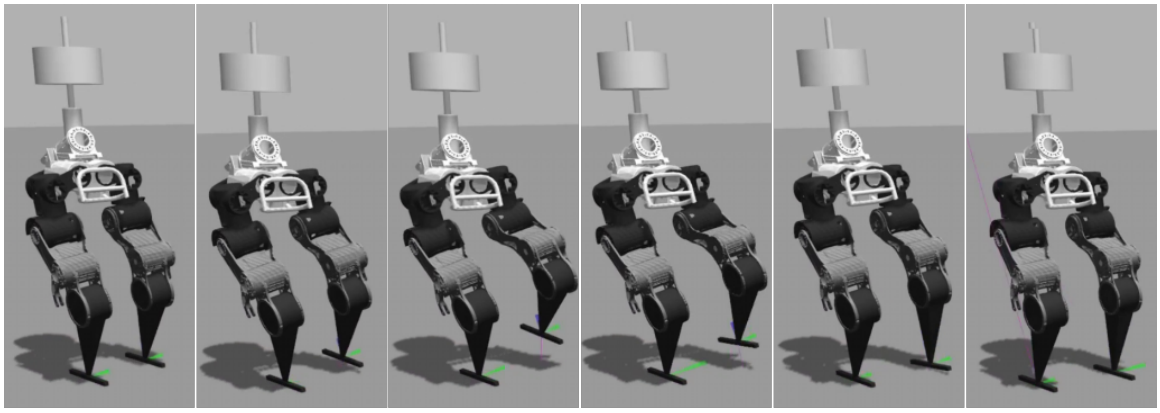


Figure 3-10: Snapshots of WALK-MAN walking with narrow feet (interval: 0.2 second).

switches at landing.

3.3.2 Walk with Narrow Feet

To exploit the potential of the MPC planner, a challenging task was set that forces the robot to walk with narrow feet. The ability to walk with narrow feet may be required in many bipedal robot cases. For instance, for bipedal robots whose ankle roll actuators are excluded to reduce leg inertia and achieve highly dynamic motions[20], the inability to perform the feet roll motion requires the use of feet with narrow width to cope with lack of ability to orient the feet around its longitudinal axis. Other examples include but are not limited to robots wearing ice skates or robots with weak ankle actuation. On the other side, the study of this topic is also helpful when considering the line contacts of walking on rough terrain, such as traversing irregular jagged blocks.

With little revise on the controller of walking on uneven terrain, the lower body of WALK-MAN was also able to walk with narrow feet (foot size: length 0.32 m, width and thickness 0.02 m) as shown in Fig. 3-10. The desired CoM height was set around 0.76 meter, and the desired feet distance was 0.35 meter.

Considering the used narrow feet, in the MPC planner, $\mathbf{P}^{y}_{k+1}$ is expected to stay close to $\mathbf{P}^{y\text{-}ref}_{k+1}$ during single support phase, because the torque of ankle roll joint is not available and the support polygon shrinks to a line. Hence, the two weights γ in

X and Y directions should be different, and the one in Y direction should be much bigger than in X direction. For the same reason, less weight is put on the tracking of CoM trajectories in the lateral plane while more weight on forcing the torque of ankle roll joint of support foot to zero in the low level torque controller introduced in Chapter 5.

The tracking data of CoM (blue line) and two feet (red line: left foot, green line: right foot) in the world frame is shown in Fig. 3-11. The measured and desired trajectories are presented separately with solid and dash lines. The trajectories in z direction were tracked very well, which indicates that the feature of constant CoM height was guaranteed. The trajectories in X and Y directions drifted a bit as time went on, because the tracking worked in the local frame of each step defined by the stationary foot, no global localization was implemented as told before.

3.3.3 Push Recovery

As said at the beginning of this chapter, the essential reason to simplify a model for MPC is to reduce the computation and implement it in real time. To demonstrate this, based on the MPC planner with optimized foot placements, the lower body of WALK-MAN was controlled to resist external push and resume stable walking by updating foot placement and re-planning CoM trajectory online. Fig. 3-12 is the snapshot of a successful push recovery in simulation. For the online re-planning, the event triggering strategy is adopted and the threshold for the tracking error of CoM positon is set as 2 cm in both X and Y directions.

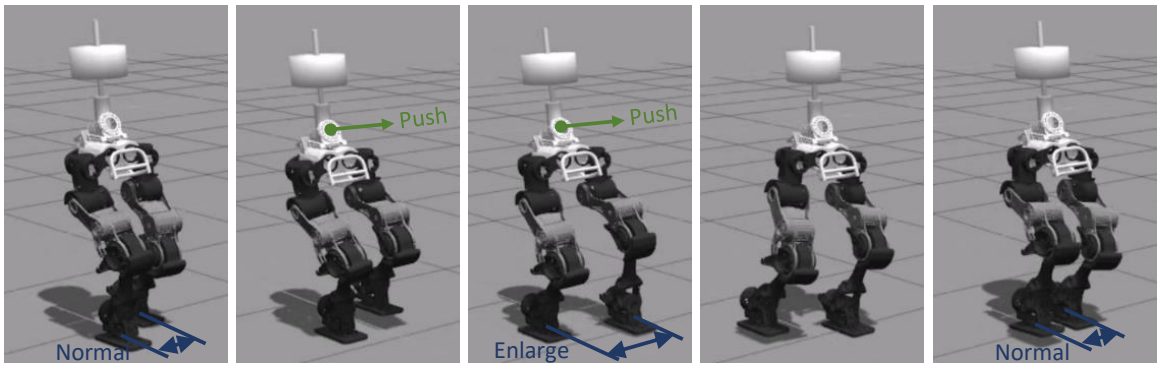


Figure 3-12: Push recovery of WALK-MAN lower body based on MPC re-planning.

Fig. 3-13 shows the tracking data of CoM (blue line) and two feet (red line: left foot, green line: right foot) in the lateral direction in the world frame. The same with above, the measured and desired trajectories are presented individually with solid and dash lines. In the simulation, the external push (200N*0.1s) was exerted three times, and as a result, the tracking error of CoM position would go out of the threshold immediately. Thereafter, the robot would re-plane the foot placements and CoM trajectory at the same time. As shown in Fig. 3-12, when the push was exerted on the waist, a side step was taken on the left side of the robot to keep balancing. Thanks to the prompt re-planning, the tracking is still very good even after the push and the robot is able to restore normal walking within two steps after the disturbance.

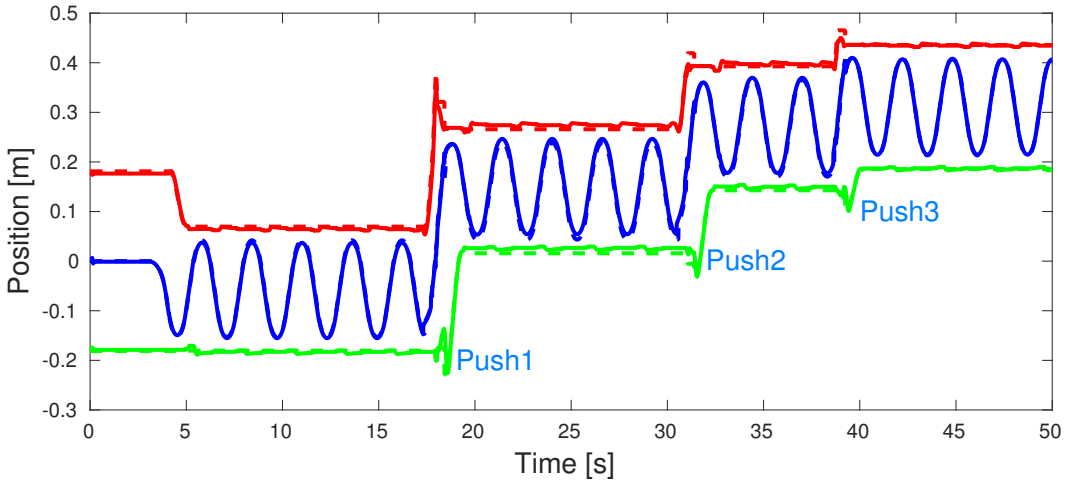
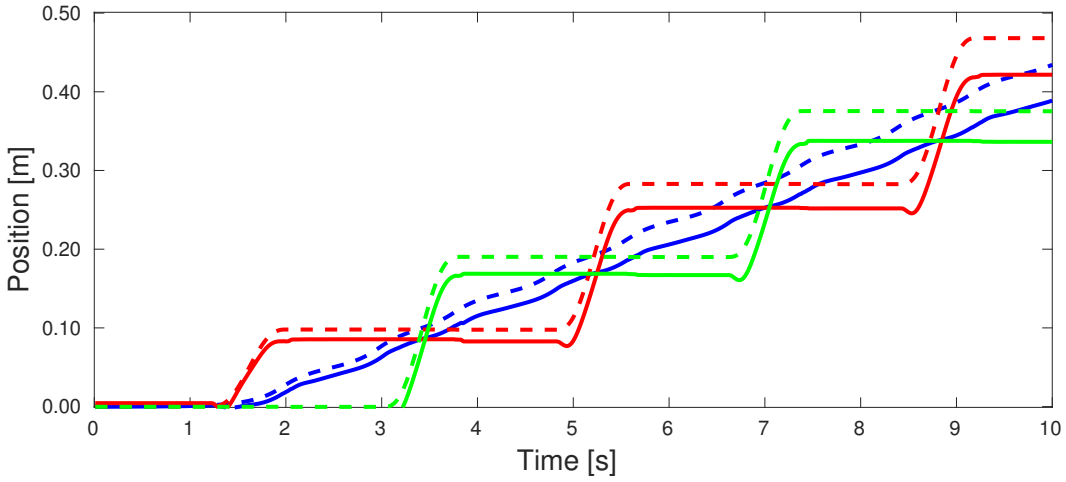


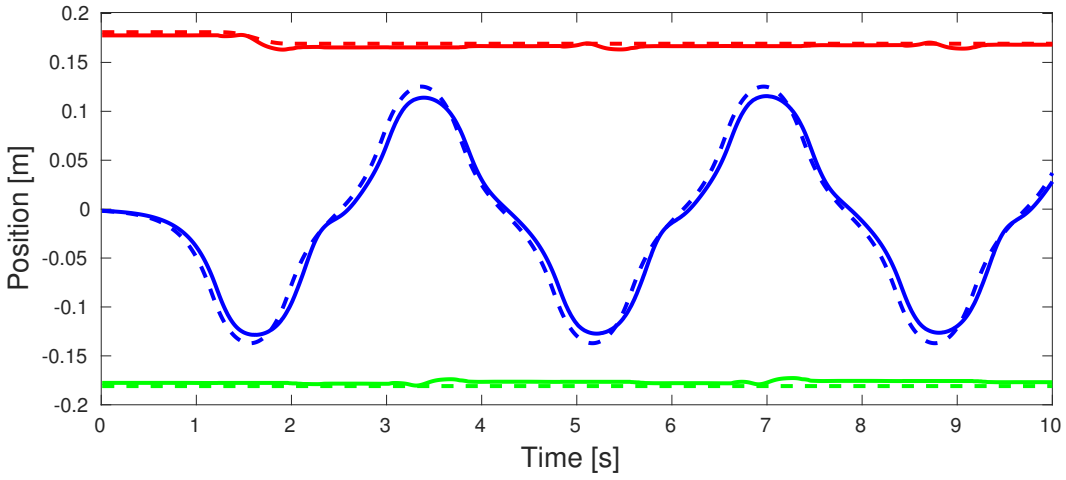
Figure 3-13: Lateral CoM trajectory of WALK-MAN under push recovery.

3.4 Summary

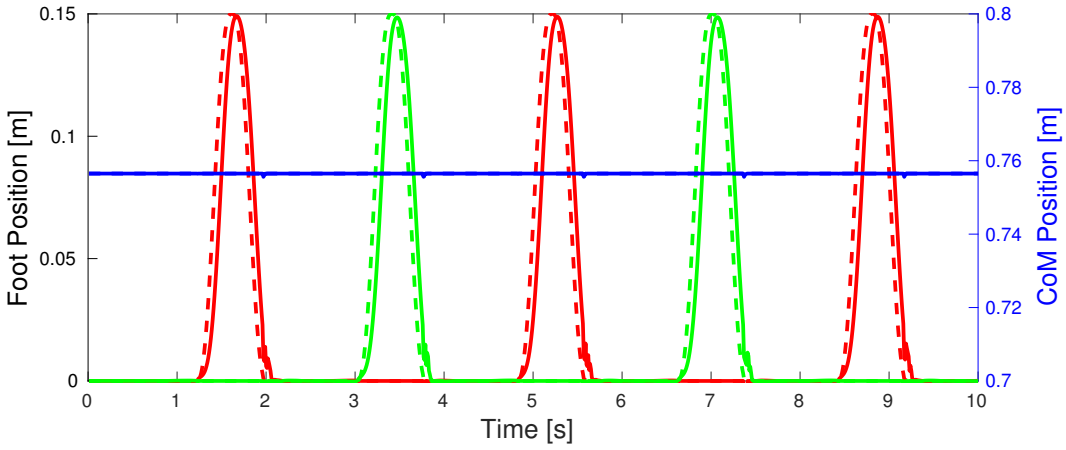
In this chapter, the robot is simplified as a point-mass model and the CoM trajectory is discretized for optimization. Two types of model predictive control are implemented, one with predefined foot placements and the other one able to optimize foot placements and CoM trajectory simultaneously. The MPC with sparse discretization, like $T = 0.1$, is good enough to guarantee walking stability and also fast enough to run in real time. Two re-planning strategies are given, re-planning at a constant frequency and re-planning by event triggering. In addition, which kind of state should be used as the starting point of re-planning is also discussed, the measured current state or the desired one. At the end of this chapter, three simulations were performed on the lower body of WALK-MAN robot. The feasibility and robustness of the MPC planner is demonstrated.



(a) CoM and foot trajectories in X direction.



(b) CoM and foot trajectories in Y direction.



(c) CoM and foot trajectories in Z direction.

Figure 3-11: CoM and foot trajectories of WALK-MAN in the world frame (blue line: CoM, red line: left foot, green line: right foot, solid line: measured, dash line: desired).

Chapter 4

Nonlinear Model Planning: Multiple Shooting Method

Studies on sensorimotor control reveals that the central nervous system (CNS) internally simulates forward the behaviors of the motor system for planning and control, thus "forward" model is termed (or internal model) as a dynamic representation for the motor system to use the current state to predict the future state [40]. Also, forward/internal models inherit the delay property coming from sensory feedback and motor response [39], which can be considered a-priori in the control stage while generating a new motor command [64]. New study also proves that not only the vertebrates, but also the invertebrates, the Plathemis lydia, whose sensorimotor control exploits the dynamic models of both itself and the prey for the prediction and planning during the high-performance control of interception steering [41].

While recent robotics research has used inverse dynamics approach [19], similar to the inverse models for CNS, to solve whole-body control problem, I am motivated to investigate what types of models can be used, as those forward models in CNS, and how to exploit forward models for planning and control that can further enhance dynamic walking in terms of robustness and human-level of likeness. Particularly, inspired by Raibert's idea of controlling foot placement to balance legged robot, I hereby proposed two nonlinear models for forward simulation to decide foot placement more precisely through multiple shooting method with gradient descent search inside.

It was shown that these nonlinear models have much better representation of the full robot dynamics than the over simplified single point mass model and meanwhile is computationally easy to realize. Therefore, more accurate foot placements can be achieved based on these improved models compared to the previous trial-and-error tuning of control coefficients as in [50].

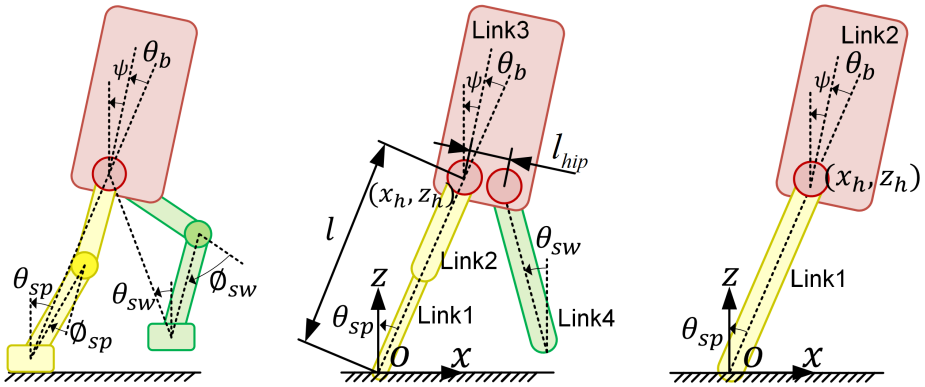
This chapter is organized as follows. In Section 4.1, two nonlinear models are introduced, and the mapping between the models and the real robot is also explained. Section 4.2 presents the algorithm of using forward simulation of nonlinear models to predict the foot placement, and the trajectories of all the joints are generated according to the desired foot placement. In Section 4.3, several simulations are performed to demonstrate the robust and human-like performance of the proposed method. The chapter ends with a summary.

4.1 Nonlinear models

4.1.1 Nonlinear Model for Sagittal Dynamics

To capture the major dynamic characteristics of bipedal walking in sagittal/lateral plane, the robot is simplified to a four-link model as Fig. 4-1 (b) shows. In this four-link model, the support leg and the swing leg are individually modeled as a prismatic link and a constant link. Here, using a prismatic joint to represent the revolute knee joint is for improving the stability of numerical calculation. The dynamics of swing motion matters, however, the dynamic property caused by the changing leg length is much less critical. Hence, it is reasonable to model the swing leg as one link with constant length. The foot is modeled as force wrench by applying torque to first joint. Assuming no slippage, the equation of motion (EoM) can be written as a fix-based robot instead of a floating base one for faster computation:

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{h}(\mathbf{q}, \dot{\mathbf{q}}) = \boldsymbol{\tau}, \tag{4.1}$$



(a) Full model (b) Four-link model (c) Two-link model

Figure 4-1: Two nonlinear models derived from a full robot model.

where $\mathbf{q}$ is the joint angles, $\mathbf{M}(\mathbf{q})$ is inertia matrix, $\mathbf{h}(\mathbf{q}, \dot{\mathbf{q}})$ is the sum of centrifugal, Coriolis and gravity forces, and $\boldsymbol{\tau}$ is joint torque.

The task space vector $\mathbf{s} = [\theta_{\text{sp}}\ l\ \psi\ \theta_{\text{sw}}]^T$ denoted in Fig. 4-1 is concerned first. Define the line connecting hip and ankle joints as the virtual leg. Then θ_{sp} is the angle between virtual support leg and vertical line, l is the length of virtual support leg, ψ is the upper body posture, and θ_{sw} is the angle between virtual swing leg and vertical line. The joint space vector $\mathbf{q}$ can be transformed to the task space vector $\mathbf{s}$ by a constant transformation matrix $\mathbf{A}$ determined by the robot parameters, such that $\mathbf{s} = \mathbf{A}\mathbf{q}$ and $\ddot{\mathbf{s}} = \mathbf{A}\ddot{\mathbf{q}}$. Thus, the EoM can be rewritten as:

$$\mathbf{M}^*(\mathbf{s})\ddot{\mathbf{s}} + \mathbf{h}^*(\mathbf{s}, \dot{\mathbf{s}}) = \boldsymbol{\tau}, \tag{4.2}$$

where $\mathbf{M}^*(\mathbf{s}) = \mathbf{M}(\mathbf{q})\mathbf{A}^\dagger$, $\mathbf{h}^*(\mathbf{s}, \dot{\mathbf{s}}) = \mathbf{h}(\mathbf{q}, \dot{\mathbf{q}})$, and $\mathbf{A}^\dagger$ is the pseudo inverse of $\mathbf{A}$.

Equation (4.2) provides options to include either closed loop control, actuator dynamics, or both into forward simulation. The close loop control can be calculated based on the desired and current state and generate joint torque τ, then the EoM (4.2) can be utilized to do forward simulation to predict the future state. In the next numerical integration step, the newly simulated state will be treated as the current state and the computation iterates. If available, the actuators dynamics can be added in a cascade manner after the feedback control, then the output torques of the actuators are the input of (4.2).

4.1.2 Nonlinear Model for Lateral Dynamics

It shall be noted that the model in Fig. 4-1 (b) and the method above can be perfectly applied to both sagittal and lateral gait control. The only difference is the parameter l_{hip} that describes the separation of hips. However, interestingly, the prior comparison study found out that the lateral dynamics has a particular feature which allows the model to be further simplified, as depicted by a two-link model in Fig. 4-1 (c), where the first link represents the support leg and the second link is one equivalent body lumping the upper body and the swing leg.

For the sagittal gait, the target is to pass over the support leg and continue to place next foot, so it includes a convergent phase and a divergent phase. The change of support leg length affects the potential energy of the whole robot and the future evolution of robot state. In contrast, the lateral gait needs to achieve a stable upper body oscillation laterally, and the upper body never surpasses the stance foot sideways. Therefore, a limited variation of support leg length will not cause the upper body to overshoot sideway. Meanwhile, the swing leg has much smaller range of lateral motion compared to the sagittal one, thereby the torso dynamics dominates over the swing leg dynamics in the lateral scenario, which leads to no distinctive difference when a two-link model is used.

In order to improve the energy efficiency, the knee joint is expected to keep as straight as possible, similar to what humans do. So the support link in Fig. 4-1 (c) is set as full leg length. The reduced dynamics equation can be written as below:

$$\begin{bmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{bmatrix} \begin{bmatrix} \ddot{q}_1 \\ \ddot{q}_2 \end{bmatrix} + \begin{bmatrix} h_1 \\ h_2 \end{bmatrix} = \begin{bmatrix} \tau_1 \\ \tau_2 \end{bmatrix}, \tag{4.3}$$

where m_{11}, m_{12}, m_{21} and m_{22} are the elements of inertia matrix, $\ddot{q}_1$ and $\ddot{q}_2$ are the two joint accelerations, h_1 and h_2 are the elements of centrifugal, Coriolis and gravity force, and τ_1 and τ_2 are the two joint torques. The transformation between $(\theta_{\text{sp}}, \psi)$

and (q_1, q_2) is:

$$\begin{bmatrix} \theta_{\text{sp}} \\ \psi \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} q_1 \\ q_2 \end{bmatrix} \tag{4.4}$$

Given the expected posture trajectory ψ of torso with perfect tracking, a math trick here is to consider only the first row in (4.3) and to eliminate τ_2. Substitute equation (4.4) to the first row of (4.3), and then $\ddot{\theta}_{\text{sp}}$ can be derived as:

$$\ddot{\theta}_{\text{sp}} = \frac{\tau_1 - m_{12}\ddot{\psi} - h_1}{m_{11} - m_{12}} = f(\tau_1, \ddot{\psi}, \theta_{\text{sp}}, \psi, \dot{\theta}_{\text{sp}}, \dot{\psi}) \tag{4.5}$$

By using (4.5), the trajectory of support leg θ_{sp} can be obtained from the ankle torque τ_1 and the expected upper body posture ψ without considering the internal joint torque τ_2. The same knack also works for systems with four links or even more links. Similar arguments about this kind of simplification were made via conservation of angular momentum in [38].

4.1.3 From Nonlinear Models to the Robot

Here only address the mapping between model and robot concerning the support leg only, and the planning of swing leg trajectory will be explained in Section 4.2.2. For the four-link model, it is easy to map due to the same degrees of freedom. As for the two-link model, the distance from hip to ankle joint in the robot is always changing while the length of link 1 in the two-link model is constant. Define the hip position and velocity from the state estimation as (x_h, z_h) and $(\dot{x}_h, \dot{z}_h)$ respectively, as shown in Fig. 4-1. θ_{sp} and $\dot{\theta}_{\text{sp}}$ are calculated without considering the movement along virtual leg direction as:

$$\begin{aligned} \theta_{\text{sp}} &= \arctan(-x_h/z_h), \\ \dot{\theta}_{\text{sp}} &= -\dot{x}_h \cos(\theta_{\text{sp}}) - \dot{z}_h \sin(\theta_{\text{sp}}). \end{aligned} \tag{4.6}$$

4.2 Forward Simulation for Control

For a simple linear model like the linear inverted pendulum model, to achieve the optimal CoM trajectory, MPC can be easily implemented. However, once the model

is nonlinear and particularly involves discrete change of energy state as in the above models, utilizing MPC requires a lot of computation and real-time requirement is hard to guarantee. Instead of optimizing continuous trajectories of robot state, one faster option to significantly reduce the computation time is to optimize only key parameters. A great example is Raibert's work in [51] that a stable walking was realized by choosing foot placement at the moment of touch-down, where an intuitive equation combining a neutral point and a feedback item was used to calculate foot placement. One major disadvantage is that the feedback gain over the walking velocity error needs to be tuned empirically due to the lack of dynamics information. To address this, the proposed nonlinear models are taken into account.

4.2.1 Accurate Foot Placement Control

Foot placement primarily determines the discrete nature of step-to-step transition, and thus the stability of walking and control of the robot movement [51],[58],[65],[66]. If foot placement has a monotonous relationship with the velocity of robot, then it is possible to iteratively search for an appropriate foot placement fast to achieve desired velocity and keep stable walking. Following this idea, Fig. 4-2 is drawn to show a general result based on the two-link model, in which the length of link 1 is equal to 1 m, upper body mass is 60 kg and the gravity acceleration is 9.81 m/s^2. Here define the initial position $\theta_{\mathrm{sp}}^{\mathrm{ini}}$ as foot placement. With regard to the same initial velocity $\dot{\theta}_{\mathrm{sp}}^{\mathrm{ini}}$, the velocity $\dot{\theta}_{\mathrm{sp}}^{\mathrm{end}}$ at the end of a fix period of 0.5s changes almost linearly to the different initial position $\theta_{\mathrm{sp}}^{\mathrm{ini}}$. Hence given the robot state predicted by the forward simulation at the end of the current step, it is very easy to use the gradient descent method to find an appropriate foot placement for the next step to reach a desired state. This is the essential idea of utilizing forward simulation for accurate foot placement control in the presence of multi-body effects during very dynamic locomotion.

To determine the foot placement for next step, the initial velocity ${}^{k+1}\dot{\theta}_{\mathrm{sp}}^{\mathrm{ini}}$ of next step should be obtained first. k and $k+1$ in the left superscript means individually the current and next steps hereafter. The initial velocity of next step ${}^{\mathrm{k+1}}\dot{\theta}_{\mathrm{sp}}^{\mathrm{ini}}$ can be derived from the predicted state $({}^{\mathrm{k}}\theta_{\mathrm{sp}}^{\mathrm{end}}, {}^{\mathrm{k}}\dot{\theta}_{\mathrm{sp}}^{\mathrm{end}})$ at the end of current step. Fig. 4-3

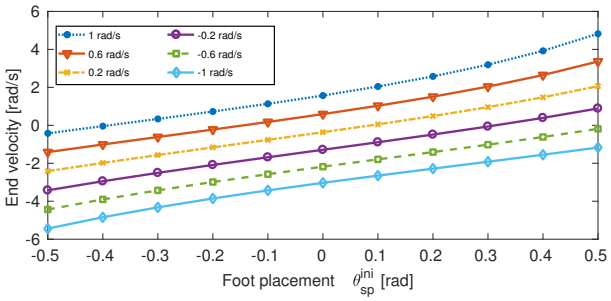
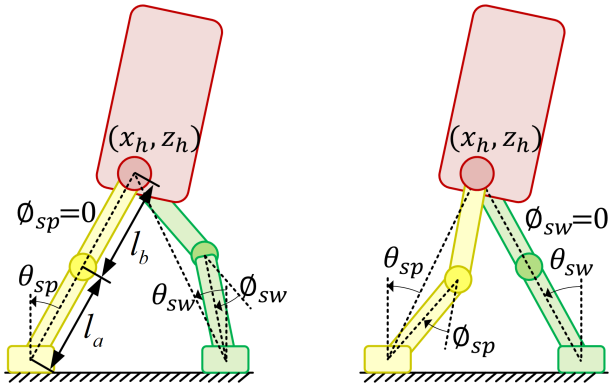


Figure 4-2: Relationship between foot placement and end velocity.



(a) Straight support leg (b) Straight swing leg

Figure 4-3: Two robot configurations for touch-down.

shows the robot configurations at touch-down moment when the support and swing legs swap. To represent the ground impact, the energy along the direction of virtual support leg of next step is lost, and the rest produces only the angular velocity around the new stance foot. Then $\dot{\theta}_{\mathrm{sp}}^{\mathrm{ini}}$ of next step can be calculated as in (4.7). For narrative convenience, the switch function for both two- and four-link models is termed as $f_{\mathrm{sw}}({}^{\mathrm{k}}\theta_{\mathrm{sp}}^{\mathrm{init}}, {}^{\mathrm{k+1}}\theta_{\mathrm{sp}}^{\mathrm{ini}}, {}^{\mathrm{k}}\dot{\theta}_{\mathrm{sp}}^{\mathrm{end}})$,

$$\begin{aligned} {}^{\mathrm{k+1}}\dot{\theta}_{\mathrm{sp}}^{\mathrm{ini}} &= f_{\mathrm{sw}}({}^{\mathrm{k}}\theta_{\mathrm{sp}}^{\mathrm{init}}, {}^{\mathrm{k+1}}\theta_{\mathrm{sp}}^{\mathrm{ini}}, {}^{\mathrm{k}}\dot{\theta}_{\mathrm{sp}}^{\mathrm{end}}) \\ &= {}^{\mathrm{k}}\dot{\theta}_{\mathrm{sp}}^{\mathrm{end}} \cos({}^{\mathrm{k+1}}\theta_{\mathrm{sp}}^{\mathrm{ini}} - {}^{\mathrm{k}}\theta_{\mathrm{sp}}^{\mathrm{end}}). \end{aligned} \tag{4.7}$$

Algorithm 1 is used to determine the foot placement. First, predict the robot state $({}^{\mathrm{k}}\theta_{\mathrm{sp}}^{\mathrm{end}}, {}^{\mathrm{k}}\dot{\theta}_{\mathrm{sp}}^{\mathrm{end}})$ at the end of current step by performing forward simulation from the current measured state. Then ${}^{\mathrm{k+1}}\theta_{\mathrm{sp}}^{\mathrm{ini}}(0)$ and ${}^{\mathrm{k+1}}\theta_{\mathrm{sp}}^{\mathrm{ini}}(1)$ are set to specific values to start the iteration of multiple shooting via gradient descent method. Afterwards

Algorithm 1: Determine foot placement (Pseudo code).

Input : ${}^{\mathrm{k}}\theta_{\mathrm{sp}}^{\mathrm{ini}}$, ${}^{\mathrm{k}}\dot{\theta}_{\mathrm{sp}}^{\mathrm{ini}}$, ${}^{\mathrm{k+1}}\dot{\theta}_{\mathrm{sp}}^{\mathrm{des}}$.
Output: ${}^{\mathrm{k+1}}\theta_{\mathrm{sp}}^{\mathrm{ini}}$

${}^{\mathrm{k}}\theta_{\mathrm{sp}}^{\mathrm{end}}, {}^{\mathrm{k}}\dot{\theta}_{\mathrm{sp}}^{\mathrm{end}} \leftarrow \mathrm{ForwardSim}({}^{\mathrm{k}}\theta_{\mathrm{sp}}^{\mathrm{ini}}, {}^{\mathrm{k}}\dot{\theta}_{\mathrm{sp}}^{\mathrm{ini}})$
${}^{\mathrm{k+1}}\theta_{\mathrm{sp}}^{\mathrm{ini}}(0) = 0$
${}^{\mathrm{k+1}}\theta_{\mathrm{sp}}^{\mathrm{ini}}(1) = \alpha$
for $n \leftarrow 1$ **to** N **do**
  ${}^{\mathrm{k+1}}\dot{\theta}_{\mathrm{sp}}^{\mathrm{ini}}(n-1) \leftarrow f_{\mathrm{sw}}({}^{\mathrm{k}}\theta_{\mathrm{sp}}^{\mathrm{end}}, {}^{\mathrm{k+1}}\theta_{\mathrm{sp}}^{\mathrm{ini}}(n-1), {}^{\mathrm{k}}\dot{\theta}_{\mathrm{sp}}^{\mathrm{end}})$
  ${}^{\mathrm{k+1}}\dot{\theta}_{\mathrm{sp}}^{\mathrm{ini}}(n) \leftarrow f_{\mathrm{sw}}({}^{\mathrm{k}}\theta_{\mathrm{sp}}^{\mathrm{end}}, {}^{\mathrm{k+1}}\theta_{\mathrm{sp}}^{\mathrm{ini}}(n), {}^{\mathrm{k}}\dot{\theta}_{\mathrm{sp}}^{\mathrm{end}})$
  ${}^{\mathrm{k+1}}\dot{\theta}_{\mathrm{sp}}^{\mathrm{end}}(n-1) \leftarrow \mathrm{ForwardSim}({}^{\mathrm{k+1}}\theta_{\mathrm{sp}}^{\mathrm{ini}}(n-1), {}^{\mathrm{k+1}}\dot{\theta}_{\mathrm{sp}}^{\mathrm{ini}}(n-1))$
  ${}^{\mathrm{k+1}}\dot{\theta}_{\mathrm{sp}}^{\mathrm{end}}(n) \leftarrow \mathrm{ForwardSim}({}^{\mathrm{k+1}}\theta_{\mathrm{sp}}^{\mathrm{ini}}(n), {}^{\mathrm{k+1}}\dot{\theta}_{\mathrm{sp}}^{\mathrm{ini}}(n))$
  if $|({}^{\mathrm{k+1}}\dot{\theta}_{\mathrm{sp}}^{\mathrm{end}}(n) - {}^{\mathrm{k+1}}\dot{\theta}_{\mathrm{sp}}^{\mathrm{des}})| < \varepsilon$ **or** $n == N$ **then**
    return ${}^{\mathrm{k+1}}\theta_{\mathrm{sp}}^{\mathrm{ini}}(n)$
  else
    $$\Delta^{\mathrm{k+1}}\theta_{\mathrm{sp}}^{\mathrm{ini}} = \frac{({}^{\mathrm{k+1}}\dot{\theta}_{\mathrm{sp}}^{\mathrm{end}}(n) - {}^{\mathrm{k+1}}\dot{\theta}_{\mathrm{sp}}^{\mathrm{des}}))({}^{\mathrm{k+1}}\theta_{\mathrm{sp}}^{\mathrm{ini}}(n) - {}^{\mathrm{k+1}}\theta_{\mathrm{sp}}^{\mathrm{ini}}(n-1))}{{}^{\mathrm{k+1}}\dot{\theta}_{\mathrm{sp}}^{\mathrm{end}}(n) - {}^{\mathrm{k+1}}\dot{\theta}_{\mathrm{sp}}^{\mathrm{end}}(n-1)}$$
    ${}^{\mathrm{k+1}}\theta_{\mathrm{sp}}^{\mathrm{ini}}(n+1) = {}^{\mathrm{k+1}}\theta_{\mathrm{sp}}^{\mathrm{ini}}(n) - \Delta^{\mathrm{k+1}}\theta_{\mathrm{sp}}^{\mathrm{ini}}$
  end
end

${}^{\mathrm{k+1}}\dot{\theta}_{\mathrm{sp}}^{\mathrm{ini}}(n-1)$ and ${}^{\mathrm{k+1}}\dot{\theta}_{\mathrm{sp}}^{\mathrm{ini}}(n)$ are derived from the corresponding switch function (4.7). Based on this, ${}^{\mathrm{k+1}}\dot{\theta}_{\mathrm{sp}}^{\mathrm{end}}(n-1)$ and ${}^{\mathrm{k+1}}\dot{\theta}_{\mathrm{sp}}^{\mathrm{end}}(n)$ are calculated from forward simulation using (4.2). Then, if the error is smaller than the tolerance ε or the iteration times are more than the maximum number, the result ${}^{\mathrm{k+1}}\theta_{\mathrm{sp}}^{\mathrm{ini}}(n)$ is returned; otherwise, a gradient descent method will be applied to calculate ${}^{\mathrm{k+1}}\theta_{\mathrm{sp}}^{\mathrm{ini}}(n+1)$ for the next iteration.

4.2.2 Mapping From Virtual Legs to Revolute Knees

From the algorithms mentioned above, the trajectory of θ_{sp} of support leg and $(\theta_{\mathrm{sw}}^{\mathrm{end}}, \dot{\theta}_{\mathrm{sw}}^{\mathrm{end}})$ of swing leg can be obtained. By now, the knee joint states $(\phi_{\mathrm{sp}}, \dot{\phi}_{\mathrm{sp}})$ and $(\phi_{\mathrm{sw}}, \dot{\phi}_{\mathrm{sw}})$ are still not taken into account. As illustrated by the two robot configurations at touch-down in Fig. 4-3, the support leg is expected to keep straight during the most period of walking. In this case, the knee angle $\phi_{\mathrm{sp}}^{\mathrm{end}}$ and $\phi_{\mathrm{sw}}^{\mathrm{end}}$ of the support and swing legs at touch-down can be calculated given the shin length l_a and

the thigh length l_b:

$$\phi_{\text{sp}}^{\text{end}} = \begin{cases} \pi - \arccos(\dfrac{l_a^2 + l_b^2 - c_{\text{sp}}^2}{2 l_a l_b}), & \theta_{\text{sp}}^{\text{end}} \geq \theta_{\text{sw}}^{\text{end}} \\ 0, & \theta_{\text{sp}}^{\text{end}} < \theta_{\text{sw}}^{\text{end}} \end{cases}$$

$$\phi_{\text{sw}}^{\text{end}} = \begin{cases} 0, & \theta_{\text{sp}}^{\text{end}} \geq \theta_{\text{sw}}^{\text{end}} \\ \pi - \arccos(\dfrac{l_a^2 + l_b^2 - c_{sw}^2}{2 l_a l_b}), & \theta_{\text{sp}}^{\text{end}} < \theta_{\text{sw}}^{\text{end}} \end{cases}$$

$$c_{\text{sp}} = (l_a + l_b)\frac{\cos(\theta_{\text{sw}}^{\text{end}})}{\cos(\theta_{\text{sp}}^{\text{end}})}, c_{\text{sw}} = (l_a + l_b)\frac{\cos(\theta_{\text{sp}}^{\text{end}})}{\cos(\theta_{\text{sw}}^{\text{end}})},$$

The posture of upper body is expected to always keep upright $\psi = 0$, then all the measured current states $(\phi_{\text{sp}}^{\text{init}}, \psi^{\text{init}}, \theta_{\text{sw}}^{\text{init}}, \phi_{\text{sw}}^{\text{init}}, \dot{\phi}_{\text{sp}}^{\text{init}}, \dot{\psi}^{\text{init}}, \dot{\theta}_{\text{sw}}^{\text{init}}, \dot{\phi}_{\text{sw}}^{\text{init}})$ and the expected end states $(\phi_{\text{sp}}^{\text{end}}, \psi^{\text{end}}, \theta_{\text{sw}}^{\text{end}}, \phi_{\text{sw}}^{\text{end}}, \dot{\phi}_{\text{sp}}^{\text{end}}, \dot{\psi}^{\text{end}}, \dot{\theta}_{\text{sw}}^{\text{end}}, \dot{\phi}_{\text{sw}}^{\text{end}})$ of current step are known by now, so they can be smoothly connected by polynomials and keep continuity of the state. An intermediate point should be added into the trajectory of ϕ_{sw} to lift up the swing leg and create ground clearance for walking. Combining them with the trajectory of θ_{sp} generated by forward simulation and the desired $\psi = 0$, the trajectories for all joints are available. As mentioned before, this method can be applied to the movement of both sagittal and lateral planes with the only difference on the parameters of model.

4.3 Simulation

To evaluate the effectiveness and performance, the proposed control method was tested in several different simulation scenarios on two simulation platforms. Both of them showed promising results.

4.3.1 Traverse A Stair Blindly

This simulation was performed on a planar bipedal robot built in ADAMS. The parameters of this robot are listed in Table 4.1. The ankle and knee joints were position controlled with stiffness 1000 $\text{N} \cdot \text{m/rad}$ and viscous damping 30 $\text{N} \cdot \text{m} \cdot \text{s/rad}$

Table 4.1: Parameters of Planar Bipedal Robot.

Part	**Mass(kg)**	**Inertia(kg·m^2)**	**Length(m)**
Torso	19.8	0.300	0.4
Thigh	2.8	0.020	0.25
Shin	2.5	0.020	0.25
Sole	0.35	0.001	0.1

while the hip joint was torque controlled. For the support leg, the hip torque stabilizes the upper body posture. On the other hand, the hip torque of swing leg was used to ensure correct foot placement. The torque applied on hip joint was calculated by a PD controller with proportional and derivative coefficients of 500 $\mathrm{N \cdot m/rad}$ and 20 $\mathrm{N \cdot m \cdot s/rad}$. In this simulation, the four-link model with joint actuator dynamics included are adopted and the trajectories were generated by the proposed controller at the beginning of each step. The maximum iteration times N was 10, the tolerance ε for terminating the iteration of forward simulation was 0.05 rad/s, the period of one walking step was 0.4 s and the time step for forward simulation was 0.01 s.

The robot successfully walked through a stair of 5 cm height blindly in the sagittal plane. The movement was natural and compliant as Fig. 4-4 shows. The angular velocity $\dot{\theta}_{\mathrm{sp}}$ of virtual support leg and upper body posture ψ are given in Fig. 4-5 and 4-6 separately, where the solid blue line is for the measured data, and the red dash-dot line is the trajectory planned on-line for each step.

The Fig. 4-5 shows that the robot stepped up and down the stair at about 6 s and 9 s. When the robot was stepping up, the sole touched the ground earlier than expected, so the planning of next step was triggered immediately. Because of the limited torque of the under-actuated ankle joint and the early foot placement caused by disturbance, the angular velocity of following step was not tracked very well. However, for the next second step, a good foot placement was computed by the control algorithm based on the nonlinear models, and the robot was able to restore a stable state very quickly.

Similar situation happened during the step-down that the swing leg could not

land on the ground on time and the robot tried to keep the last planned joint position and waiting for landing. After landing with a natural leaning motion, the velocity could not be tracked well at this step because of the unexpected ground impact. Nevertheless, soon at the second step after the ground disturbance, the tracking performance resumed very well.

The upper body posture in Fig. 4-6 demonstrates a good tracking of real upright orientation with respect to on-line planned reference, although there are some small deviations from the desired values when stepping up and down. For the step right after the stair disturbance, the robot may not restore immediately. Whereas, the forward model is still capable of predicting the deviation accurately. This simulation indicates that the controller built on the forward model is very robust and is able to predict the motion of next step precisely after disturbance and recover to desired state within a very few steps.

4.3.2 Sagittal Push Recovery

To further study the robustness of disturbance rejection, a forward push recovery simulation is carried out on the same platform as the previous one. In this simulation, a push force of 100 N is applied in the center of upper body starting from 2 s and lasted for 0.1 s as shown in Fig. 4-7. The responses of angular velocity $\dot{\theta}_{\text{sp}}$ of virtual support leg and upper body posture ψ are given in Fig. 4-8 and 4-9. The definition of data lines in these figures is the same with the previous simulation.

When exerting an external force, a large error was produced immediately in the tracking of angular velocity of virtual support leg, because the external force was not modeled in the controller. But at the next step where the push has finished, by updating the prediction of the robot's movement using the proposed nonlinear model, the controller can restore the nominal gait by choosing a correct foot placement. For both of these two simulations, the upper body posture was tracked strictly due to the joint torque applied on the hip joint.

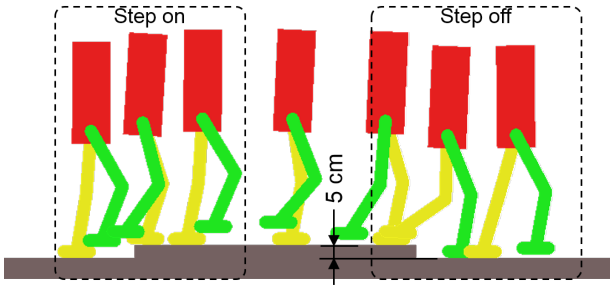


Figure 4-4: Snapshots of crossing a stair.

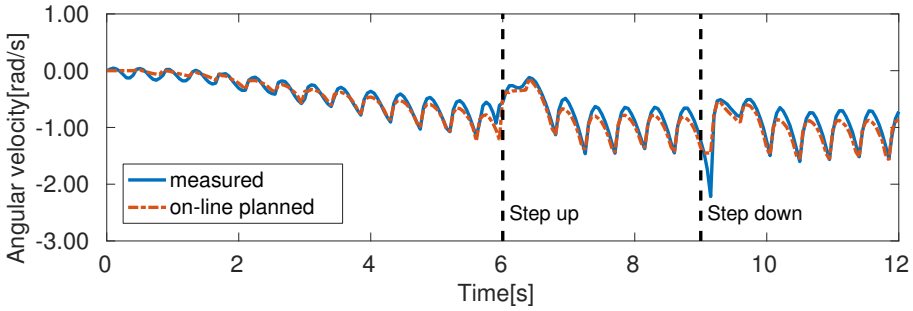


Figure 4-5: Angular velocity of virtual support leg when crossing a step.

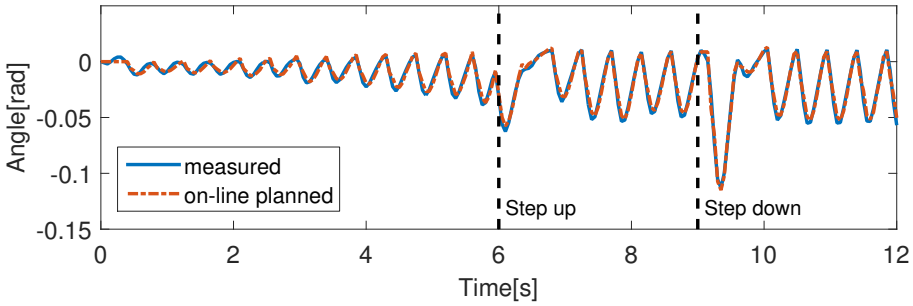


Figure 4-6: Upper body posture when crossing a step.

4.3.3 Lateral Push Recovery

The simulations above only concern about the movement in the sagittal plane. Here, the Open Dynamics Engine (ODE) was set up to validate the feasibility of the proposed controller in the lateral plane on a simulated humanoid robot developed previously in [22]. This robot has the same joint configuration and kinematics as the COMAN robot. Here all the joints of this robot are set to position control mode. According to the experience of using this robot before, the simulation results match quite well with the experimental data.

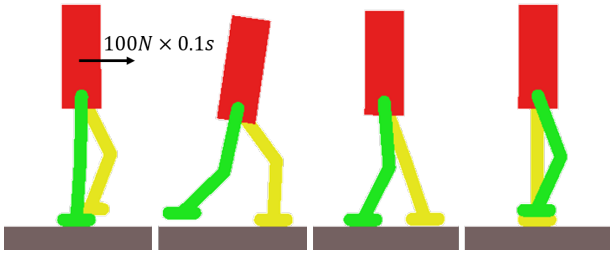


Figure 4-7: Snapshots of push recovery.

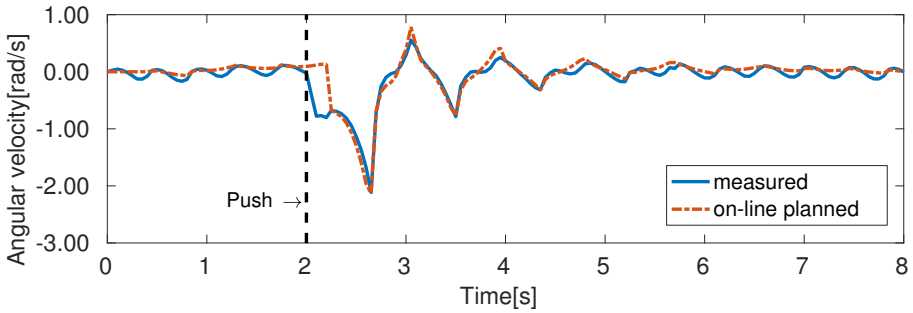


Figure 4-8: Angular velocity of virtual support leg in push recovery.

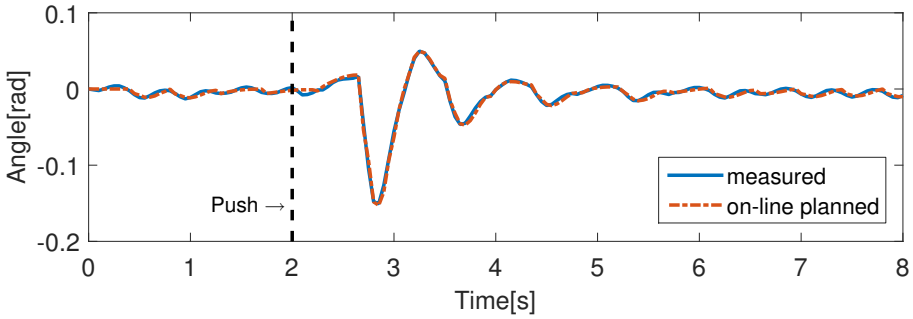


Figure 4-9: Upper body posture in push recovery

The same approach was applied by using the two-link variant as the forward model for the lateral push recovery. The robot walks in a 3-dimensional ODE simulation, meaning that the robot has no constraints in the sagittal plane. Therefore, a simple trajectory generator was implemented in sagittal plane to steer the ankle for keeping upper body upright and orient the swing foot to be parallel with the ground. For the lateral control based on the two-link model, the trajectories were re-planned whenever the tracking errors of θ_{sp} and $\dot{\theta}_{\text{sp}}$ exceeded limits of 0.1 rad and 1 rad/s or the planned trajectories were expired. The maximum iteration times N was 5, the tolerance ε was 0.02 rad/s, the duration of one walking step is 0.4 s, and the time step for forward

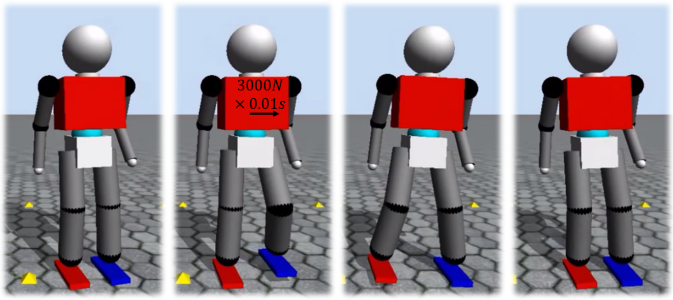


Figure 4-10: COMAN pushed by a lateral force.

simulation is 2 ms. First, a push force of 3000 N was exerted in the center of upper body for 0.01s as shown in Fig. 4-10. The angular velocities $\dot{\theta}$ of left and right virtual legs are shown in Fig. 4-11 and 4-12.

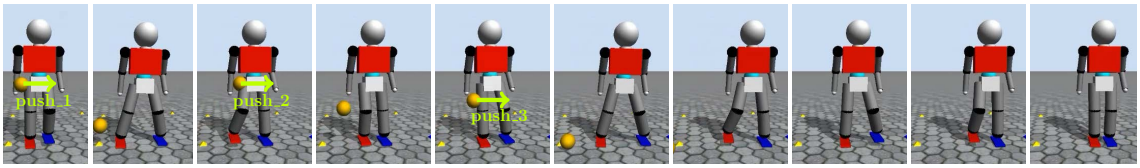


Figure 4-14: Snapshots of dynamic recovery from a series of lateral pushes in ODE (interval 0.25s).

In Fig. 4-10 to 4-12, the robot was pushed around the moment 1.5 s, and a big tracking error evoked in the trajectory tracking. The re-planned trajectories were able to guide the robot to stabilize after two steps. The tracking error was quite small when the robot was stepping in place stably.

Fig. 4-14 shows an additional aggressive test during which the robot was intermittently impacted three times by a ball with mass of 1 kg and initial velocity of 10 m/s. Nevertheless, the robot was able to sprawl legs aside and place the foot swiftly to stop falling over. So the control successfully dealt with these impacts and its good robustness was showcased by these trials. Note that in both cases shown by Fig. 4-10 and Fig. 4-14, the robot has very straight leg feature which is comparable to humans. Because the forward model a-priori considers the kinematics into the two-step lookahead simulation, the returned solution from the iterative search (Algorithm 1) is always physically realizable. Hence, there is no knee singularity issue by principle.

The coded algorithm in C++ runs each query of the iterative planning for less than 0.5 ms, which is very efficient as the computational time shown in Fig. 4-13. This is entirely feasible for real-time implementation.

4.4 Summary

This chapter explored the forward model concept inspired by the sensorimotor control realm and applied the principle to the control of robust and dynamic bipedal walking. First, good candidates of nonlinear models are studied for forward simulation, four-link and two-link models, which are suited to meet different requirements of sagittal and lateral gaits towards the minimum computation without downgrading much of the accuracy. These models can also include the characteristics of the control system, i.e. close loop control and actuator dynamics. Based on these nonlinear models, multiple shooting methods with gradient descent search inside is utilized to best exploit forward simulation and automatically search for precise foot placement. With the proposed method, a planar bipedal robot was able to blindly travel over a stair and recover from a push successfully in simulation. Furthermore, the simulated COMAN robot was able to withstand several aggressive impact attacks without falling over.

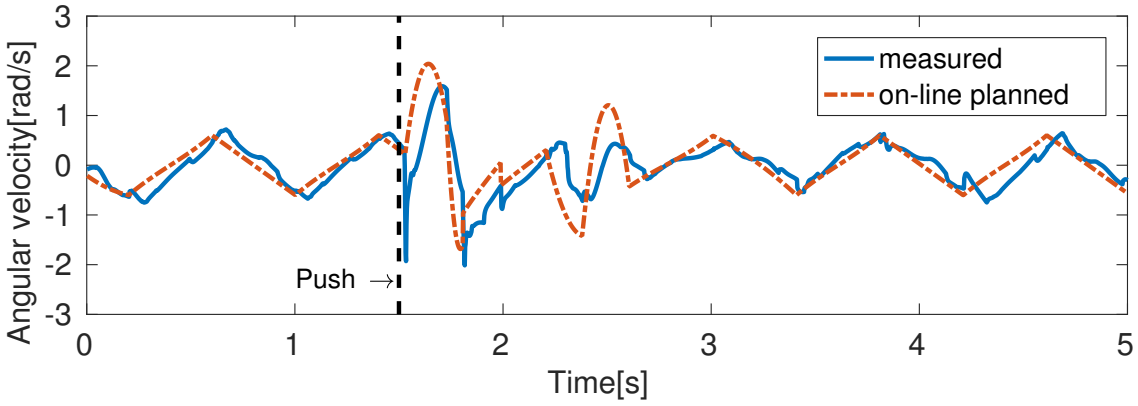


Figure 4-11: Angular velocity of left virtual leg.

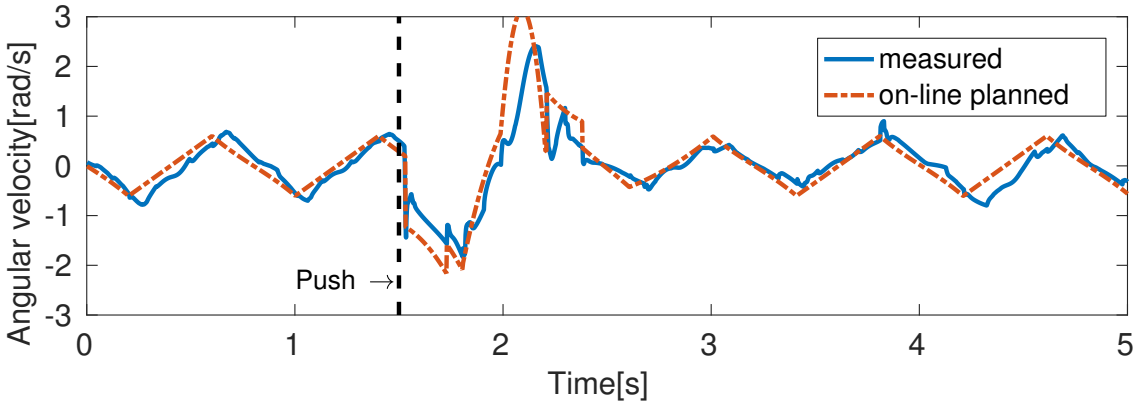


Figure 4-12: Angular velocity of right virtual leg.

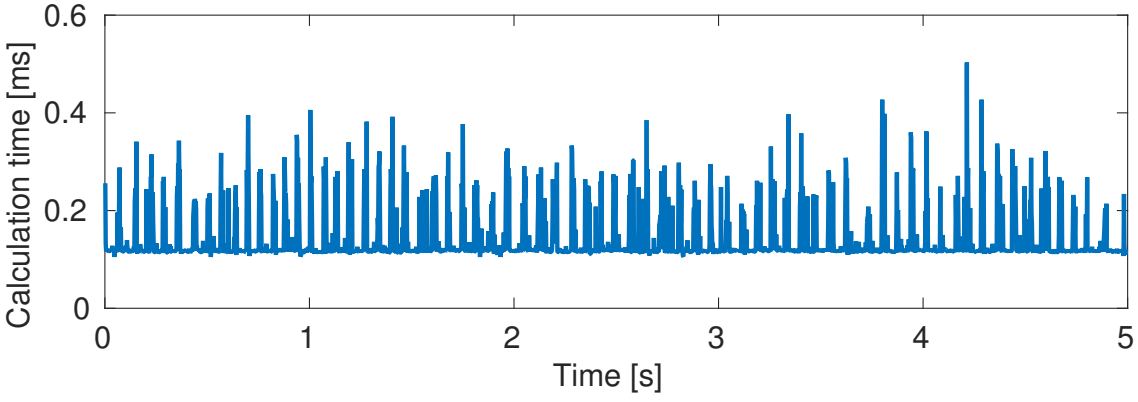


Figure 4-13: Time consuming of controller.

Chapter 5

Low Level Controller for Stabilization and Motion Tracking

Previous chapters focused on the walking pattern generation. Another important aspect for dynamic walking is stabilization and motion tracking. Compared with the high level planning of motion pattern, the stabilization and motion tracking is named low level controller here. In the low level controller, not only the relative motion of each joint should be tracked, but also the whole body movement with respect to the world should be stabilized. According to different types of joint actuators, three approaches of low level controllers are implemented here: CoM stabilizer for the robots with position controlled joints, quasi-static impedance control and optimization-based full body torque control for the torque-controlled robots.

5.1 Position Control and Stabilizer

A lot of existing humanoid robots, like well-known ASIMO[53], HRP-3[29] and NAO[56], have position-controlled joints, where each joint is actuated individually and relatively easy to implement in practice. To achieve accurate joint tracking, the stiffness and damping of the joints in position control mode is usually set very high. This distributed control strategy without considering the effect between each other may work well in the structural known environment, but easily falls into failure when the ground

is not that perfect. In this case, a CoM stabilizer is preferred to improve the overall tracking stability.

$$\begin{aligned} \Delta\dot{\mathbf{C}} &= k\frac{\mathbf{C}_d - \mathbf{C}}{dT} \\ \mathbf{q}_{revised} = \mathbf{q}_d + dT * \Delta\dot{\mathbf{q}} &= \mathbf{q}_d + dT * J^+_{com}\Delta\dot{\mathbf{C}} = \mathbf{q}_d + kJ^+_{com}(\mathbf{C}_d - \mathbf{C}) \end{aligned} \tag{5.1}$$

In the equation above, $\mathbf{C}$ and $\mathbf{C}_d$ are the measured and desired CoM position, dT is the period of the low level position control loop, k is the feedback coefficient, $\mathbf{q}_d$ is the desired joint position solved from high level planner by inverse kinematics, and $\mathbf{q}_{revised}$ is the real joint position command sent to the joint actuator after revised by the CoM stabilizer. The adjustment quantities of CoM velocity $\Delta\dot{\mathbf{C}}$ and joint velocity $\Delta\dot{\mathbf{q}}$ are related to each other through the sudo-inverse of CoM jacobian J^+_{com}.

The stabilizer tries to eliminate the CoM tracking error of the whole robot by adjusting the joint position through inverse kinematics. It can work well if the joint stiffness is very high. However, one side-effect brought by the high stiffness joints is that even with the stabilizer, if the modeling of the ground or the tracking is not accurate enough, the big landing impact caused by the error will cause drastic changing in the robot state and result inevitable falling. Therefore, quasi-static impedance control and torque control methods are implemented for accurate tracking and compliant behavior of humanoid locomotion based on the improved hardware of joint actuator.

5.2 Quasi-static Cartesian Impedance Control

Impedance control has been widely applied to the interaction between robot and environment due to its ability of regulating the contact force. Considering the difficulty of knowing the exact contact state between feet and ground, quasi-static Cartesian impedance control is utilized here as the low level controller for the local trajectory tracking. "Local" means that the trajectory is for the feet with respect to the local pelvis frame.

5.2.1 Formulation

Quasi-static Cartesian impedance control has been proposed for quite a long time, a brief introduction is given to cover what is used here.

$$\begin{aligned} \mathbf{F}_d &= \mathbf{K}_{cp} \cdot (\mathbf{X}_d - \mathbf{X}) - \mathbf{K}_{cd} \cdot \dot{\mathbf{X}} \\ \boldsymbol{\tau}_d &= \mathbf{J}^T \cdot \mathbf{F}_d \end{aligned} \tag{5.2}$$

where $\mathbf{X}$ and $\mathbf{X}_d$ are individually the measured and desired pose of foot, $\mathbf{K}_{cp}$ and $\mathbf{K}_{cd}$ are the assigned stiffness and damping in the Cartesian space. $\mathbf{F}_d$ is the desired wrench at the end-effector while $\boldsymbol{\tau}_d$ is the required joint torque to generate the desired wrench $\mathbf{F}_d$ under quasi-static condition. $\mathbf{J}$ is the Jacobian of foot with respect to the pelvis, whose transpose associates the end-effector wrench with the joint torque.

To make full use of the high frequency bandwidth of joint position control and improve the tracking stability, a combined position and torque control was utilized for each joint:

$$\boldsymbol{\tau} = \boldsymbol{\tau}_d + \mathbf{K}_{jp} \cdot (\mathbf{q}_d - \mathbf{q}) - \mathbf{K}_{jd} \cdot \dot{\mathbf{q}} \tag{5.3}$$

where $\boldsymbol{\tau}$ is the joint toque command sent to the motor, $\boldsymbol{\tau}_d$ is the joint torque calculated by the impedance control, $\mathbf{K}_{jp}$, $\mathbf{K}_{jd}$ are the designed stiffness and damping for the joint position control, $\mathbf{q}$, $\dot{\mathbf{q}}$ are the measured joint position and velocity, and $\mathbf{q}_d$ are the desired joint position calculated by numerical inverse kinematics.

5.2.2 Walk Like A Inverted Pendulum

As an example of utilizing the impedance control, a walking pattern similar to Inverted Pendulum Model (IPM) was realized successfully on the upper body of CENTAURO robot as shown in Fig. 5-1. The arms of the CENTAURO robot were treated as legs resulting to a big ratio of feet separation distance to CoM height. And the sticks at the end of the two arms worked as narrow feet. During double support phase, the feet separation distance is 0.35 meter while the CoM height is around 0.4 meter.

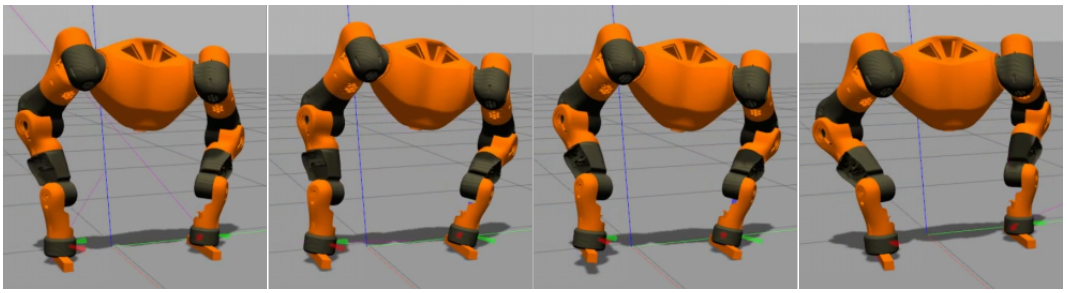
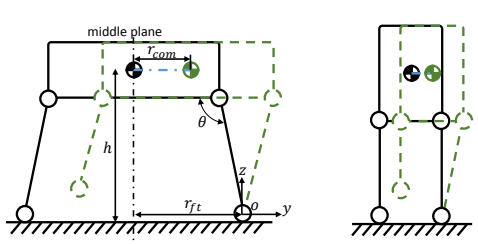
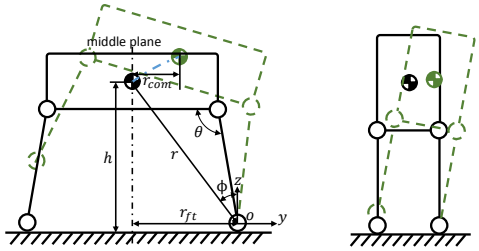


Figure 5-1: Snapshots of CENTAURO walking (interval: 0.2 second).



(a) LIPM motion pattern under different feet distance (left: large feet distance, right: small feet distance).



(b) IPM motion pattern under different feet distance (left: large feet distance, right: small feet distance).

Figure 5-2: Different motion patterns with narrow feet in lateral plane.

Analysis

Compared with the narrow feet walking introduced in Section 3.3.2, the most important difference is the walking pattern, one following the LIPM and the other following the IPM. Before implementing the quasi-static Cartesian impedance control in details, the pros and cons of the motion patterns based on these two models are analyzed first. Since only the ankle roll joint is not actuated actively due to the narrow feet, the analysis will be constrained in the lateral plane. Observing bipedal walking of animals, it seems that the bigger width-height ratio, the fiercer the oscillation of the animal's upper body. For instance, bears and gorillas tend to shake their upper bodies more than human during walking, which looks more like IPM instead of LIPM. To understand the physical principles behind them, the characteristics of these two walking motion patterns are studied.

In the motion pattern generated by LIPM, the CoM height is expected to be constant, and the pelvis is always kept upright as shown in Fig. 5-2 (a). On the other

hand, for the motion generated by IPM, the whole body is considered as one rigid body and rotate together around the support foot during walking (see Fig. 5-2 (b)). As mentioned before, the analysis will be given in the lateral plane.

The first thing to take care is the CoM displacement of these two models when walking with the same feet separation distance (shorten as feet distance below), initial CoM height and step duration. Considering the most common walking style, symmetric walking, robot motions are symmetric about the middle plane of two feet (below called as middle plane). And CoM displacement r_{com} is the farthest distance of CoM to the middle plane when its velocity decrease to zero. r_{ft} represents half of the feet distance.

Based on the differential equations $\ddot{y} = \frac{g}{h}y$, $\ddot{\phi} = \frac{g}{r}\sin\phi$ of LIPM and IPM, $y = f(h_0, y_0, \dot{y}_0, t)$ and $\dot{y} = f'(h_0, y_0, \dot{y}_0, t)$ can be numerically solved to get the CoM state after a duration t, where g is the gravity acceleration, h is the constant CoM height of LIPM and r is the constant CoM rotation radius of IPM, h_0, y_0 and $\dot{y}_0$ are the initial height, horizontal position and velocity of CoM. Assuming stable symmetric walking, the CoM should pass through the middle plane with an equivalent but opposite velocity after one walking step. Therefore the CoM displacement r_{com} can be calculated as follow:

$$\begin{aligned} r_{com} &= r_{ft} - |f(h_0, r_{ft}, \dot{y}_0, T/2)| \\ s.t. \quad & f'(h_0, r_{ft}, \dot{y}_0, T) = -\dot{y}_0 \end{aligned} \tag{5.4}$$

where T is the time of one walking step.

Given feet distance r_{ft} and initial CoM height h_0, the CoM height of LIPM and the rotation radius of IPM can be derived individually as $h = h_0$, $r = \sqrt{h_0^2 + r_{ft}^2}$. Further, by setting the walking step time $T = 1$s, the farthest CoM displacement of these two models could be figured out by equation (5.4) with respect to different feet distance (0.02 to 1.0 meter) and initial CoM height (0.6 to 1.0 meter) as Fig. 5-3 shows.

The results agree with the intuition: CoM displacement gets larger with increasing feet distance and it applies to both models. Comparing to IPM, the CoM displacement

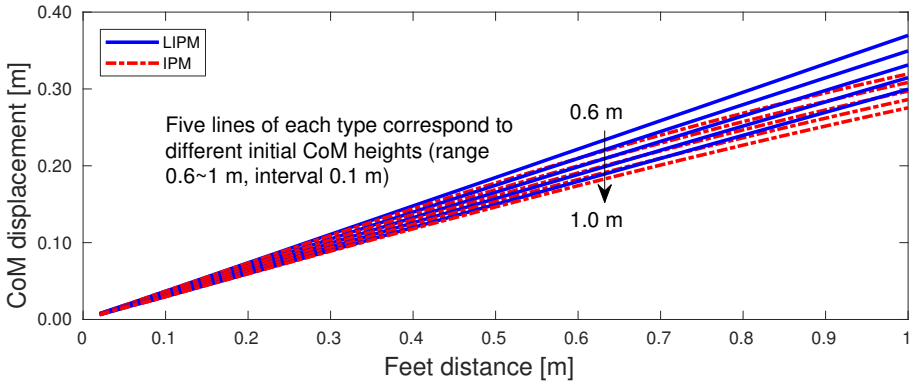


Figure 5-3: CoM displacement at different feet distances and CoM heights.

of LIPM increases faster. This would increase the risk of reaching hip roll joint limits for the robot. Besides, one extra benefit of IPM motion pattern is that the pelvis rotates together with the legs, which results in even less movement of hip roll joint during walking. Considering this, using IPM to generate walking motion patterns is a wiser choice when large feet distance involved.

Another issue to be noted is that for bipedal robots with articulated joints, the bent leg walking of LIPM motion usually consumes more energy than the IPM case in which the leg can be more stretched. You can get an intuitive feeling about this by keeping your hip at a constant height during walking. You will get tired very soon, especially for the knee joints.

Despite of the shortages discussed above, the constant CoM height of LIPM motion pattern can be helpful for mitigating the ground impact and therefore reducing energy loss. This is inevitable for robots with the IPM motion pattern unless energy storage elements are implemented. To figure out the relationship among ground impact, feet distance and initial CoM height, a series of numerical calculations were performed whose results are shown in Fig. 5-4.

The setup of numerical calculation was the same with previous section. The duration of one walking step was 1 second. Given a initial CoM height and feet distance, the angular velocity of CoM passing the middle plane can be calculated out iteratively for IPM motion, which is unique for symmetric walking. And the vertical component of this velocity is used to evaluate the ground impact, called CoM impact

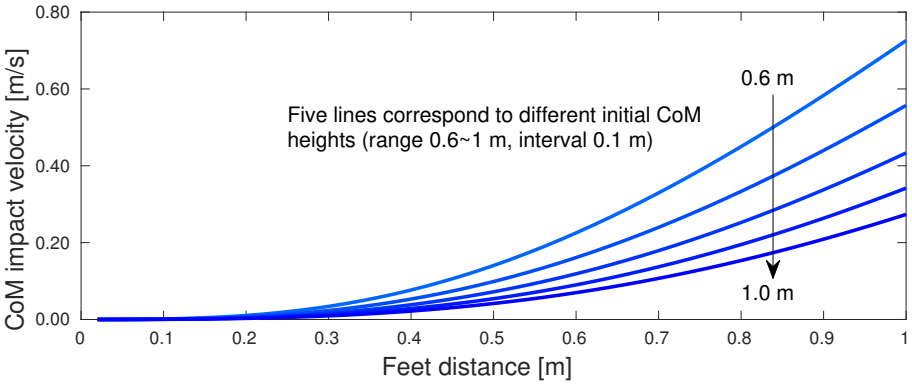


Figure 5-4: CoM impact velocity at different feet distances and CoM heights.

velocity here, because it would reverse its direction instantly along with energy loss at that moment of support foot switch.

As shown in Fig. 5-4, the impact gets more gentle and the CoM impact velocity decreases if the feet distance is smaller. Meanwhile, higher initial CoM can also alleviate the ground impact.

Validation

To validate the feasibility of the quasi-static Cartesian impedance control and conform the analysis of comparing LIPM and IPM, walking simulation is performed by using the upper body of the Centauro robot. Instead of planning the CoM trajectory in the global world frame, here the leg motion in the pelvis local frame is taken care more so that the whole robot is able to rotate around the ankle roll joint of the supporting feet passively. A state machine is introduced to describe the designed leg motion. It includes four phases: landing, stretching, retracting and holding as shown in Figure 5-5.

Landing phase starts from the first moment that the two feet are both on the ground and ends when one leg starts stretching. During this phase, the robot will try to keep its current configuration aiming to absorb the oscillation of the pelvis after the ground impact. Then, the stretching phase happens, during which, one leg stretches and makes the robot rotate around the other foot. During this phase, the stretch motion will inject energy to the robot system, and at the end of this phase,

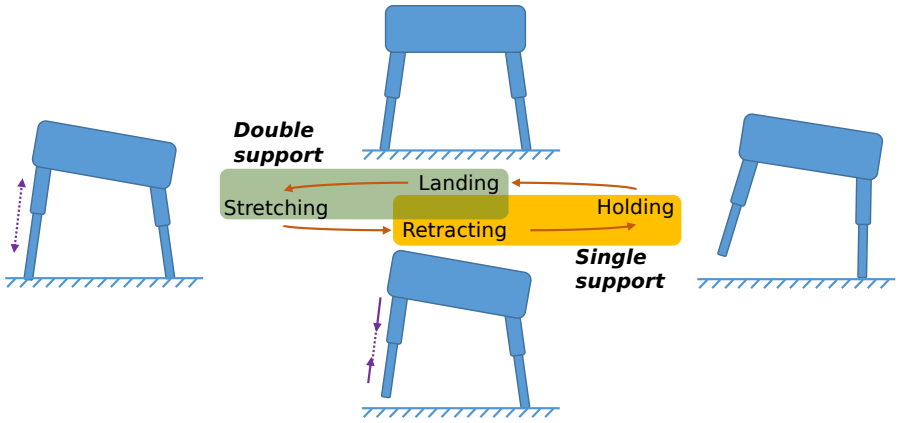


Figure 5-5: State machine of CENTAURO walking pattern.

the stretch leg may leave off the ground. Thereafter, the stretching leg will retract to prepare for the next landing, called as retracting phase. Meanwhile, both legs will move in forward direction to make the robot walk or turn. At the end of this walking cycle, the period between the retracting phase and the subsequent landing is named as holding phase, when the robot waits for landing.

Fig. 5-6 shows the desired and measured trajectories of two feet in the local frame with respect to the pelvis, where the red and green lines are for the left and right feet, and the solid and dash lines are individually for the measured and desired data. The contact state is indicated by the purple dash line with three states: left foot single support (LS), right foot single support (RS) and double support (DS). As stated above, the leg would stretch first before swinging and then retract to the normal length which can be seen in Fig. 5-6 (c). In the double support phase, stretching state dominated most of the period, and landing state was very short to make the walking motion continuous and smooth. Thereafter, the swinging leg started retracting and the contact state switched to single support. Meanwhile, both of the feet will move along X direction to drive the robot forward as shown in Fig. 5-6 (a). Because the robot is not expected to move sideways, so the feet was fixed in the Y direction (see Fig. 5-6 (b)). Since gravity compensation was not taken into account, the tracking error of supporting leg is quite obvious during single support phase. This can be further improved in the future.

The measured global trajectories of CoM and two feet in the world frame are

shown in Fig. 5-7. The global desired trajectories are missing, because the motion planning is performed in the local frame. The robot moved forward with a little lateral drift (see Fig. 5-7 (a,b)). CoM height oscillation and ground impact were observed from Fig. 5-7 (c) as expected.

Comparing to the simulation data in Section 3.3.2, the characteristics of these motion patterns are distinct to each other. For the motion pattern of LIPM, the CoM can be controlled at a certain height all the time resulting less ground impact at landing. Its shortcoming is that the required big movement of hip roll joint is not always possible. Despite that the CoM height of WALK-MAN is double of that of CENTAURO, its hip roll joint movement is still much bigger. On the contrary, IPM motion pattern can mitigate this issue a lot. As the price, the CoM height of IPM motion would oscillate, which produces extra energy loss due to the ground impact.

5.3 Optimization-based Full Body Torque Control

Walking is actually a multi-task motion. It involves Cartesian space trajectory tracking, body posture regulation while maintaining dynamic balance. While dealing with multiple tasks, traditional null-space projection based techniques could be applied to solve the problem in a hierarchical manner [54, 55]. But this analytical techniques can not properly handle inequality constraints, such as torque limit and friction cone limit. Researchers turn to numerical method which is better at considering different constraints. Although detailed formulations differ, most of approaches formulate the floating base inverse dynamics as a quadratic programming (QP) problem with equality and inequality constraints [7, 10, 19, 22, 62].

5.3.1 Formulation

In the optimization-based full body torque control method, the low level stabilization and motion tracking is formulated as a quadratic optimization problem to calculate the joint torques according to given tasks with respect to constraints, such as dynamic feasibility, friction cone, torque limits.

Quadratic formulation is adopted to solve whole body dynamics. Different weights are used to balance multiple tasks in the cost function without considering strict priorities among them. It is simple to implement and also numerically robust. Hard constraints such as joint torque limits and friction cone limits are formulated as inequality constraints. The details are given below, starting from the Equation of Motion (EoM) of the whole robot:

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{h}(\mathbf{q}, \dot{\mathbf{q}}) = \mathbf{S}^{\mathrm{T}}\boldsymbol{\tau} + \mathbf{J}_c^{\mathrm{T}}(\mathbf{q})\boldsymbol{\lambda} \tag{5.5}$$

with the inertia matrix $\mathbf{M}(\mathbf{q})$, the force vector $\mathbf{h}(\mathbf{q})$ which is sum of Coriolis, centrifugal and gravitational forces and the ground reaction force $\boldsymbol{\lambda}$. $\mathbf{J}_c^{\mathrm{T}}$ is corresponding Jacobian, $\boldsymbol{\tau}$ is joint torque, $\mathbf{q}$ represents the n degrees of freedom (DoF) generalized coordinates which include base and body joint coordinate $\mathbf{q} = [\mathbf{q_f^{\mathrm{T}}}, \mathbf{q_r^{\mathrm{T}}}]^{\mathrm{T}}$, and $\mathbf{S} = [\mathbf{0_{n_r \times n_f}}, \mathbf{I_{n_r}}]$ is a selection matrix which separates the $n_r = n - n_f$ actuated joints from the $n_f = 6$ floating-base DoFs.

EoM (5.5) relates generalized acceleration $\ddot{\mathbf{q}}$, contact forces $\boldsymbol{\lambda}$ and joint torques $\boldsymbol{\tau}$ together. We choose $\mathbf{X} = [\ddot{\mathbf{q}}^{\mathrm{T}}, \boldsymbol{\lambda}^{\mathrm{T}}]^{\mathrm{T}}$ as optimization variables for the following QP problem :

$$\min_{\mathbf{X}} \sum_{i=1}^{n} \frac{\omega_i}{2} ||\mathbf{A_i X} - \mathbf{b_i}||^2 \tag{5.6}$$

subject to

$$\mathbf{M_f}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{h_f}(\mathbf{q}, \dot{\mathbf{q}}) = \mathbf{J}_{\mathrm{cf}}^{\mathrm{T}}(\mathbf{q})\boldsymbol{\lambda} \tag{5.7}$$

$$\boldsymbol{\tau} = \mathbf{S}(\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{h}(\mathbf{q}, \dot{\mathbf{q}}) - \mathbf{J}_{\mathrm{c}}^{\mathrm{T}}(\mathbf{q})\boldsymbol{\lambda}) \in [\boldsymbol{\tau}_{\min}, \boldsymbol{\tau}_{\max}] \tag{5.8}$$

$$\mathbf{J}_{\mathrm{c}}\ddot{\mathbf{q}} + \dot{\mathbf{J}}_{\mathrm{c}}\dot{\mathbf{q}} = \mathbf{0} \tag{5.9}$$

$$|\frac{f_x}{f_z}| \leq \mu, \ |\frac{f_y}{f_z}| \leq \mu \tag{5.10}$$

$$d_x^- \leq \frac{m_y}{f_z} \leq d_x^+, \ d_y^- \leq -\frac{m_x}{f_z} \leq d_y^+ \tag{5.11}$$

The objective function (5.6) tries to minimize the tracking error of different tasks, but their relative importance is decided by corresponding weight ω_i. Each task is

defined by corresponding matrix $\mathbf{A}_{\text{task}}$ and vector $\mathbf{b}_{\text{task}}$.

The constraints (5.7) and (5.8) ensure the dynamics feasibility and joint torque availability, the subscript f in (5.7) stands for the six DoFs of floating base. (5.9) makes sure there is no slip in contact points. The contact wrench can be expressed as: $\boldsymbol{\lambda} = [f_x, f_y, f_z, m_x, m_y, m_z]^T$. The nonlinear friction cone is approximated by a linear polyhedral cone in which constraint (5.10) makes the contact force stay. (5.11) restricts ZMP stay inside support polygon which is defined by the limits $[d_x^-, d_x^+]$ and $[d_y^-, d_y^+]$.

The tasks usually involved in walking includes: motion tasks (regulating CoM position or tracking end-effectors' spacial trajectory), contact force tasks (optimizing contact force distribution) and joint torque tasks (assigning joint torques). Below are the details about how to formulate each type of task.

Motion tasks

As the most common tasks for robots, two examples are given: CoM trajectory tracking and end-effector trajectory tracking.

For CoM tracking, considering the centroidal dynamics [45], the system's linear momentum $\mathbf{P}$ and angular momentum $\mathbf{L}$ is linear with the generalized velocity $\dot{\mathbf{q}}$:

$$\begin{bmatrix} \mathbf{P} \\ \mathbf{L} \end{bmatrix} = \mathbf{H}(\mathbf{q})\dot{\mathbf{q}} \tag{5.12}$$

with $\mathbf{H}$ is called the centroidal momentum matrix. Taking derivative of this equation will give:

$$\begin{bmatrix} \dot{\mathbf{P}} \\ \dot{\mathbf{L}} \end{bmatrix} = \mathbf{H}\ddot{\mathbf{q}} + \dot{\mathbf{H}}\dot{\mathbf{q}} \tag{5.13}$$

It is obvious that the changing rate of momentum $\dot{\mathbf{P}}$ and $\dot{\mathbf{L}}$ is linear function of $\ddot{\mathbf{q}}$. As a result, the task matrix below could be used to track desired changing rate of momentum:

$$\mathbf{A}_{\text{H}} = [\mathbf{H},\ \mathbf{0}],\ \mathbf{b}_{\text{H}} = \begin{bmatrix} \dot{\mathbf{P}}_{ref} \\ \dot{\mathbf{L}}_{ref} \end{bmatrix} - \dot{\mathbf{H}}\dot{\mathbf{q}} \tag{5.14}$$

Typically, reference changing rate of momentum could be defined as:

$$\begin{bmatrix} \dot{\mathbf{P}}_{ref} \\ \dot{\mathbf{L}}_{ref} \end{bmatrix} = \begin{bmatrix} \dot{\mathbf{P}}_{des} \\ \dot{\mathbf{L}}_{des} \end{bmatrix} + \mathbf{K}_p \begin{bmatrix} \mathbf{c}_{des} - \mathbf{c} \\ \mathbf{0} \end{bmatrix} + \mathbf{K}_{\mathrm{d}} \begin{bmatrix} \mathbf{P}_{des} - \mathbf{P} \\ \mathbf{L}_{des} - \mathbf{L} \end{bmatrix} \tag{5.15}$$

with $\mathbf{K}_p$ and $\mathbf{K}_d$ the gains of the PD feedback controller.

Trajectory tracking for end-effector in Cartesian space is formulated as:

$$\mathbf{A}_{cartesian} = [\mathbf{J},\ \mathbf{0}],\ \mathbf{b}_{cartesian} = \ddot{\mathbf{x}}_{ref} - \dot{\mathbf{J}}\dot{\mathbf{q}} \tag{5.16}$$

with $\mathbf{J}$ the spacial jacobian matrix corresponding to the frame attached to the robot. $\ddot{\mathbf{x}}_{\mathrm{ref}}$ is the reference spacial acceleration which can be calculated with:

$$\ddot{\mathbf{x}}_{ref} = \ddot{\mathbf{x}}_{des} + \mathbf{K}_p(\mathbf{x}_{des} - \mathbf{x}) + \mathbf{K}_d(\dot{\mathbf{x}}_{des} - \dot{\mathbf{x}}) \tag{5.17}$$

where $\mathbf{x}_{des}$, $\dot{\mathbf{x}}_{des}$ and $\ddot{\mathbf{x}}_{des}$ are desired end-effectors' position, velocity and acceleration.

Contact force tasks

In order to realize compliant interaction between the robot and environment, contact force tasks can be formulated to achieve desired contact forces $\boldsymbol{\lambda}_{des}$.

$$\mathbf{A}_{force} = [\mathbf{0},\ \mathbf{I}],\ \mathbf{b}_{force} = \boldsymbol{\lambda}_{des} \tag{5.18}$$

Joint torque tasks

To directly control joint torque, the task could be formulated as:

$$\mathbf{A}_{\tau} = \mathbf{S}[\mathbf{M},\ \mathbf{J}_c^T],\ \mathbf{b}_{\tau} = \boldsymbol{\tau}_{des} - \mathbf{Sh} \tag{5.19}$$

with $\boldsymbol{\tau}_t extitdes$ is the desired joint torque vector.

5.3.2 Balance with Arms

Based on the optimization-based full body torque control, the robot is able to handle multiple tasks at the same time and do trade-off among them by tunning the weights. One typical example to demonstrate this for humanoid robots is to balance with full bodies, especially with arms, just like human does. Here the torque control framework introduced above was implemented on the WALK-MAN robot. Several tasks were assigned to achieve human-like full body balancing:

- Keep CoM in a desired position whose ground projection should be inside the support polygon;
- Maintain a fixed configuration of upper body;
- Keep the pelvis upright;
- Regulate the centroidal angular momentum to zero;
- Distribute contact forces according to the distance between the feet and CoM.

Different weights were assigned to these tasks. Compared with others, keeping CoM staying in the desired position had higher weight, otherwise the robot would tend to fall if the ground projection of CoM moves out of the support polygon. Keeping pelvis upright is a task defined in the global world frame while maintaining a fixed configuration of upper body is defined in the local joint frame. Both of them were considered more important than regulating angular momentum and distributing contact forces. One thing should be noted that although the weight of maintaining a fixed configuration of upper body is high, the weight for holding the shoulder joints in place (part of the fixed configuration task) was set quite low intentionally so that the robot will make more use of these joints to keep balancing. With above settings, the balancing behavior of WALK-MAN robot is quite like a human. Fig. 5-8 is the snapshot when WALK-MAN was pushed forward from behind on the pelvis.

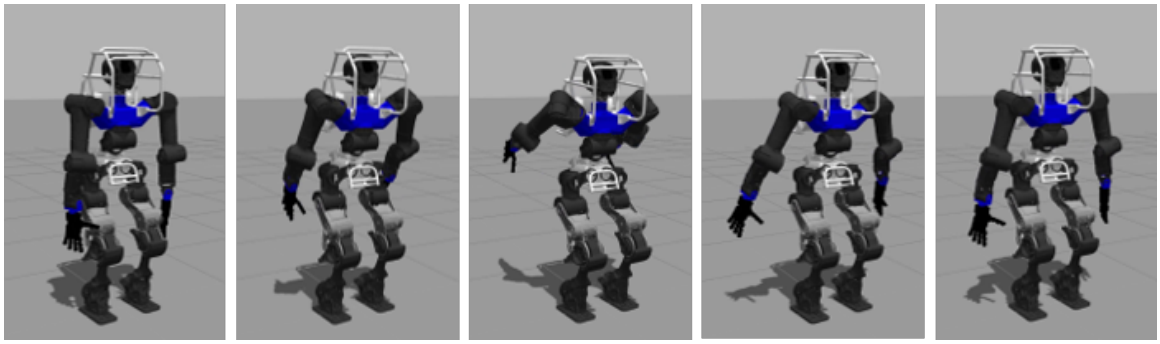


Figure 5-8: WALK-MAN used full bodies to keep balancing when a push was exerted on the waist from behind(interval 0.8s).

5.3.3 Walk with Straight Leg

Many existing humanoid robots walk in an unnatural way with bent knees due to the use of LIPM which constrains CoM in a horizontal plane, which results in high knee joint torque and extra energy consumption. To address this issue, a simple yet efficient control strategy is proposed here to realize straight leg walking based on the optimization-based full body torque control. First, theoretical analysis on a simplified model provided insight into ZMP deviations during straight knee walking. After observing that the deviation is limited comparing to the support polygon, the high level planning part keeps the same with the Section 3.2.1, but the robot is able to perform straight leg walking automatically by utilizing the optimization-base full body torque controller. By setting the desired CoM height slightly over the robot's reachable height, the low level controller will attempt to straighten the robot's leg to reach this vertical reference, in the meanwhile, also satisfy the constraints (i.e. dynamic feasibility, friction cone, torque limits).

Analysis

For on-line planning, a simplified, especially linear model is preferred to provide a longer preview horizon. Nevertheless, it is difficult to consider complicated constraints, such as the kinematic constraint of legs. To realize straight leg walking, the idea here is using LIMP for planning but considering the kinematic constraint in

low-level controller which involves the full body dynamic model of the robot. More detailedly, the predefined height of LIPM is set a bit higher than the maximum reachable one. And then the low level controller will try its best to stretch the leg to track the desired height but still meet the constraints. To evaluate its feasibility, ZMP deviation caused by the proposed control strategy is analyzed below.

Assuming the reference CoM trajectory is planned based on LIPM with constant desired CoM height h and desired ZMP at the center of foot $P_{x,y} = 0$. The resulting CoM dynamic could be derived by substituting h and $\ddot{C}_z = 0$ to equation ((3.3)):

$$\begin{cases} \ddot{C}_{x,y} = \dfrac{g}{h} C_{x,y} \\ \ddot{C}_z = 0 \end{cases} \tag{5.20}$$

The ordinary differential equations (5.20) have analytic solutions [24]:

$$\begin{gathered} C_{x,y} = C_{x,y}(0)\cosh(t/T_c) + T_c \dot{C}_{x,y}(0)\sinh(t/T_c) \\ T_c = \sqrt{h/g} \end{gathered} \tag{5.21}$$

Where t is the time, $C_{x,y}(0)$ and $\dot{C}_{x,y}(0)$ are the initial position and velocity of CoM. To further simplify the analysis, the CoM motions in sagittal and lateral plane are considered separately. As mentioned above, in order to make the robot walk with straight leg, the desired CoM height h will be set higher than the maximum reachable one r. And this would encourage the robot to stretch the legs as much as possible. In this case, the resulting CoM motion achieved by the low level controller will be an arc. So the CoM position and acceleration along z direction are:

$$\begin{gathered} C_z = \sqrt{r^2 - (C_{x,y})^2} \\ \ddot{C}_z = -\frac{C_{x,y}\ddot{C}_{x,y}}{C_z} - \frac{(\dot{C}_{x,y})^2}{C_z} - \frac{(C_{x,y}\dot{C}_{x,y})^2}{(C_z)^3} \end{gathered} \tag{5.22}$$

It should be noted that actually r is not constant and will change when the robot is moving like lifting its swing leg. But here for simplicity, it is assumed constant for

analysis. Substitute equations (5.22) into the definition of ZMP,

$$P_{x,y} = C_{x,y} - \frac{C_z}{\ddot{C}_z + g}\ddot{C}_{x,y} \tag{5.23}$$

Considering $P_{x,y} = 0$ for LIPM, the ZMP deviation caused by the CoM vertical variation is

$$\Delta P_{x,y} = C_{x,y} - \frac{(C_z)^4 \ddot{C}_{x,y}}{(gC_z - C_{x,y}\ddot{C}_{x,y} - (\dot{C}_{x,y})^2)(C_z)^2 - (C_{x,y}\dot{C}_{x,y})^2} \tag{5.24}$$

According to equation (5.21), (5.22), (5.24), the deviation of ZMP is related to the time t, initial position $C_{x,y}(0)$ and velocity $\dot{C}_{x,y}(0)$ of CoM, CoM's maximum reachable length r and the desired CoM height C_{z0}. To ensure the knees stretching straight during walking, h should not be smaller than r. Here set $h = r = 1$ m which is similar to the CoM height of WALK-MAN robot in its static standing posture. Then the ZMP deviations can be calculated in sagittal and lateral planes by setting typical initial states. The initial position and velocity for the sagittal plane are -0.3 m and 0.98 m/s while the ones for lateral plane are -0.16 m and 0.48 m/s when left leg is supporting alone. Fig. 5-9 and 5-10 show the ZMP deviations in the two directions. Blue solid line is the CoM trajectory and red dash-dot line is the ZMP deviation. The maximum ZMP deviation in lateral plane is less than 1 cm, quite small compared with foot width 16 cm while the one in sagittal plane is around 6 cm. However since the foot size is also longer (30 cm) in sagittal plane, the deviation is acceptable. This analysis result indicates that it is possible to plan CoM trajectory via LIPM with an unreachable CoM height and then try to consider the constraints neglected by the simplified model in the low level controller.

Validation

To evaluate the effectiveness and performance, the proposed control strategy was tested on the full body of WALK-MAN robot [60] in ROS-Gazebo simulation environment as shown in Fig. 5-11.

In this simulation, the high level planner used MPC to generate the CoM and foot trajectories first. Since the CoM height during straight standing was around 1.15 m, therefore the desired CoM height of LIPM was set to 1.17 m to let the low level controller enforce leg straightening during walking. The period of each walking step was 1.5 second, and no double-support phase was considered except the starting and ending steps. The foot placements were determined automatically by the MPC planner, but for the ending step, the foot placement was set the same with the starting step in Y direction. Detailed trajectories generated by the high level planner can refer to Fig. 5-12.

The objectives set for the low level controller in this simulation were to track the CoM and foot trajectories, and keep upper body upright. At the same time, the low level controller also needed to guarantee that the kinematic and dynamic constraints were satisfied. So even when the trajectories generated by the high level planner were not tracked very strictly, the ZMP constraints would still be satisfied by the low level controller. Here the desired CoM height is set a little higher than maximum and the low level controller would try to track it as well as possible.

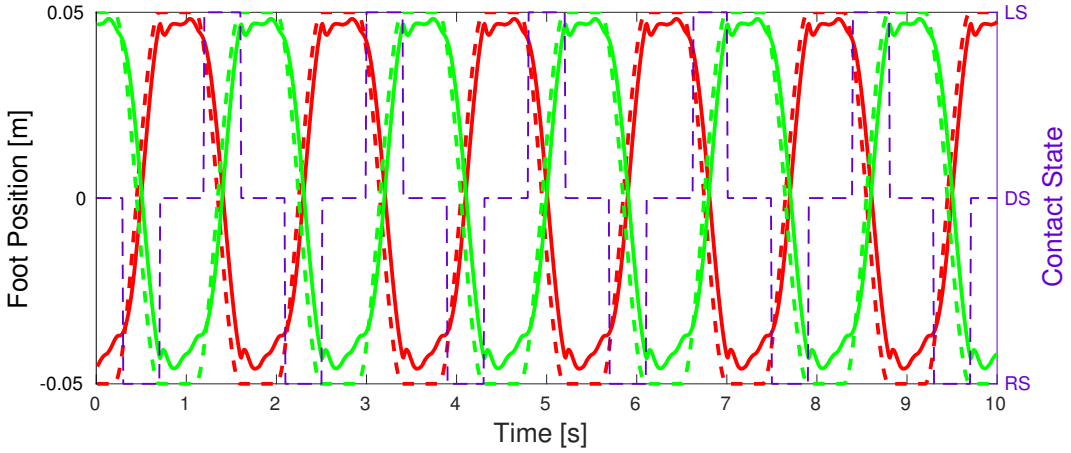
The simulation data is shown in Fig. 5-12, where the blue solid line is the measured data collected from the simulated WALK-MAN robot and the red dash-dot line is the desired one. The CoM and ZMP trajectories along X direction were tracked quite well while the ones along Y direction were a bit worse. It is because foot width is smaller compared with its length resulting less ankle torque available to keep stable and maintain precise tracking in y direction. Notable result in this simulation data is the CoM tracking along Z direction, where it was not tracked strictly, instead, it oscillated under the desired height just as expected. Besides, during walking, CoM height sometimes was higher than the one when the robot was standing straight just as shown in Fig. 5-12(c), because the robot needed to lift its swing leg for walking and the maximum CoM reachable height would increase. The CoM heights at the beginning and the end of walking were different, because the robot postures at the two moments were not exactly the same and corresponding maximum CoM reachable heights were different. Fig. 5-13 shows the CoM trajectory in the front Y-Z plane

and it is quite similar with the typical butterfly shape observed in human walking [44].

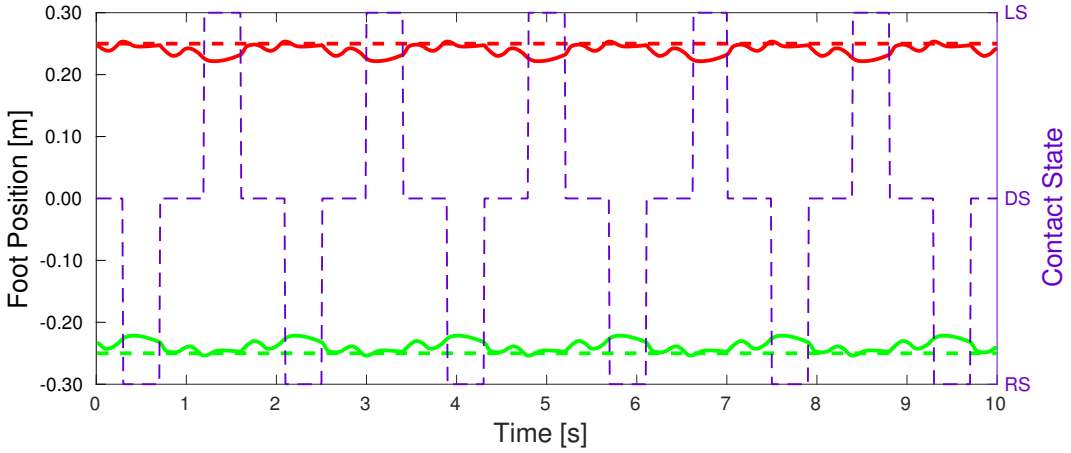
To compare the proposed control strategy with the normal LIPM one with low CoM height, the robot was simulated again with a constant CoM height 1.1 m. The robot can track the planned CoM trajectory very well but needs much bigger torque in knee joints. Fig. 5-14 shows the comparison result. Blue lines are collected from the control strategy while the red ones are for the walking with constant CoM height 1.1 m which is typical for WALK-MAN robot generating LIPM walking pattern. And solid and dash-dot lines represent left and right knees separately. By utilizing the proposed control strategy, the knee angles are quite close to zero and the corresponding joint torques are almost half of the ones with constant low CoM height. High energy efficiency is promised for the proposed control strategy. This simulation result supported the raised hypothesis well and proved the feasibility of the methods.

5.4 Summary

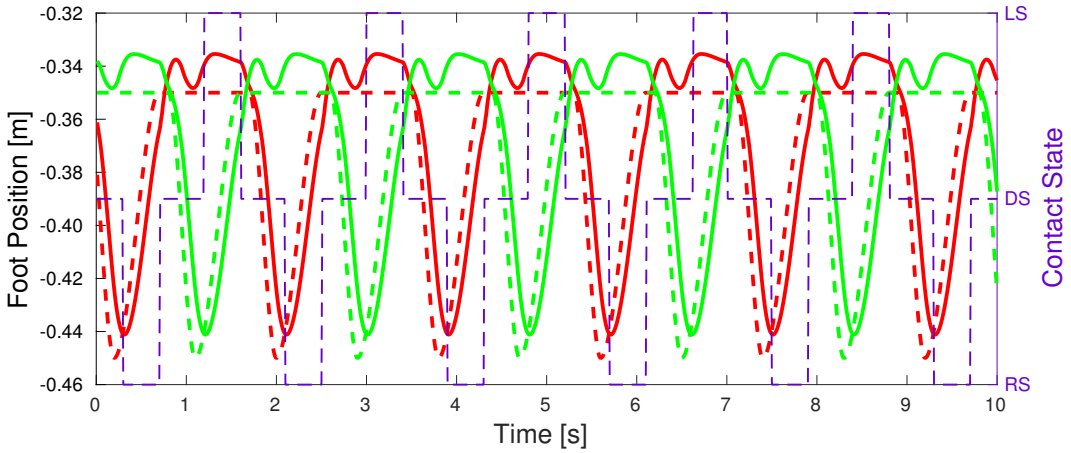
In this chapter, with respect to different types of joint actuators, various control algorithms were proposed to achieve accurate and stable tracking of the planned trajectories. For the robots with position controlled joints, CoM stabilizer is developed to consider the overall tracking error of all the joints systematically, so that the walking stability can be improved. To achieve compliant behaviors and reduce the ground impact during landing, robots with torque controlled joints are preferred and corresponding control strategies are chosen. The quasi-static Cartesian impedance control is first explored to allow the robots to interact with the environment compliantly without knowing the exact contact state. Inverted pendulum walking pattern was realized successfully on the upper body of Centauro robot with the impedance control. Thereafter, the optimization-based full body torque control framework was implemented and exhibited amazing performance when dealing with multiple targets simultaneously. Very human-like behaviors, like balancing with arms, can be generated automatically based on the full body dynamics.



(a) Foot trajectories in X direction.

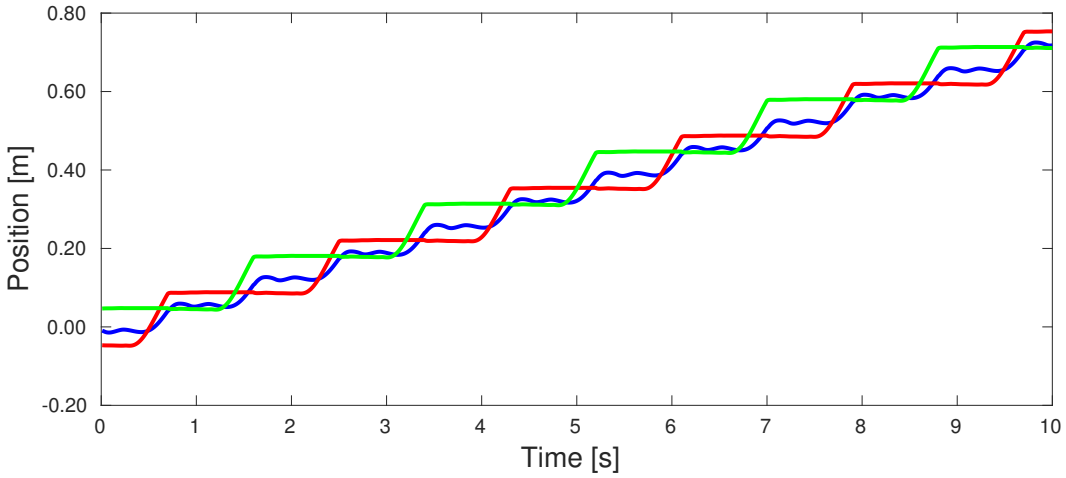


(b) Foot trajectories in Y direction.

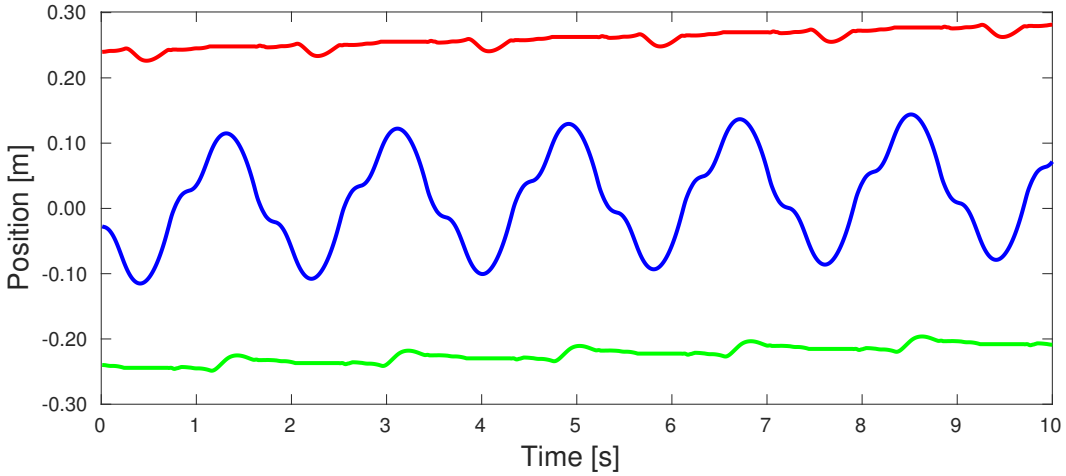


(c) Foot trajectories in Z direction.

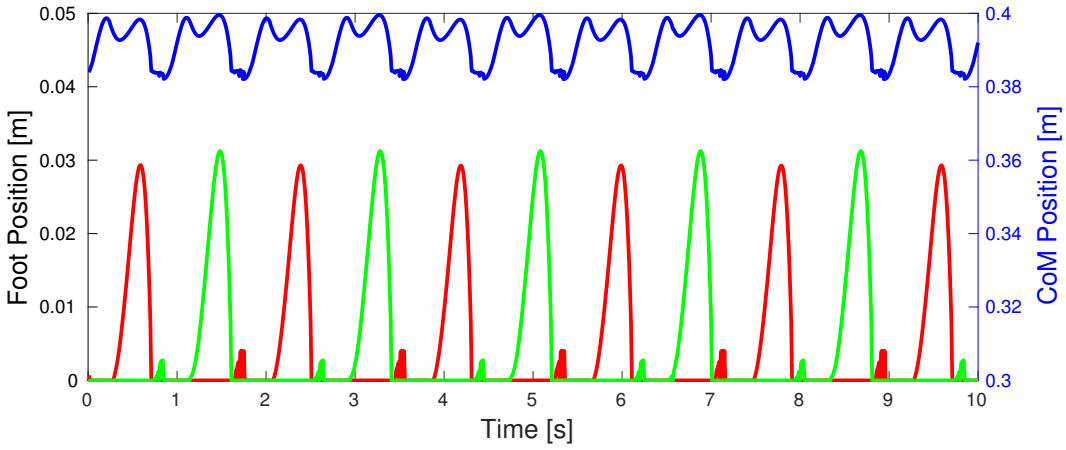
Figure 5-6: Foot trajectories of CENTAURO in the local frame of pelvis (red: left foot, green: right foot, purple: contact state, solid: measured, dash-dot: desired).



(a) CoM and foot trajectories in X direction.

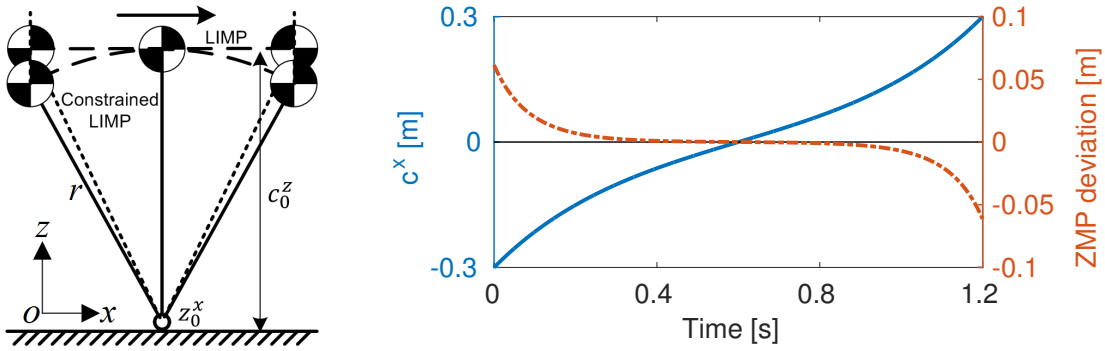


(b) CoM and foot trajectories in Y direction.



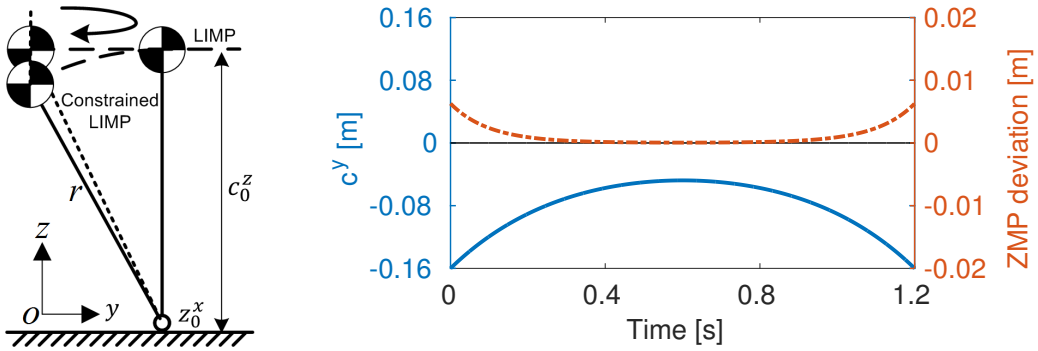
(c) CoM and foot trajectories in Z direction.

Figure 5-7: CoM and foot trajectories of CENTAURO upper body in the world frame (blue: CoM, red: left foot, green: right foot).



(a) Model (b) Trajectory and ZMP deviation

Figure 5-9: Simplified model and ZMP deviation in sagittal plane.



(a) Model (b) Trajectory and ZMP deviation

Figure 5-10: Simplified model and ZMP deviation in lateral plane.

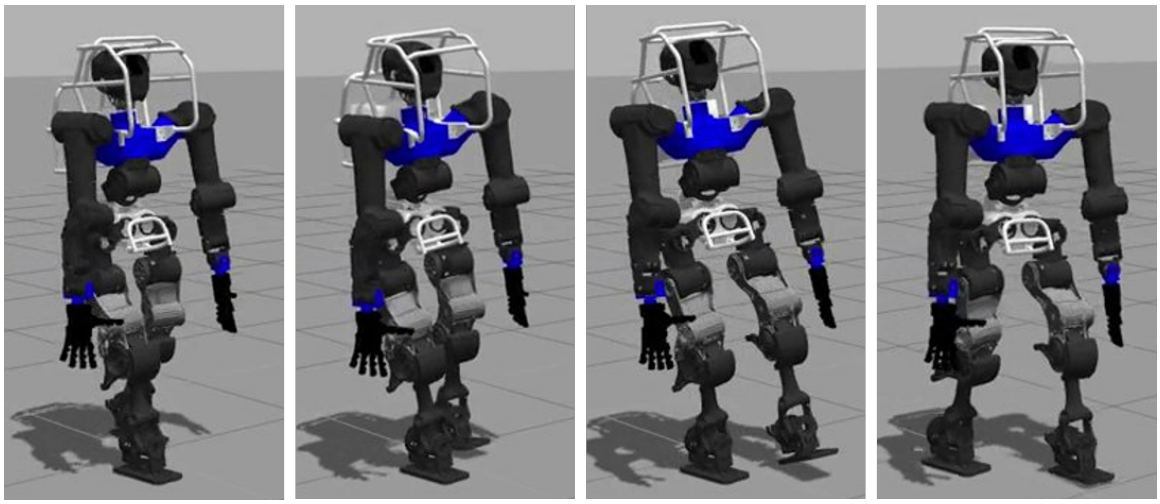
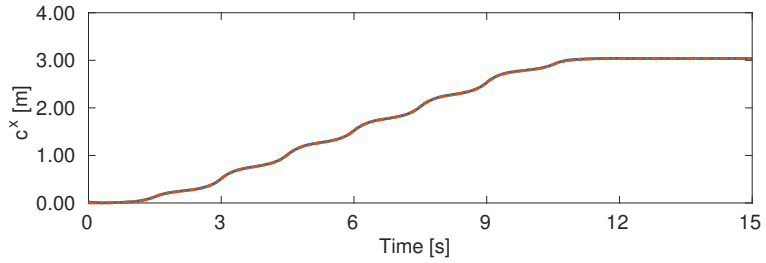
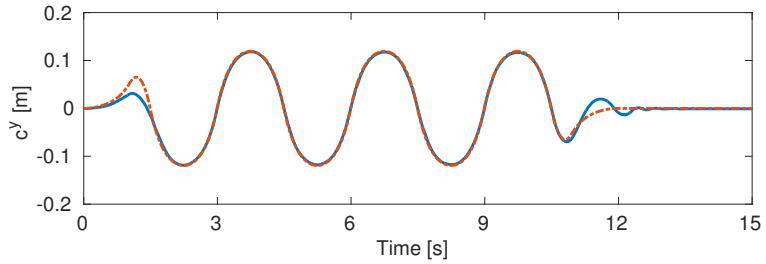


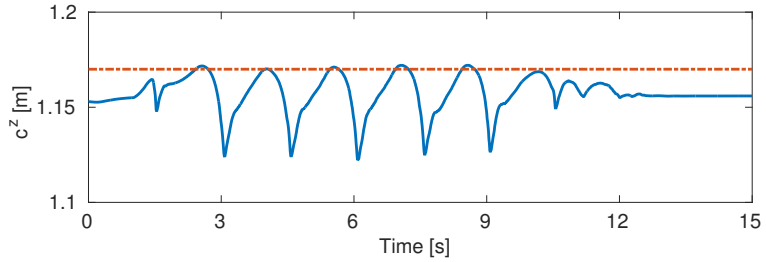
Figure 5-11: WALK-MAN walking with straight leg (interval: 0.5 second).



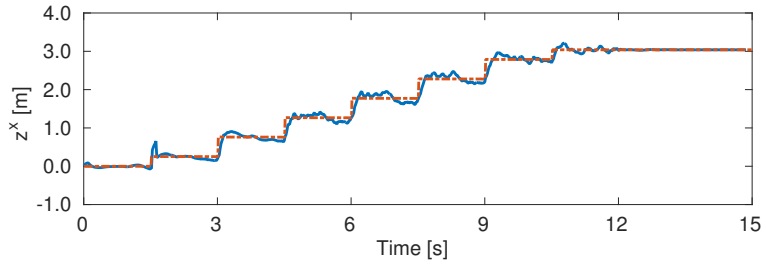
(a) CoM trajectory in x direction.



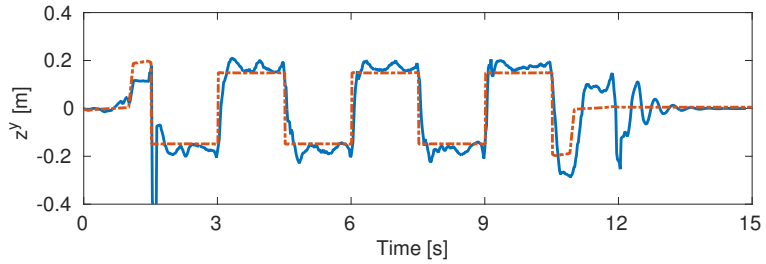
(b) CoM trajectory in y direction.



(c) CoM trajectory in z direction.



(d) ZMP trajectory in x direction.



(e) ZMP trajectory in y direction.

Figure 5-12: CoM and ZMP trajectories when walking straight (blue solid line is measured and red dash-dot line is desired).

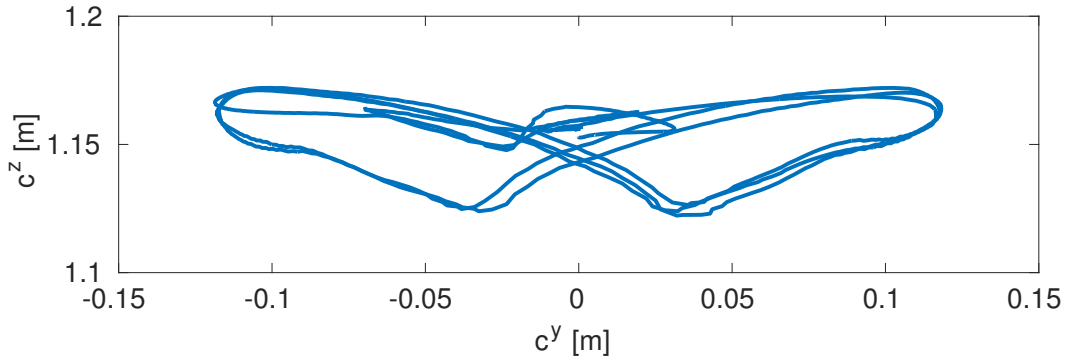
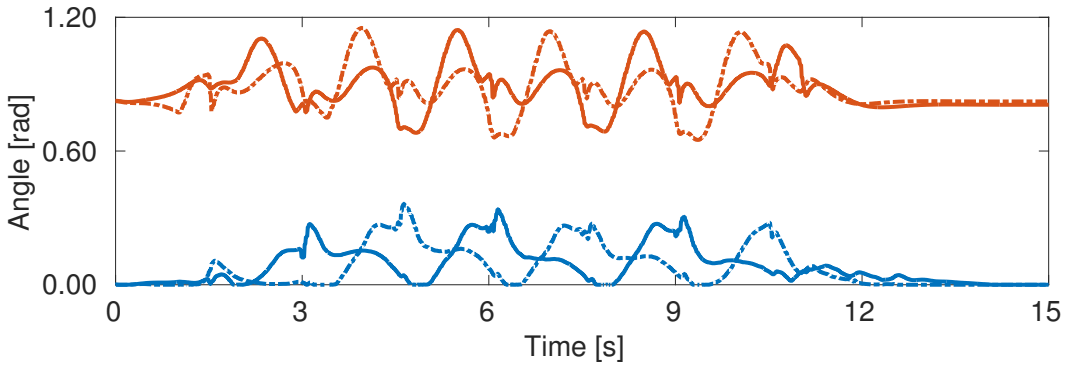
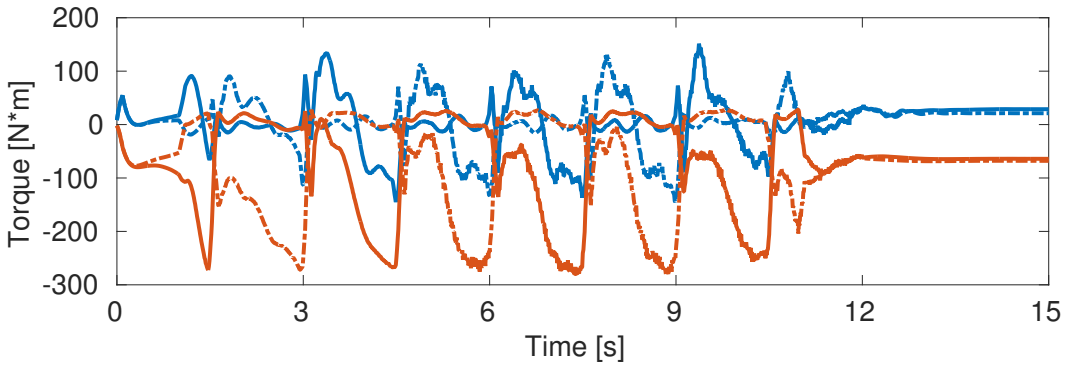


Figure 5-13: CoM trajectory projected in front y-z plane.



(a) Knee trajectory.



(b) Knee torque.

Figure 5-14: Knee position and torque data (blue solid and dash-dot lines are for left and right straight legs,red solid and dash-dot lines are for left and right bent legs).

Chapter 6

Conclusion and Future Work

On the way to reproduce the stable and versatile mobility of human on robots, this thesis contributed on the aspect of motion planning and control in humanoid locomotion. Robust walking is achieved to cross uneven terrain and resist external disturbance. Good balancing capability is demonstrated that the robot is able to make full use of whole body to keep balancing. Nevertheless, with regard to our expectation on humanoid robots, there are still some advanced features missing in the proposed control framework.

6.1 Full Body Motion Planning

As seen in Chapter 2-4, the motion planning targets only on the locomotion, and extra consideration is needed if other motion tasks, such as reaching an apple by arms or rotating the waist to avoid obstacles, are involved. It is much more natural and convenient to construct a unified framework for planning to take all the bodies of robot into account so that not only the locomotion tasks but also manipulation tasks or any other task can be considered at the same time.

There are already some researches working toward this target. The full body motion planning is usually constructed as a big nonlinear optimization problem which is general enough to describe various tasks performed by humanoid robots. However, although all kinds of nonlinear optimization technologies are implemented, how to

compute fast enough to run in real time is still an open question. For a task which looks not so complicated for human, it would take the robots minutes, or even hours to solve, which is definitely not the case of human solving the problem.

To address the computation-intensive problem in nonlinear optimization, Clever *et al.* tried to combine the optimal control and learning of movement primitives to accelerate the optimization process [4, 5]. This should be a promising direction for complicated real-time motion planning problems which involves multiple bodies of the robot. Based on the experiences learned from past experiments, the robot is expected to use less time and computation power to resolve problems similar to the ones encountered before.

6.2 Stable Torque Control for Motion Tracking

As introduced in Chapter 5, the optimization-based full body torque control is amazing when dealing with multiple tasks and able to generate very human-like natural behaviors automatically. However, there are still some issues lying behind to drive the robot out of control now and then.

Some attempts were carried out to realize energy-efficient straight leg walking in this thesis. The first problem encountered is the singularity issue resulting unstable numerical computation during optimization when the leg is almost completely straight and drives the leg Jacobian matrix ill-conditioned. One widely-used way to mitigate this problem, which is also what I took here, is to add a damping term and consider joint limits during the optimization [3, 32]. Most of the time, this approach works well while the optimization solver would still give unreasonable results sometimes. Deeper insight could be casted into the optimization theory to give a better solution on this.

Besides the singular issue, another defect in the existing torque controller is its short-sight. The optimization tries to minimize the objective functions only considering the current time slot which may not be proper in the long run. One straightforward way to eliminate the drawback is to construct a bigger optimization problem by

taking more time slots into account. The same trouble encountered in the full body motion planning shows up that according to existing optimization solvers, very long calculation time is inevitable due to the limited computation power and therefore impractical for real time implementing. Some efforts could be done in the future to solve this problem along with the one emerged in the full body motion planning.

6.3 Conclusion

Based on the material above, the conclusion can be drawn here. This thesis explored two most important aspects of humanoid locomotion, motion planning and control (motion tracking and stabilization).

Motion Planning

For the motion planning, various methods were proposed in different levels of model dependence. First, a model free approach was proposed which utilizes linear regression to estimate the relationship between foot placement and moving velocity. Relying on the estimated relationship, appropriate foot placement can be predicted according to the current and desire velocities, and therefore the motion of all the bodies can be figured out afterwards. The philosophy behind this method can be considered as a unified solution to realize diverse gaits of legged robots, such as the one-legged hopping, bipedal walking with point feet or normal soles as demonstrated. In addition, the robustness and adaptivity to external disturbances and modeling error is also observed within this method.

A disadvantage of model-free methods is the poor generalization ability compared to that of model-based methods. If the robot is changed, new experimental data should be collected to train or update the controller, which may be too risky for real robots. To address this issue, model-based planning technologies are exploited. The well-known simplified model, linear inverted pendulum model, is utilized to form a quadratic programming problem following the model predictive control paradigm. CoM trajectory can be optimized given predefined foot placements or together with

foot placements at the same time with respect to the ZMP constraint. The elaborately designed re-planning algorithm with sparse discretization based on the simplified model can be fast enough to run in real time and robust enough to resist external pushes and cross uneven terrain.

Inspired by the sensorimotor control realm, nonlinear models are proposed to perform forward simulation iteratively following the multiple shooting method.Good candidates of nonlinear models are studied for sagittal and lateral planes to achieve minimum computation without downgrading much of the accuracy. And the desired foot placement can be calculated after the numerical computation. Since a manually designed walking pattern is predefined to fix most of the degrees of the robot, the decision variables left for the nonlinear optimization are much less and could be solved in milliseconds which is sufficient to run in real time. A simulated COMAN sized robot with the nonlinear model controller insider is shown to be capable of blindly traveling across a stair and withstand external push or continuously ball impact attacks without falling over.

With the three proposed planning methods, most of the locomotion tasks of humanoid robot can be completed. However, when coming down to motion tasks involving all the bodies of robot and considering all the kinematic and dynamic constraints simultaneously, the existing frameworks could not do too much on this. The whole problem turns to a big nonlinear optimization problem, which is not solvable nowadays under the real-time requirement. Combination with machine learning to speed up the computation based on experience data is a promising direction for the future research on complicated motion planning.

Motion Tracking and Stabilization

Regarding to the motion tracking and stabilization, to improve the walking stability, CoM stabilizer is developed for the position controlled robots as a tool of considering the tracking error of all the joints systematically. Whereas, the big landing impact induced by the stiff position controlled joints is still inevitable and will make the robot fall down easily if the ground modeling is not correct. The compliant behav-

ior and precise tracking brought by the torque controlled robot is very appealing for robotics researchers, and torque controlled robots are getting more and more popular despite of its much higher requirements on the joint actuators. The corresponding optimization-based full body torque control framework was implemented and exhibited amazing performance when dealing with multiple targets at the same time. It is able to generate very human-like behaviors automatically based on the full body dynamics like balancing with arms. Sometimes, without knowing the exact contact state of feet, quasi-static Cartesian impedance control can be a good alternative of the optimization-based full-body torque control for torque-control robots to interact with the environment. Hence, a short introduction of walking like a inverted pendulum model is given based on the quasi-static Cartesian impedance control.

As one of the most popular frameworks for low level control, optimization-based full body torque control framework works very well most of the time. Whereas, there are still some moments when the controller can go wrong. Deep insight into the optimization theory and longer horizon for the optimization instead of only considering the current time slot is expected to be investigated in the future.

Appendix A

WALK-MAN Robot

The WALK-MAN robot simulated in the Gazebo simulation platform is consistent with the real robot in the lab [60]. It contains 31 DoFs with height around 1.9 m and total mass around 130 Kg as shown in Fig. A-1. It has two 6 DoFs legs and two 7 DoFs arms, and another 3 DoFs are for the waist and 2 more DoFs for the neck joints. For the lower body of WALK-MAN used in this thesis, only the two 6 DoFs legs and the pelvis base are included, and an extra 20 kg load is mounted on top of the pelvis base. In the simulation, each joint is torque controlled with combined feed-forward and feedback terms:

$$\boldsymbol{\tau} = \boldsymbol{\tau}_{\text{ref}} + K_p(\mathbf{q}_{\text{ref}} - \mathbf{q}) + K_d(\dot{\mathbf{q}}_{\text{ref}} - \dot{\mathbf{q}}) + K_i \int (\mathbf{q}_{\text{ref}} - \mathbf{q}) \tag{A.1}$$

Where τ_{ref} is the feed-forward joint torque, $\mathbf{q}_{\text{ref}}$ and $\dot{\mathbf{q}}_{\text{ref}}$ are the joint position and velocity references. K_p, K_d and K_i are PID gains for the feedback term.

For the optimization-based full-body torque control, joint accelerations are figured out first as the decision variables. And then the joint positions and velocities can be integrated from the optimized joint accelerations and the current joint positions and velocities. The integration item of PID feedback was not used here, therefore K_i was set zero. Feed-forward torque dominates the control command while feedback gains are so small that the robot can not even stand up without feed-forward torques. The control frequency is set as 1 kHz for the torque control.

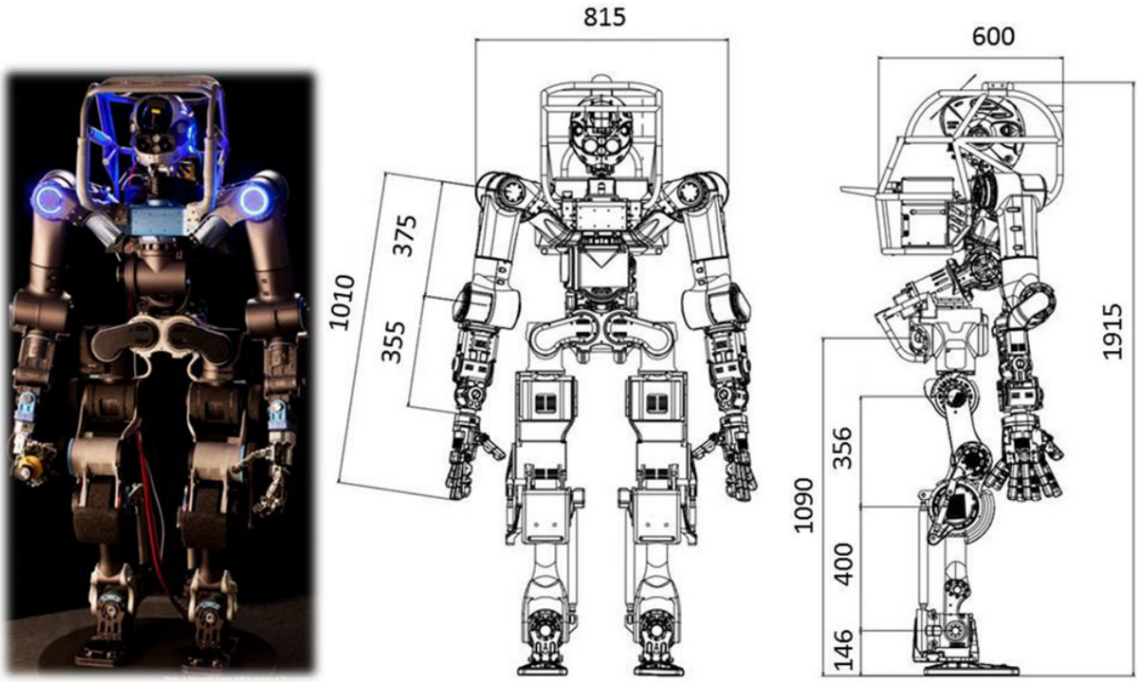


Figure A-1: WALK-MAN body size specifications (all dimensions are in mm)

To be noted, in simulation, the joint command without feedback terms or pure torque control still works, and usually is better than the one with feedback terms, because directly adding the feedback torques to the command kind of ruin the optimization result. However, considering the noise and imperfect torque tracking of the joint actuators, it is usually more stable to utilize the high bandwidth of position feedback control.

Appendix B

Publication

During the period of my Ph.D. study, I published six conference papers (4 first author, 2 second author) as listed below:

1. Yangwei You, Chengxu Zhou, Zhibin Li, Nikos Tsagarakis. A study of nonlinear forward models for dynamic walking. In International Conference on Robotics and Automation (ICRA), 2017.

2. Songyan Xin, Yangwei You, Chengxu Zhou, and Nikos Tsagarakis. Humanoid Running Based on Centroidal Dynamics and Heuristic Foot Placement. In International Conference on Robotics and Biomimetics (ROBIO), 2017.

3. Songyan Xin, Yangwei You, Chengxu Zhou, Cheng Fang, and Nikos Tsagarakis. A torque-controlled humanoid robot riding on a two-wheeled mobile platform. In International Conference on Intelligent Robots and Systems (IROS), 2017.

4. Yangwei You, Songyan Xin, Chengxu Zhou, and Nikos Tsagarakis. Straight leg walking strategy for torque-controlled humanoid robots. In International Conference on Robotics and Biomimetics (ROBIO), 2016.

5. Yangwei You, Zhibin Li, Darwin Caldwell, and Nikos Tsagarakis. From one-legged hopping to bipedal running and walking: A unified foot placement control based on regression analysis. In International Conference on Intelligent Robots and Systems (IROS), 2015.

6. Yangwei You, Zhibin Li, Nikos Tsagarakis, and Darwin Caldwell. Foot placement control for bipedal walking on uneven terrain: An online linear regression analysis approach. In International Conference on Climbing and Walking Robots and Support Technologies for Mobile Machines (CLAWAR), 2015.

Bibliography

[1] Mojtaba Ahmadi and Martin Buehler. Stable control of a simulated one-legged running robot with hip and leg compliance. *Transactions on Robotics and Automation*, 13(1):96–104, 1997.

[2] Justin Carpentier, Steve Tonneau, Maximilien Naveau, Olivier Stasse, and Nicolas Mansard. A versatile and efficient pattern generator for generalized legged locomotion. In *Robotics and Automation (ICRA), 2016 IEEE International Conference on*, pages 3555–3561. IEEE, 2016.

[3] Stefano Chiaverini, Bruno Siciliano, and Olav Egeland. Review of the damped least-squares inverse kinematics with experiments on an industrial robot manipulator. *IEEE Transactions on control systems technology*, 2(2):123–134, 1994.

[4] Debora Clever, Monika Harant, Henning Koch, Katja Mombaur, and Dominik Endres. A novel approach for the generation of complex humanoid walking sequences based on a combination of optimal control and learning of movement primitives. *Robotics and Autonomous Systems*, 83:287–298, 2016.

[5] Debora Clever, Monika Harant, Katja Mombaur, Maximilien Naveau, Olivier Stasse, and Dominik Endres. Cocomopl: A novel approach for humanoid walking generation combining optimal control, movement primitives and learning and its transfer to the real robot hrp-2. *IEEE Robotics and Automation Letters*, 2(2):977–984, 2017.

[6] Hongkai Dai, Andrés Valenzuela, and Russ Tedrake. Whole-body motion planning with centroidal dynamics and full kinematics. In *Humanoid Robots (Humanoids), 2014 14th IEEE-RAS International Conference on*, pages 295–302. IEEE, 2014.

[7] Martin de Lasa, Igor Mordatch, and Aaron Hertzmann. Feature-based locomotion controllers. In *ACM Transactions on Graphics (TOG)*, page 131, 2010.

[8] Johannes Englsberger, Christian Ott, and Alin Albu-Schäffer. Three-dimensional bipedal walking control based on divergent component of motion. *IEEE Transactions on Robotics*, 31(2):355–368, 2015.

[9] Johannes Englsberger, Christian Ott, and Alin Albu-Schäffer. Three-dimensional bipedal walking control based on divergent component of motion. *Transactions on Robotics*, 31(2):355–368, 2015.

[10] Adrien Escande, Nicolas Mansard, and Pierre-Brice Wieber. Hierarchical quadratic programming: Fast online humanoid-robot motion generation. *The International Journal of Robotics Research*, 2014.

[11] Salman Faraji, Soha Pouya, Christopher G Atkeson, and Auke Jan Ijspeert. Versatile and robust 3d walking with a simulated humanoid robot (atlas) a model predictive control approach. *International Conference on Robotics and Automation*, 2014.

[12] Siyuan Feng. *Online Hierarchical Optimization for Humanoid Control*. PhD thesis, 2016.

[13] Siyuan Feng, E. Whitman, X. Xinjilefu, and C.G. Atkeson. Optimization based full body control for the atlas robot. In *International Conference on Humanoid Robots*, pages 120–127, 2014.

[14] Siyuan Feng, X Xinjilefu, Weiwei Huang, and Christopher G Atkeson. 3d walking based on online optimization. In *International Conference on Humanoid Robots*, pages 21–27, 2013.

[15] Toshio Fukuda, Youichirou Komata, and Takemasa Arakawa. Stabilization control of biped locomotion robot based learning with gas having self-adaptive mutation and recurrent neural networks. In *Robotics and Automation, 1997. Proceedings., 1997 IEEE International Conference on*, volume 1, pages 217–222. IEEE, 1997.

[16] Hartmut Geyer, Andre Seyfarth, and Reinhard Blickhan. Compliant leg behaviour explains basic dynamics of walking and running. *Proceedings of the Royal Society B: Biological Sciences*, 273(1603):2861–2867, 2006.

[17] Ioannis Havoutis, Claudio Semini, and Darwin G Caldwell. Virtual model control for quadrupedal trunk stabilization. *Dynamic Walking*, 2013.

[18] Andrei Herdt, Holger Diedam, Pierre-Brice Wieber, Dimitar Dimitrov, Katja Mombaur, and Moritz Diehl. Online walking motion generation with automatic footstep placement. *Advanced Robotics*, pages 719–737, 2010.

[19] Alexander Herzog, Ludovic Righetti, Felix Grimminger, Peter Pastor, and Stefan Schaal. Balancing experiments on a torque-controlled humanoid with hierarchical inverse dynamics. In *International Conference on Intelligent Robots and Systems*, pages 981–988, 2014.

[20] Christian Hubicki, Jesse Grimes, Mikhail Jones, Daniel Renjewski, Alexander Spröwitz, Andy Abate, and Jonathan Hurst. Atrias: Design and validation of a tether-free 3d-capable spring-mass bipedal robot. *The International Journal of Robotics Research*, 35(12):1497–1521, 2016.

[21] Marco Hutter, C David Remy, Mark A Hoepflinger, and Roland Siegwart. Scarleth: Design and control of a planar running robot. In *International Conference on Intelligent Robots and Systems*, pages 562–567, 2011.

[22] Marco Hutter, Hannes Sommer, Christian Gehring, Mark Hoepflinger, Michael Bloesch, and Roland Siegwart. Quadrupedal locomotion using hierarchical operational space control. *The International Journal of Robotics Research*, 2014.

[23] Matthew Johnson, Brandon Shrewsbury, Sylvain Bertrand, Duncan Calvert, Tingfan Wu, Daniel Duran, Douglas Stephen, Nathan Mertins, John Carff, William Rifenburgh, et al. Team ihmc's lessons learned from the darpa robotics challenge: Finding data in the rubble. *Journal of Field Robotics*, 34(2):241–261, 2017.

[24] Shuuji Kajita, Hirohisa Hirukawa, Kensuke Harada, and Kazuhito Yokoi. *Introduction to humanoid robotics*, volume 101. Springer, 2014.

[25] Shuuji Kajita, Fumio Kanehiro, Kenji Kaneko, Kiyoshi Fujiwara, Kensuke Harada, Kazuhito Yokoi, and Hirohisa Hirukawa. Biped walking pattern generation by using preview control of zero-moment point. In *International Conference on Robotics and Automation*, pages 1620–1626, 2003.

[26] Shuuji Kajita, Fumio Kanehiro, Kenji Kaneko, Kiyoshi Fujiwara, Kazuhito Yokoi, and Hirohisa Hirukawa. A realtime pattern generator for biped walking. In *International Conference on Robotics and Automation*, volume 1, pages 31–37, 2002.

[27] Shuuji Kajita, Kenji Kaneko, Fumio Kaneiro, Kensuke Harada, Mitsuharu Morisawa, Shinchiro Nakaoka, Kanako Miura, Kiyoshi Fujiwara, Ee Sian Neo, Isao Hara, et al. Cybernetic human hrp-4c: A humanoid robot with human-like proportions. In *Robotics Research*, pages 301–314. Springer, 2011.

[28] Shuuji Kajita, Mitsuharu Morisawa, Kanako Miura, Shin'ichiro Nakaoka, Kensuke Harada, Kenji Kaneko, Fumio Kanehiro, and Kazuhito Yokoi. Biped walking stabilization based on linear inverted pendulum tracking. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 4489–4496. IEEE, 2010.

[29] Kenji Kaneko, Kensuke Harada, Fumio Kanehiro, Go Miyamori, and Kazuhiko Akachi. Humanoid robot hrp-3. In *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, pages 2471–2478. IEEE, 2008.

[30] Sangbae Kim, Patrick M Wensing, et al. Design of dynamic legged robots. *Foundations and Trends® in Robotics*, 5(2):117–190, 2017.

[31] Twan Koolen, Sylvain Bertrand, Gray Thomas, Tomas de Boer, Tingfan Wu, Jesper Smith, Johannes Englsberger, and Jerry Pratt. Design of a momentum-based control framework and application to the humanoid robot atlas. *International Journal of Humanoid Robotics*, 13, 2016.

[32] Przemyslaw Kryczka, Kenji Hashimoto, Hideki Kondo, Aiman Omer, Hun-ok Lim, and Atsuo Takanishi. Stretched knee walking with novel inverse kinematics for humanoid robots. In *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, pages 3221–3226. IEEE, 2011.

[33] Scott Kuindersma, Robin Deits, Maurice Fallon, Andrés Valenzuela, Hongkai Dai, Frank Permenter, Twan Koolen, Pat Marion, and Russ Tedrake. Optimization-based locomotion planning, estimation, and control design for the atlas humanoid robot. *Autonomous Robots*, 40(3):429–455, 2016.

[34] Zhibin Li, Nikos G Tsagarakis, and Darwin G Caldwell. Walking trajectory generation for humanoid robots with compliant joints: Experimentation with coman humanoid. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 836–841. IEEE, 2012.

[35] Zhibin Li, Nikos G Tsagarakis, and Darwin G Caldwell. Walking pattern generation for a humanoid robot with compliant joints. *Autonomous Robots*, 35(1):1–14, 2013.

[36] Zhibin Li, Nikos G Tsagarikis, Darwin G Caldwell, and Bram Vanderborght. Trajectory generation of straightened knee walking for humanoid robot icub. In *International Conference on Control Automation Robotics and Vision*, pages 2355–2360, 2010.

[37] Duane W Marhefka, David E Orin, James P Schmiedeler, and Kenneth J Waldron. Intelligent control of quadruped gallops. *Transactions on Mechatronics*, 8(4):446–456, 2003.

[38] Tad McGeer. Passive dynamic walking. *the international journal of robotics research*, 9(2):62–82, 1990.

[39] Biren Mehta and Stefan Schaal. Forward models in visuomotor control. *Journal of Neurophysiology*, 88(2):942–953, 2002.

[40] R Christopher Miall and Daniel M Wolpert. Forward models for physiological motor control. *Neural networks*, 9(8):1265–1279, 1996.

[41] Matteo Mischiati, Huai-Ti Lin, Paul Herold, Elliot Imler, Robert Olberg, and Anthony Leonardo. Internal models direct dragonfly interception steering. *Nature*, 517(7534):333–338, 2015.

[42] Mitsuharu Morisawa, Shuuji Kajita, Kenji Kaneko, Kensuke Harada, Fumio Kanehiro, Kiyoshi Fujiwara, and Hirohisa Hirukawa. Pattern generation of biped walking constrained on parametric surface. In *International Conference on Robotics and Automation*, pages 2405–2410, 2005.

[43] Yu Ogura, Kazushi Shimomura, Hideki Kondo, Akitoshi Morishima, Tatsu Okubo, Shimpei Momoki, Hun-ok Lim, and Atsuo Takanishi. Human-like walking with knee stretched, heel-contact and toe-off motion by a humanoid robot.

In *International Conference on Intelligent Robots and Systems*, pages 3976–3981, 2006.

[44] Michael S Orendurff, Ava D Segal, Glenn K Klute, Jocelyn S Berge, et al. The effect of walking speed on center of mass displacement. *Journal of rehabilitation research and development*, 41(6A):829, 2004.

[45] David E Orin, Ambarish Goswami, and Sung-Hee Lee. Centroidal dynamics of a humanoid robot. *Autonomous Robots*, 35(2-3):161–176, 2013.

[46] Nicholas Paine, Joshua S Mehling, James Holley, Nicolaus A Radford, Gwendolyn Johnson, Chien-Liang Fok, and Luis Sentis. Actuator control for the nasa-jsc valkyrie humanoid robot: A decoupled dynamics approach for torque control of series elastic robots. *Journal of Field Robotics*, 2015.

[47] 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.

[48] Jerry Pratt, John Carff, Sergey Drakunov, and Ambarish Goswami. Capture point: A step toward humanoid push recovery. In *International Conference on Humanoid Robots*, pages 200–207, 2006.

[49] Jerry Pratt and Gill Pratt. Intuitive control of a planar bipedal walking robot. In *International Conference on Robotics and Automation*, volume 3, pages 2014–2021, 1998.

[50] Marc Raibert, Kevin Blankespoor, Gabriel Nelson, Rob Playter, et al. Bigdog, the rough-terrain quadruped robot. In *Proceedings of the 17th World Congress*, volume 17, pages 10822–10825, 2008.

[51] Marc H Raibert et al. *Legged robots that balance*, volume 3. MIT press Cambridge, MA, 1986.

[52] Marc H Raibert and Frank C Wimberly. Tabular control of balance in a dynamic legged system. *Transactions on Systems, Man and Cybernetics*, (2):334–339, 1984.

[53] Yoshiaki Sakagami, Ryujin Watanabe, Chiaki Aoyama, Shinichi Matsunaga, Nobuo Higaki, and Kikuo Fujimura. The intelligent asimo: System overview and integration. In *Intelligent Robots and Systems, 2002. IEEE/RSJ International Conference on*, volume 3, pages 2478–2483. IEEE, 2002.

[54] Luis Sentis. *Synthesis and control of whole-body behaviors in humanoid systems*. PhD thesis, Citeseer, 2007.

[55] Luis Sentis and Oussama Khatib. A whole-body control framework for humanoids operating in human environments. In *International Conference on Robotics and Automation*, pages 2641–2648, 2006.

[56] Syamimi Shamsuddin, Luthffi Idzhar Ismail, Hanafiah Yussof, Nur Ismarrubie Zahari, Saiful Bahari, Hafizan Hashim, and Ahmed Jaffar. Humanoid robot nao: Review of control and motion exploration. In *Control System, Computing and Engineering (ICCSCE), 2011 IEEE International Conference on*, pages 511–516. IEEE, 2011.

[57] Tomomichi Sugihara. Standing stabilizability and stepping maneuver in planar bipedalism based on the best com-zmp regulator. In *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*, pages 1966–1971. IEEE, 2009.

[58] Russell L Tedrake. *Applied optimal control for dynamically stable legged locomotion*. PhD thesis, Massachusetts Institute of Technology, 2004.

[59] Koji Terada and Yasuo Kuniyoshi. Online gait planning with dynamical 3d-symmetrization method. In *International Conference on Humanoid Robots*, pages 222–227, 2007.

[60] NG Tsagarakis, DG Caldwell, A Bicchi, F Negrello, M Garabini, W Choi, L Baccelliere, V Loc, J Noorden, M Catalano, et al. Walk-man: A high performance humanoid platform for realistic environments. *Journal of Field Robotics (JFR)*, 2016.

[61] Pieter Van Zutven, Dragan Kostic, and Henk Nijmeijer. Foot placement for planar bipeds with point feet. In *International Conference on Robotics and Automation*, pages 983–988, 2012.

[62] P. M. Wensing and D. E. Orin. Generation of dynamic humanoid behaviors through task-space control with conic optimization. In *International Conference on Robotics and Automation*, pages 3103–3109, 2013.

[63] Derek L Wight, Eric G Kubica, and David W Wang. Introduction of the foot placement estimator: A dynamic measure of balance for bipedal robotics. *Journal of computational and nonlinear dynamics*, 3(1), 2008.

[64] Daniel M Wolpert and Mitsuo Kawato. Multiple paired forward and inverse models for motor control. *Neural networks*, 11(7):1317–1329, 1998.

[65] Yangwei You, Zhibin Li, Darwin G Caldwell, and Nikos G Tsagarakis. From one-legged hopping to bipedal running and walking: A unified foot placement control based on regression analysis. In *International Conference on Intelligent Robots and Systems*, pages 4492–4497, 2015.

[66] Yangwei You, Zhibin Li, Nikos G Tsagarakis, and Darwin G Caldwell. Foot placement control for bipedal walking on uneven terrain: An online linear regression analysis approach. In *International Conference on Climbing and Walking Robots and Support Technologies for Mobile Machines*, page 478, 2015.