Online Virtual Repellent Point Adaptation for Biped Walking using Iterative Learning Control

handed in
MASTER'S THESIS

B.Sc. Shengzhi Wang

born on the 14.03.1995
living in:
Kohlstr. 7
80469 Munich
Tel.: 17657642589

Human-centered Assistive Robotics
Technical University of Munich

Univ.-Prof. Dr.-Ing. Dongheui Lee

Supervisor:	Prof. Dongheui Lee
Advisor:	George-Adrian Mesesan, Dr. Christian Ott
Start:	01.04.2020
Intermediate Report:	26.01.2021
Delivery:	09.02.2021

TECHNISCHE UNIVERSITÄT MÜNCHEN
Human-centered Assistive Robotics
UNIV.-PROF. DR.-ING. DONGHEUI LEE

January 31, 2021

MASTER'S THESIS
for
Shengzhi Wang
Student ID 03709138, Degree EI

Online Virtual Repellent Point Adaption for Biped Walking

Problem description:

Legged robot locomotion is a challenging problem due to its hybrid dynamics (discrete contact sequencing and continuous whole-body motion), and the constraints imposed on the contact forces. Recently, the related concepts of the three-dimensional Divergent Component of Motion (DCM) and the Virtual Repellent Point (VRP) were introduced in [1], decomposing the second-order center-of-mass (CoM) dynamics into two first-order dynamics, with the CoM converging to the DCM (stable dynamics), and the DCM diverging away from the VRP (unstable dynamics). Based on this formulation, continuous closed-form DCM and CoM trajectories can be generated using a piecewise interpolation of the VRP trajectory over a sequence of waypoints [2]. This highly compact motion representation is a natural way of handling the locomotion hybrid dynamics, with the discrete contact sequencing being mapped onto the VRP waypoints. This approach has been used successfully for bipedal locomotion in [1], and dynamic multi-contact motion in [3].
For a practical implementation on a real robot there are two important aspects to consider: First, during the swing leg motion, the whole-body dynamics requires a compensating torque to be applied at the point of contact. This corresponds to a shift of the center-of-pressure (CoP) within the foot from the nominal position (typically the middle of the foot). Second, model uncertainties lead to imperfect tracking of the reference trajectories, and require adjustments of the commanded VRP. Both aspects thereby limit the maximum walking speed and reduce the robustness with respect to external disturbances. A compensative Zero Moment Point (ZMP) trajectory addressing these two aspects in the context of the Linear Inverted Pendulum (LIP) model was proposed in [4].
This thesis will apply the iterative learning control approach similar to the one presented in [4] for the LIP to the three-dimensional DCM framework. The goal is to learn and generate online a VRP compensation trajectory which reduces the CoP motion within the foot and keeps the commanded VRP close to the center of the foot. The algorithm will be tested in simulation using a point-mass model and a whole-body simulation, and in experiments with the humanoid robot TORO.

Tasks:

- Literature research on 3-D DCM-based control, and online iterative learning approach for ZMP compensation
- Build an online learning framework for VRP compensation against unmodeled uncertainties
- Test the approach in simulation and on the real robot

Bibliography:

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

[2] George Mesesan, Johannes Englsberger, Christian Ott, and Alin Albu-Schäffer. Convex properties of center-of-mass trajectories for locomotion based on divergent component of motion. *IEEE Robotics and Automation Letters*, 3(4):3449–3456, 2018.

[3] George Mesesan, Johannes Englsberger, Bernd Henze, and Christian Ott. Dynamic multi-contact transitions for humanoid robots using divergent component of motion. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4108–4115. IEEE, 2017.

[4] Kai Hu, Christian Ott, and Dongheui Lee. Learning and generalization of compensative zero-moment point trajectory for biped walking. *IEEE Transactions on Robotics*, 32(3):717–725, 2016.

Supervisor:	Prof. Dongheui Lee
Start:	01.04.2020
Intermediate Report:	26.01.2021
Delivery:	09.02.2021

(D. Lee)
Univ.-Professor

With my signature below, I assert that the work in this thesis has been composed by myself independently and no source materials or aids other than those mentioned in the thesis have been used.

München, February 1, 2021

Place, Date

Signature

Abstract

The legged robot locomotion based on the concepts of the three dimensional Divergent Component of Motion (DCM) and the Virtual Repellent Point (VRP) is a state-of-the-art walking algorithm, which decomposes the second-order center of mass (CoM) dynamics into a first-order stable dynamics (with CoM converging to DCM) and a first-order unstable dynamics (with DCM diverging away from VRP). However, the swing leg motion and model inaccuracies cause imperfect tracking of the reference trajectories, limiting the maximum walking speed and affecting the walking robustness. In this thesis, an online learning framework is proposed to enhance the DCM-based walking algorithm's robustness. The framework learns a feedforward adjusted VRP reference trajectory from the VRP errors of the repetitive walking behaviors by using the concept of iterative learning control (ILC). This adjusted VRP reference trajectory is applied to the walking pattern generator to generate the adjusted DCM and CoM reference trajectory. Specifically, the framework is presented in two different implementations learning from different VRP error signals: measured and commanded error. For the measured VRP, the measurement-error-based implementation performs better reference trajectory tracking performance. Moreover, a pre-learned knowledge database, which consists of compensative VRP trajectories obtained from the walking trials with different parameter sets, is constructed to predict an initial pre-learned knowledge based on the k-nearest neighbor regression. This initial knowledge can further stabilize the initial two steps when the framework is first applied. The improved walking robustness is demonstrated from the reduced average VRP error, suppressed sliding behavior and increased achievable walking speed on the DLR humanoid robot TORO.

Contents

1 Introduction **5**
1.1 Problem Statement and Related Work . . . 5
1.2 Outline . . . 7

2 Technical Background **9**
2.1 Bipedal Walking Based on Divergent Component of Motion . . . 9
2.1.1 Fundamentals . . . 9
2.1.2 Reference Trajectory Generation . . . 10
2.1.3 Divergent Component of Motion Tracking Control . . . 14
2.1.4 Measurement of Virtual Repellent Point . . . 15
2.2 Iterative Learning Control . . . 16
2.2.1 General Algorithm . . . 16
2.2.2 Stability Analysis . . . 17

3 Technical Approach **21**
3.1 Design of Online Iterative Learning Control Framework . . . 21
3.1.1 Definition of Iteration . . . 21
3.1.2 Measurement-Error-Based Reference Trajectories Adaptation . 22
3.1.3 Commanded-Error-Based Reference Trajectories Adaptation . 27
3.1.4 Convergence Condition . . . 28
3.1.5 Failed Design . . . 29
3.2 Stability Proof . . . 31
3.2.1 Simple One Waypoint Case . . . 32
3.2.2 Simple Two Waypoints Case . . . 39
3.2.3 Generalized Case . . . 44
3.3 Convergence Analysis . . . 46

4 Evaluation **49**
4.1 Simulation Results . . . 49
4.1.1 General Framework . . . 50
4.1.2 Framework with Pre-Learning Procedure . . . 53
4.2 Experimental Results . . . 54
4.2.1 Measurement-Error-Based Reference Trajectories Adaptation . 54

4.2.2 Commanded-Error-Based Reference Trajectories Adaptation . 54
4.3 Result Summary 58

5 Discussion 63

6 Conclusion and Future Work 65

Appendix 66

A Parameter Setting in Simulation 67
A.1 Model Inaccuracies Modeling 67
A.2 Construction of Pre-Learned Knowledge Database 68

List of Figures 69

Bibliography 73

Chapter 1

Introduction

1.1 Problem Statement and Related Work

Humanoid robot locomotion is a complex problem due to the hybrid system dynamics and the system non-linearity. Based on the concept of zero-moment point (ZMP) [VS72], many early works were successful on the biped walking of legged robots [TLTK90]-[Tak89]. The key idea for controlling walking is to generate a stable walking pattern and subsequently track the planned trajectory precisely. The walking pattern can be planned offline based on an accurate robot model [Tak89]. To simplify the complexity of the bipedal walking problems, a simplified walking model, Linear Inverted Pendulum (LIP) model [SNI02], was commonly used to achieve an online walking control by focusing on the center of mass (CoM) dynamics. The LIP model defines that an inverted pendulum remains the same height during the movement in practice. However, the model inaccuracies between the actual robot dynamics model and the simplified model lead to an imperfect tracking of the reference trajectories. In the worst case, the imperfect tracking causes the robot to fall if the ZMP leaves the support polygon. The standard way to avoid falling is by adding a feedback stabilization controller into the system [KMM+10, CKOY07]. However, these balance methods do not fundamentally solve the effect caused by the model inaccuracies. A persistent ZMP error still exists.

Learning control algorithm is one of the solutions to improve the trajectory tracking performance. In [KKF+85], a feedforward compensation was learned to track the reference trajectory of each joint precisely. However, this method does not use the concept of ZMP, which means the walking stability is not guaranteed. Li et al. [LTK93] used the concept of ZMP and learned a compensative trunk motion based on the ZMP error from the previous complete walking trial. Yet, this method only plans the whole walking trial offline. To compensate for the ZMP error online, Kajita et al. [KKK+03] utilized the dynamical filter (DF), which calculates the expected ZMP error from the multibody dynamics to adjust the ZMP reference trajectory and uses the preview control [She66] to generate an adjusted CoM reference trajectory subsequently. The drawback of this method is that it requires the computation of the multibody dynamics. It means that it is model-based and has a heavy computation load.

Iterative learning control (ILC) is one of the widely used learning control algorithms, which is presented by Uchiyama [Uch78] originally and introduced by Arimoto et al. [AKM84] in English. Since humans can learn the experience from the repetitive trials to improve motion performance, the motivation of developing ILC is to empower the robots with these learning capabilities to improve the trajectory tracking performance. According to [AKM84, BTA06], ILC can reduce the trajectory tracking errors caused by the large repeating disturbances and model uncertainties. Inspired by ILC, Hu et al. [HOL16] applied an Online Iterative Learning Control approach (ZMP-OILC) to learn a compensative ZMP (CZMP) trajectory that reduced the ZMP error (i.e., improved the robustness of biped walking). The comparisons between the DF and the ZMP-OILC were presented in [HOL16]: *1.* In the case of short-period disturbance, both DF and ZMP-OILC can resist the disturbance. However, the ZMP of ZMP-OILC recovers slower; *2.* In the long-period disturbance scenario, the ZMP of ZMP-OILC still converged to the desired trajectory after few learning iterations, whereas there was a large ZMP deviation in the case of DF; *3.* During the non-repetitive transition phases, both DF and ZMP-OILC augmented with a pre-compensation procedure are suitable for non-repetitive transition phases. These comparisons show that DF and ZMP-OILC have similar but slightly different performance. Compared with DF, the main advantages of ZMP-OILC are model-free, lighter computation load, and online adaptation to unknown constant disturbances. However, both DF and ZMP-OILC use the ZMP-based walking algorithm, which needs a simplifying model assumption.

The concept of three dimensional Divergent Component of Motion (DCM) and Virtual Repellent Point (VRP) [EOAS15] has been used recently, where the CoM dynamics can be split into a stable dynamics (CoM converges to DCM) and an unstable dynamics (DCM diverges from VRP). This DCM-based walking algorithm can reduce the calculation load of the CoM reference trajectory generation without using any approximation models. Moreover, walking on uneven and compliant terrains can also be achieved by combining the DCM-based algorithm with the passivity-based whole-body controller [MEG$^+$19].

Inspired by ZMP-OILC and DCM framework, this research aims to develop an online VRP adaptation framework using the iterative learning control method for DCM-based walking controller, which is called VRP-OILC. The VRP-OILC learns an adjusted VRP reference trajectory from the repetitive walking motions and reduces the VRP error caused by the model inaccuracies or unmodeled external disturbances. In particular, two different implementations with stability and convergence analysis for VRP-OILC are proposed: measurement-error-based and commanded-error-based framework. These implementations present the learning from different VRP error signals. Furthermore, the VRP-OILC is also augmented with a predicted pre-compensation (or it can also be called pre-learned knowledge) to stabilize the initial two steps after the VRP-OILC is applied and to achieve a faster VRP convergence. This pre-learned knowledge is predicted from a database composed of the compensative VRP trajectories obtained from different walking trials. Unlike ZMP-OILC, our VRP-OILC does not need any approximated robot models and is compatible with force-controlled robots.

1.2 Outline

The thesis is organized as follows. Chapter 2 introduces the fundamentals of DCM-based walking and iterative learning control (ILC). Chapter 3 clarifies the principles and shows the stability proof and convergence analysis of the framework. The simulation and experimental results are evaluated in Chapter 4. Their discussions and interpretations are shown in Chapter 5. Chapter 6 concludes the thesis and provides some potential ideas for future work.

Moreover, a shortened version of this thesis was submitted for publication at the 2020 IEEE-RAS International Conference on Humanoid Robots.

Chapter 2

Technical Background

All the relevant technical background of this thesis is elaborated in the following sections. Section 2.1 describes the mechanism of the DCM-based biped walking algorithm. In particular, the concepts of DCM and VRP, the generation of reference trajectory, and the DCM-tracking control are introduced in this section. In Section 2.2, the general ILC approach and its stability analysis are present.

2.1 Bipedal Walking Based on Divergent Component of Motion

2.1.1 Fundamentals

Before we start with the introduction, one thing needs to be emphasized: there is no angular momentum balance strategy used in this work. In this case, the Center of Pressure (CoP, equivalent to ZMP) coincides with the Enhanced Centroidal Moment Pivot Point (eCMP) and the Centroidal Moment Pivot (CMP), as described in [EOAS15]. Then, the biped walking dynamics can be shown in Figure 2.1.

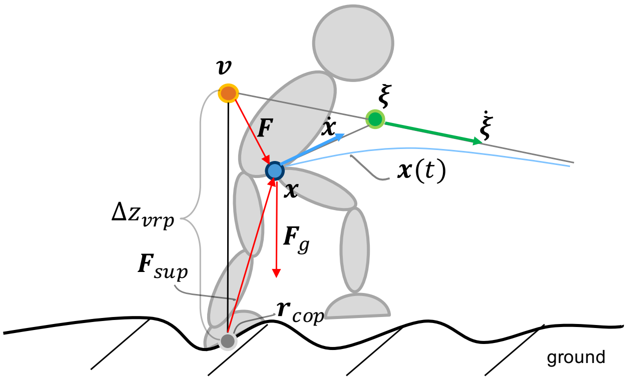


Figure 2.1: General humanoid robot dynamics.

As described in [EOAS15], the 3-D DCM $\boldsymbol{\xi}$ can be obtained by the CoM position $\boldsymbol{x}$ and its velocity $\dot{\boldsymbol{x}}$:

$$\boldsymbol{\xi} = \boldsymbol{x} + b\,\dot{\boldsymbol{x}}, \tag{2.1}$$

where $b = \sqrt{\Delta z_{vrp}/g}$ is the time constant of the DCM dynamics, with Δz_{vrp} is the distance in Z direction between VRP $\boldsymbol{v}$ and CoP $\boldsymbol{r}_{cop}$ on the contact foot and g is the gravitational constant. (2.1) can be reordered to:

$$\dot{\boldsymbol{x}} = -\frac{1}{b}(\boldsymbol{x} - \boldsymbol{\xi}). \tag{2.2}$$

The (2.2) is a stable first-order dynamics. As shown in Figure 2.1, the velocity of CoM points towards DCM. In other words, the CoM follows the DCM. To obtain the DCM dynamics, it is necessary to differentiate (2.1) as:

$$\dot{\boldsymbol{\xi}} = \dot{\boldsymbol{x}} + b\,\ddot{\boldsymbol{x}}. \tag{2.3}$$

Inserting Newton's 2nd law $\ddot{\boldsymbol{x}} = \boldsymbol{F}/m$ ($\boldsymbol{F}$ is the total force acting on the CoM, and m is the robot's total mass) into (2.3), it becomes:

$$\dot{\boldsymbol{\xi}} = \dot{\boldsymbol{x}} + \frac{b\boldsymbol{F}}{m}. \tag{2.4}$$

According to the concept of ZMP [VS72], the distributed ground reaction force acting on the robot's supporting foot can be replaced by a single support force acting on the ZMP. When there is no angular momentum on the hip of the robot, this support force points towards the CoM, which is represented as the $\boldsymbol{F}_{sup}$ shown in Figure 2.1. Basically, the total force $\boldsymbol{F}$ is the sum of the support force $\boldsymbol{F}_{sup}$ and the gravitational force $\boldsymbol{F}_g$ when no other external forces act on the robot.

According to [EOAS15], the VRP $\boldsymbol{v}$ encodes the total force $\boldsymbol{F}$ by:

$$\boldsymbol{F} = \frac{m}{b^2}(\boldsymbol{x} - \boldsymbol{v}). \tag{2.5}$$

Inserting (2.1) and (2.5) into (2.4), the DCM dynamics is obtained as:

$$\dot{\boldsymbol{\xi}} = \frac{1}{b}(\boldsymbol{\xi} - \boldsymbol{v}). \tag{2.6}$$

Here, the DCM dynamics is an unstable first-order dynamics when $b > 0$. As shown in Figure 2.1, the DCM is "pushed" away by VRP, while the CoM tracks the DCM.

2.1.2 Reference Trajectory Generation

The total robot walking motion can be split into n_φ transition phases' sequence[1]. There are $n_{\text{wp}} = n_\varphi + 1$ VRP, DCM and CoM waypoints respectively. In the transition phase $\varphi \in \{1, \dots, n_\varphi\}$, we denote the VRP trajectory by $\boldsymbol{v}_\varphi(t)$, the DCM

[1] One single support or double support phase is one transition phase.

trajectory by $\boldsymbol{\xi}_\varphi(t)$, and the CoM trajectory by $\boldsymbol{x}_\varphi(t)$. Here, t is the local time in each transition phase. Moreover, we use $\boldsymbol{v}_\varphi(0)$, $\boldsymbol{\xi}_\varphi(0)$ and $\boldsymbol{x}_\varphi(0)$ to represent the start waypoints, and $\boldsymbol{v}_\varphi(T_\varphi)$, $\boldsymbol{\xi}_\varphi(T_\varphi)$ and $\boldsymbol{x}_\varphi(T_\varphi)$ to represent the end waypoints in phase φ, where T_φ is the phase duration. For the case of the single support (SS) phase, the VRP reference remains at the same position, i.e., $\boldsymbol{v}_\varphi(0) = \boldsymbol{v}_\varphi(T_\varphi)$, whereas it does not remain at the same position for the double support (DS) phase. To ensure the continuity of the whole trajectories, the equations $\boldsymbol{v}_\varphi(0) = \boldsymbol{v}_{\varphi-1}(T_\varphi)$, $\boldsymbol{\xi}_\varphi(0) = \boldsymbol{\xi}_{\varphi-1}(T_\varphi)$ and $\boldsymbol{x}_\varphi(0) = \boldsymbol{x}_{\varphi-1}(T_\varphi)$ must be held for all the transition phases. The terminal DCM waypoint is denoted as $\boldsymbol{\xi}_f = \boldsymbol{\xi}_{n_\varphi}(T_\varphi)$.

Figure 2.2(a) shows an example of the VRP, DCM and CoM reference trajectory for 4-step walking. The points $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_{12}$ are the VRP waypoints, $\boldsymbol{\xi}_1, \ldots, \boldsymbol{\xi}_{12}$ are the DCM waypoints and $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{12}$ are the CoM waypoints. For example, the second transition phase ($\varphi = 2$) is a SS phase, which means that the start VRP waypoint $\boldsymbol{v}_2$ (is equivalent to $\boldsymbol{v}_2(0)$ of the reference trajectory) and the end VRP waypoint $\boldsymbol{v}_3$ in the figure (is equivalent to $\boldsymbol{v}_2(T_\varphi)$ of the reference trajectory) of the second transition phase are at the same position. $\boldsymbol{\xi}_2$ and $\boldsymbol{x}_2$ are the start DCM and CoM waypoints, whereas $\boldsymbol{\xi}_3$ and $\boldsymbol{x}_3$ are the end DCM and CoM waypoints respectively at this phase. In the DS phase for $\varphi = 3$, the start VRP waypoint $\boldsymbol{v}_3$ (is equivalent to $\boldsymbol{v}_3(0)$) and the end VRP waypoint $\boldsymbol{v}_4$ (is equivalent to $\boldsymbol{v}_3(T_\varphi)$) are not at the same position. $\boldsymbol{\xi}_3$ and $\boldsymbol{x}_3$ are the start DCM and CoM waypoints, whereas $\boldsymbol{\xi}_4$ and $\boldsymbol{x}_4$ are the end DCM and CoM waypoints.

In this section, the calculation for the reference trajectory of VRP, DCM and CoM is reported in a single transition phase φ.

VRP Trajectory

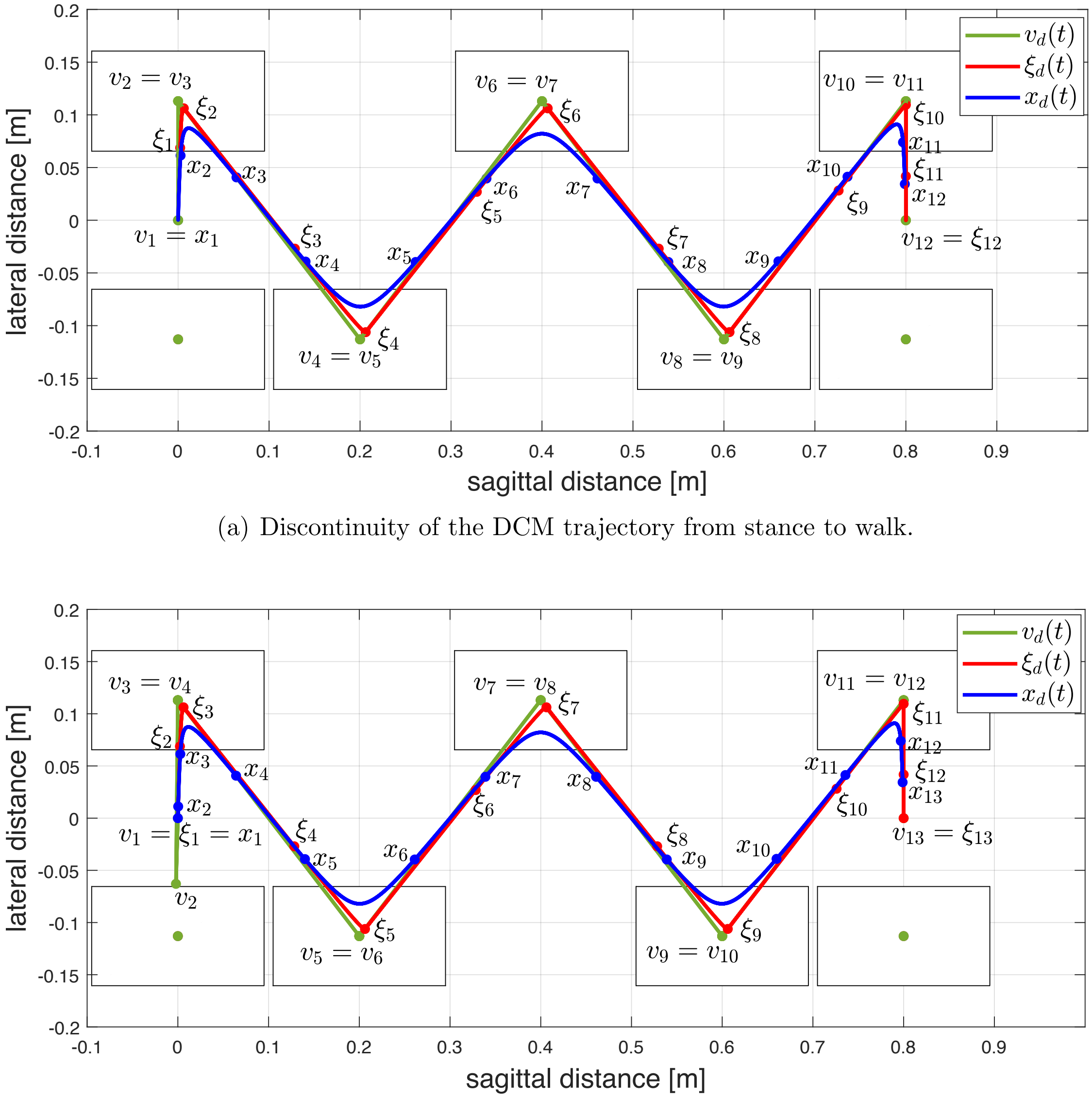
For generating the VRP trajectory $\boldsymbol{v}_\varphi(t)$ at phase φ, spatial linear interpolation between the start and end VRP waypoint is used. According to [EMO17], the general form of the interpolation can be expressed as:

$$\boldsymbol{v}_\varphi(t) = (1 - f_\varphi(t))\, \boldsymbol{v}_\varphi(0) + f_\varphi(t)\, \boldsymbol{v}_\varphi(T_\varphi),\ t \in [0, T_\varphi] \tag{2.7}$$

where $f_\varphi(t)$ is a time-dependent polynomial satisfying $f_\varphi(0) = 0$, $f_\varphi\,(T_\varphi) = 1$, and $0 \leqslant f_\varphi(t) \leqslant 1$ for all $t \in [0, T_\varphi]$. For different polynomial order n_{poly}, the expression of $f_\varphi(t)$ is different. For instance, the first-order polynomial is $f_\varphi(t) = t/T_\varphi$, whereas the third-order polynomial is represented as $f_\varphi(t) = (t/T_\varphi)^2\,(3 - 2t_\varphi/T_\varphi)$ and the fifth-order polynomial is expressed as $(t/T_\varphi)^3\left(10 - 15t/T_\varphi + 6t^2/T_\varphi^2\right)$.

DCM Trajectory

To obtain the DCM reference trajectory, Englsberger et al. [EMO17] inserted (2.7) into (2.6) and then solved the resulting differential equation. The closed-form solu-



(a) Discontinuity of the DCM trajectory from stance to walk.

(b) Continuity of the DCM trajectory from stance to walk.

Figure 2.2: Examples of 4-step VRP, DCM and CoM reference trajectory.

tion of the DCM trajectory is expressed as:

$$\begin{aligned}\boldsymbol{\xi}_{\varphi}(t) = &\underbrace{\left(1 - \sigma_{\varphi}(t) - e^{\frac{t-T_{\varphi}}{b}}\left(1 - \sigma_{\varphi}\left(T_{\varphi}\right)\right)\right)}_{\alpha_{\varphi,\xi}(t)} \boldsymbol{v}_{\varphi}(0) \\ &+ \underbrace{\left(\sigma_{\varphi}(t) - e^{\frac{t-T_{\varphi}}{b}}\sigma_{\varphi}\left(T_{\varphi}\right)\right)}_{\beta_{\varphi,\xi}(t)} \boldsymbol{v}_{\varphi}(T_{\varphi}) + \underbrace{e^{\frac{t-T_{\varphi}}{b}}}_{\gamma_{\varphi,\xi}(t)} \boldsymbol{\xi}_{\varphi}(T_{\varphi}),\end{aligned} \tag{2.8}$$

with

$$\sigma_{\varphi}(t) = \sum_{p=0}^{n_{poly}} \left(b^{p} \overset{(p)}{f_{\varphi}}(t)\right), \tag{2.9}$$

where $\overset{(p)}{f_{\varphi}}(t)$ represents the p^{th} time derivative of $f_{\varphi}(t)$. Note that the sum of $\alpha_{\varphi,\xi}(t)$, $\beta_{\varphi,\xi}(t)$ and $\gamma_{\varphi,\xi}(t)$ is always equal to 1. This coefficients' relation denotes the convex property of the DCM reference trajectory $\boldsymbol{\xi}_{\varphi}(t)$, i.e., the DCM trajectory is consistently within a triangle whose vertices are composed by $\boldsymbol{v}_{\varphi}(0)$, $\boldsymbol{v}_{\varphi}(T_{\varphi})$ and $\boldsymbol{\xi}_{\varphi}(T_{\varphi})$ (an example can be seen in Fig.3. of [MEOAS18]). By calculating the start DCM waypoint (at $t = 0$) at each transition phase recursively from the terminal waypoint $\boldsymbol{\xi}_{f} = \boldsymbol{\xi}_{n_{\varphi},T}$ with the equation (2.8), all the DCM waypoints can be obtained. The piecewise DCM reference trajectory $\boldsymbol{\xi}_{\varphi}(t)$ can be calculated by inserting the waypoints $\boldsymbol{v}_{\varphi}(0)$, $\boldsymbol{v}_{\varphi}(T_{\varphi})$ and $\boldsymbol{\xi}_{\varphi}(T_{\varphi})$ into the equation (2.8).

CoM Trajectory

According to [MEOAS18], the CoM reference trajectory is obtained by inserting (2.8) into (2.1) and solving the resulting differential equation subsequently. The solution for the CoM trajectory is:

$$\begin{aligned}\boldsymbol{x}_{\varphi}(t) = &\underbrace{\left(1 - \rho_{\varphi}(t) - \frac{1-\rho_{\varphi}(0)}{e^{\frac{t}{b}}} - \frac{e^{\frac{t}{b}} - e^{-\frac{t}{b}}}{2e^{\frac{T_{\varphi}}{b}}}\left(1 - \sigma_{\varphi}(T_{\varphi})\right)\right)}_{\alpha_{\varphi,x}(t)} \boldsymbol{v}_{\varphi}(0) \\ &+ \underbrace{\left(\rho_{\varphi}(t) - \frac{\rho_{\varphi}(0)}{e^{\frac{t}{b}}} - \frac{e^{\frac{t}{b}} - e^{-\frac{t}{b}}}{2e^{\frac{T_{\varphi}}{b}}}\sigma_{\varphi}(T_{\varphi})\right)}_{\beta_{\varphi,x}(t)} \boldsymbol{v}_{\varphi}(T_{\varphi}) \\ &+ \underbrace{\frac{e^{\frac{t}{b}} - e^{-\frac{t}{b}}}{2e^{\frac{T_{\varphi}}{b}}}}_{\gamma_{\varphi,x}(t)} \boldsymbol{\xi}_{\varphi}(T_{\varphi}) + \underbrace{e^{-\frac{t}{b}}}_{\delta_{\varphi,x}(t)} \boldsymbol{x}_{\varphi}(0),\end{aligned} \tag{2.10}$$

with

$$\rho_{\varphi}(t) = \sum_{p=0}^{\lfloor n_{poly}/2 \rfloor} \left(b^{2p} \overset{(2p)}{f_{\varphi}}(t)\right). \tag{2.11}$$

The CoM trajectory also has a convex property since the sum of $\alpha_{\varphi,x}(t)$, $\beta_{\varphi,x}(t)$, $\gamma_{\varphi,x}(t)$ and $\delta_{\varphi,x}(t)$ equals 1. As shown in Fig.4. of [MEOAS18], the CoM trajectory always stays within the triangular pyramid composed by $\boldsymbol{v}_{\varphi}(0)$, $\boldsymbol{v}_{\varphi}(T_{\varphi})$, $\boldsymbol{\xi}_{\varphi}(T_{\varphi})$ and $\boldsymbol{x}_{\varphi}(0)$.

When the robot is standing and not walking, the VRP, DCM, and CoM points should be in the same position. Notice that the start DCM waypoint $\boldsymbol{\xi}_1$ in Figure 2.2(a) does not coincide with the start VRP and CoM waypoint. This causes a discontinuity of the DCM trajectory when the robot starts walking. To avoid the discontinuity, Englsberger et al. [EMO17] calculated an additional VRP waypoint in the initial transfer phase to ensure the continuity of the DCM trajectory without violating the VRP boundary condition. An example of the continuous DCM trajectory and adjusted VRP trajectory can be seen in Figure 2.2(b). Here, the $\boldsymbol{\xi}_2$ in Figure 2.2(b) is not the same as the $\boldsymbol{\xi}_1$ in Figure 2.2(a) quantitatively, while the $\boldsymbol{\xi}_3$ in Figure 2.2(b) and the $\boldsymbol{\xi}_2$ in Figure 2.2(a) are the same. And the VRP waypoint $\boldsymbol{v}_2$ in Figure 2.2(b) is the additional point.

2.1.3 Divergent Component of Motion Tracking Control

For the DCM tracking control, [EOAS15] design a DCM tracking controller with closed-loop dynamics:

$$\underbrace{\dot{\boldsymbol{\xi}} - \dot{\boldsymbol{\xi}}_d}_{\dot{\boldsymbol{e}}_\xi} = -k_\xi \underbrace{(\boldsymbol{\xi} - \boldsymbol{\xi}_d)}_{\boldsymbol{e}_\xi}, \tag{2.12}$$

where $\dot{\boldsymbol{\xi}}_d$ is the reference DCM velocity, $\boldsymbol{\xi}$ is the measured DCM position and $\boldsymbol{\xi}_d$ denotes the reference DCM position. When $k_\xi > 0$, the $\boldsymbol{\xi}$ converges to $\boldsymbol{\xi}_d$. According to the DCM dynamics (2.6), the reference DCM velocity can be expressed as:

$$\dot{\boldsymbol{\xi}}_d = \frac{1}{b}\left(\boldsymbol{\xi}_d - \boldsymbol{v}_d\right). \tag{2.13}$$

Substituting (2.6) and (2.13) into (2.12) yields a DCM control law:

$$\boldsymbol{v} = \boldsymbol{v}_d + (1 + k_\xi b)\underbrace{(\boldsymbol{\xi} - \boldsymbol{\xi}_d)}_{\boldsymbol{e}_\xi}, \tag{2.14}$$

where k_ξ is the DCM controller gain. Englsberger et al. [EOAS15] designed $\boldsymbol{v} = \boldsymbol{v}_c$ to achieve the desired tracking behavior, where $\boldsymbol{v}_c$ is the commanded VRP. The commanded VRP converges to the desired VRP if the DCM error $\boldsymbol{e}_\xi$ equals 0, while the commanded VRP deviates from its reference in case of disturbances. This property can be used for the step adjustment during walking, as shown in [EOAS15].

A commanded force is calculated through substituting the commanded VRP $\boldsymbol{v}_c$ for $\boldsymbol{v}$ in (2.5). The commanded force is then used as input to a whole-body controller, e.g., the passivity-based whole-body controller [MEG+19]. The whole-body controller generates a corresponding commanded torque to control the robot.

2.1.4 Measurement of Virtual Repellent Point

There are two general ways to measure the model's VRP position: 1) measurement based on the force-torque (FT) sensors; 2) measurement based on the DCM dynamics model. The first method is suitable for the robots in which the FT sensors are installed in the feet. If a robot is not equipped with the FT sensors in the feet, the second method can be an alternative solution.

Measurement Based on Force-Torque Sensors

Assume there are two FT sensors mounted in the feet. The $\boldsymbol{F}_L$ and $\boldsymbol{F}_R$ are the measured force in the sensor's frame from the left and right FT sensor respectively. Except for the gravity and contact force with the ground, we assume that no other external forces exist. The total force $\boldsymbol{F}$ acting on the CoM can be expressed as:

$$\boldsymbol{F} = \sum_{p=L,R} \boldsymbol{R}^{p}_{CoM} \boldsymbol{F}_p - m\boldsymbol{g}, \tag{2.15}$$

where $\boldsymbol{R}^{p}_{CoM}$ is the rotation matrix from the CoM's coordinate frame to the frame of the FT sensor in the left or right foot, and $m\boldsymbol{g}$ is the total gravitational force of the robot. From (2.5) and (2.15), the measured VRP can be derived as:

$$\begin{aligned} \boldsymbol{v}_m &= \boldsymbol{x} - \frac{b^2}{m}\boldsymbol{F} \\ &= \boldsymbol{x} - \frac{b^2}{m} \sum_{p=L,R} \boldsymbol{R}^{p}_{CoM} \boldsymbol{F}_p - b^2 \boldsymbol{g}, \end{aligned} \tag{2.16}$$

where $\boldsymbol{x}$ is the measured CoM position. It can be estimated by IMU as shown in [HRO16].

Measurement Based on DCM Dynamics Model

From the DCM dynamics (2.6), the measured VRP $\boldsymbol{v}_m$ can also be derived as:

$$\boldsymbol{v}_m = \boldsymbol{\xi} - b\,\dot{\boldsymbol{\xi}}, \tag{2.17}$$

where $\boldsymbol{\xi}$ is the measured DCM and $\dot{\boldsymbol{\xi}}$ is the measured DCM velocity. As defined in (2.1), the measured DCM can be obtained by $\boldsymbol{\xi} = \boldsymbol{x} + b\,\dot{\boldsymbol{x}}$, where $\boldsymbol{x}$ and $\dot{\boldsymbol{x}}$ are the measured CoM position and velocity.

The measurement based on the FT sensors has a higher priority because it directly uses the sensor information to compute the VRP. In comparison, the measurement based on the DCM model is noisier because of the signal noise, as discussed in Chapter 5. Therefore, the first measurement method is recommended unless there are not FT sensors in robots' feet. Nevertheless, the drawback of the first method is that no external forces can be applied to the robot except gravity and contact forces between the feet and ground.

2.2 Iterative Learning Control

2.2.1 General Algorithm

Humans can learn a new sport or improve their sports performance by learning from repeating some motions. For instance, a basketball three-point shooter can improve the shooting accuracy by practicing the shot repeatedly. Inspired by this learning process, the ILC aims to control the robots to track the reference trajectory perfectly by learning from the previous repetitive trials.

Assume there is a system with the state-space representation as follows:

$$\begin{aligned} &\begin{cases} x(t+1,k) = Ax(t,k) + Bu(t,k) + w(t,k) \\ y(t,k) = Cx(t,k) + v(t,k) \end{cases} \\ &t = 0, 1, \ldots, T-1; \quad k = 1, 2, \ldots \end{aligned} \tag{2.18}$$

where x, u, y, w and v represents states, control inputs, system outputs, system uncertainties and noise from the measurement, with certain dimensions respectively. Moreover, t is the local time step in the k^{th} iteration and T is the iteration duration. The ultimate goal of applying ILC is to reduce the tracking error $e(t,k)$ to 0, where $e(t,k)$ is expressed by:

$$e(t,k) \hat{=} y_r(t) - y(t,k), \tag{2.19}$$

with $y_r(t)$ is the reference output trajectory. A general ILC formulation is described in [WGDI09] as:

$$u(t,k) = Q_{\text{ILC}}(u(t,k-1)) + L_{\text{ILC}}(e(t,k-1)), \tag{2.20}$$

where $Q_{\text{ILC}}(\cdot)$ and $L_{\text{ILC}}(\cdot)$ are called *Q-filter* and *L-filter* respectively. In general, the *Q-filter* determines how the current iteration's control input depends on the input of the last iteration, while the *L-filter* represents how the current control input should be adjusted according to the system output error from the last iteration. Moreover, the *Q-filter* and *L-filter* can have different representation forms. For example, if the control input u is a scalar (or a vector), the Q is called the *forgetting factor* (or *forgetting factor matrix*, is a diagonal matrix) when $Q_{\text{ILC}}(u) = Q \cdot u$. In this case, the Q is within $[0,1]$ (or all the entries on the diagonal of the Q are in this range). In the literature, $Q_{\text{ILC}}(u) = u$ is widely used as described in [WGDI09]. In terms of the *L-filter*, the different forms of *L-filter* lead to different types of ILC. For instance, $L_{\text{ILC}}(e(t,k-1)) = K_{\text{ILC}} \cdot e(t,k-1)$ implies a *P-type ILC* and $L_{\text{ILC}}(e(t,k-1)) = K_{\text{ILC}} \cdot (e(t,k-1) - e(t-1,k-1))$ leads to a *D-type ILC*. The K_{ILC} here is called the *learning gain* if the output error e is a scalar (or *learning gain matrix* if the e is a vector).

2.2.2 Stability Analysis

For the stability analysis of ILC, the lifted form [WGDI09, BTA06] is used. To this end, (2.20) is reformulated as follows:

$$\underbrace{\begin{bmatrix} u(0,k+1) \\ u(1,k+1) \\ \vdots \\ u(T-1,k+1) \end{bmatrix}}_{\mathbf{U}(k+1)} = \underbrace{\begin{bmatrix} Q_0 & Q_{-1} & \dots & Q_{-(T-1)} \\ Q_1 & Q_0 & \dots & Q_{-(T-2)} \\ \vdots & \vdots & \ddots & \vdots \\ Q_{T-1} & Q_{T-2} & \dots & Q_0 \end{bmatrix}}_{\mathbf{Q}} \underbrace{\begin{bmatrix} u(0,k) \\ u(1,k) \\ \vdots \\ u(T-1,k) \end{bmatrix}}_{\mathbf{U}(k)} + \underbrace{\begin{bmatrix} K_{\text{ILC},0} & K_{\text{ILC},-1} & \dots & K_{\text{ILC},-(T-1)} \\ K_{\text{ILC},1} & K_{\text{ILC},0} & \dots & K_{\text{ILC},-(T-2)} \\ \vdots & \vdots & \ddots & \vdots \\ K_{\text{ILC},T-1} & K_{\text{ILC},T-2} & \dots & K_{\text{ILC},0} \end{bmatrix}}_{\mathbf{K}_{\text{ILC}}} \underbrace{\begin{bmatrix} e(0,k) \\ e(1,k) \\ \vdots \\ e(T-1,k) \end{bmatrix}}_{\mathbf{E}(k)}, \tag{2.21}$$

where $\mathbf{Q}$ and $\mathbf{K}_{\text{ILC}}$ are the general matrix form of the *Q-filter* and *L-filter* respectively. As aforementioned, the $\mathbf{Q}$ and $\mathbf{K}_{\text{ILC}}$ have different forms according to the different types of the *Q-filter* and *L-filter*. For example, if $Q_{\text{ILC}}(u) = Q \cdot u$, the $\mathbf{Q}$ becomes:

$$\mathbf{Q} = \begin{bmatrix} Q & 0 & \dots & 0 \\ 0 & Q & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & Q \end{bmatrix}. \tag{2.22}$$

And if $L_{\text{ILC}}(e(t,k-1)) = K_{\text{ILC}} \cdot e(t,k-1)$, the $\mathbf{K}_{\text{ILC}}$ is:

$$\mathbf{K}_{\text{ILC}} = \begin{bmatrix} K_{\text{ILC}} & 0 & \dots & 0 \\ 0 & K_{\text{ILC}} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & K_{\text{ILC}} \end{bmatrix}. \tag{2.23}$$

From (2.18), the system output $y(t,k)$ can be rewritten as follows ($2 \leq t \leq T$):

$$\begin{aligned} y(t,k) &= Cx(t,k) + v(t,k) \\ &= CAx(t-1,k) + CBu(t-1,k) + Cw(t-1,k) + v(t,k) \\ &= CA^t x(0,k) + CA^{t-1}Bu(0,k) + CA^{t-2}Bu(1,k) + \ldots + CBu(t-1,k) \\ &\quad + CA^{t-1}w(0,k) + \ldots + Cw(t-1,k) + v(t,k). \end{aligned} \tag{2.24}$$

For $t = 1$, the system output is expressed as follows:

$$y(1,k) = CAx(0,k) + CBu(0,k) + Cw(0,k) + v(1,k). \tag{2.25}$$

The system output is represented for $t = 0$ as:

$$y(0,k) = Cx(0,k) + v(0,k). \tag{2.26}$$

The lifted form of the system output can be derived from (2.24), (2.25) and (2.26) as:

$$\underbrace{\begin{bmatrix} y(0,k) \\ y(1,k) \\ \vdots \\ y(T-1,k) \end{bmatrix}}_{\mathbf{Y}(k)} = \underbrace{\begin{bmatrix} 0 & 0 & \dots & 0 \\ CB & 0 & \dots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ CA^{T-2}B & \dots & CB & 0 \end{bmatrix}}_{\mathbf{A}} \underbrace{\begin{bmatrix} u(0,k) \\ u(1,k) \\ \vdots \\ u(T-1,k) \end{bmatrix}}_{\mathbf{U}(k)} + \underbrace{\begin{bmatrix} C \\ CA \\ \vdots \\ CA^{T-1} \end{bmatrix} x(0,k)}_{\mathbf{b}(k)}$$
$$+ \underbrace{\begin{bmatrix} 0 & 0 & \dots & 0 \\ C & 0 & \dots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ CA^{T-2} & \dots & C & 0 \end{bmatrix} \begin{bmatrix} w(0,k) \\ w(1,k) \\ \vdots \\ w(T-1,k) \end{bmatrix} + \begin{bmatrix} v(0,k) \\ v(1,k) \\ \vdots \\ v(T-1,k) \end{bmatrix}}_{\boldsymbol{\varepsilon}(k)}. \tag{2.27}$$

According to (2.19), (2.21) is rearranged by inserting (2.27) into (2.21) and obtain:

$$\mathbf{U}(k+1) = (\mathbf{Q} - \mathbf{K}_{\text{ILC}}\mathbf{A})\,\mathbf{U}(k) - \mathbf{K}_{\text{ILC}}\,(\mathbf{b}(k) - \mathbf{Y}_r + \boldsymbol{\varepsilon}(k))\,, \tag{2.28}$$

where

$$\mathbf{Y}_r = \begin{bmatrix} y_r(0) \\ y_r(1) \\ \vdots \\ y_r(T-1) \end{bmatrix}. \tag{2.29}$$

The stability of the system can be verified by analyzing the eigenvalues of $\mathbf{Q} - \mathbf{K}_{\text{ILC}}\mathbf{A}$ from (2.28). An operator $\rho(\square) = \max_i |\lambda_i(\square)|$ from [BTA06] is used to extract the maximum absolute eigenvalue of the matrix $\square$, where $\lambda_i(\square)$ is the i^{th} eigenvalue of $\square$. According to [NG02], the system is asymptotically stable if and only if:

$$\rho(\mathbf{Q} - \mathbf{K}_{\text{ILC}}\mathbf{A}) < 1. \tag{2.30}$$

The $\mathbf{Q} - \mathbf{K}_{\text{ILC}}\mathbf{A}$ is Toeplitz[2] [GS58] and low triangular when $\mathbf{Q}$ and $\mathbf{K}_{\text{ILC}}$ are causal[3]. In this case, the $\mathbf{Q} - \mathbf{K}_{\text{ILC}}\mathbf{A}$ has repetitive eigenvalues that are identical to its diagonal entries. So the stability condition is simplified as:

$$|\underset{i}{diag}(\mathbf{Q} - \mathbf{K}_{\text{ILC}}\mathbf{A})| < 1,\ i \in \{1, \dots, m\}, \tag{2.31}$$

[2] Toeplitz means that the diagonal entries of the matrix are the same.

[3] According to [BTA06], the ILC learning equation is causal if the $u(t,k)$ is depending on the $u(h, k-1)$ and $e(h, k-1)$, where $h \leq t$. For example, (2.20) is causal if $Q_{\text{ILC}}(u) = Q \cdot u$ and $L_{\text{ILC}}(e(t, k-1)) = K_{\text{ILC}} \cdot e(t, k-1)$.

where m is the number of diagonal entries of $\mathbf{Q} - \mathbf{K}_{\text{ILC}}\mathbf{A}$, and $\underset{i}{diag}(\square)$ selects the i^{th} diagonal entry of $\square$. Since $\mathbf{Q} - \mathbf{K}_{\text{ILC}}\mathbf{A}$ is Toeplitz, an arbitrary diagonal entry can be selected for the stability analysis in (2.31).

Chapter 3

Technical Approach

This chapter explains the online iterative learning control framework for bipedal walking robots in more detail. Section 3.1 elaborates on how the ILC is combined with the DCM-based walking algorithm in the framework. Section 3.2 proves the stability of the framework, and Section 3.3 demonstrates the convergence property of the VRP by using the framework.

3.1 Design of Online Iterative Learning Control Framework

3.1.1 Definition of Iteration

Humanoid robots' bipedal walking can be considered as a repetitive locomotion behavior of the legs. This implies the possibility of combining the ILC with a DCM-based bipedal walking algorithm. In the framework, one iteration is defined as one walking cycle, where the robot starts on the first foot, goes to the second foot, and then ends back to the first foot. It means that one iteration contains 2 SS phases and 2 DS phases. For instance, Figure 3.1 shows a bipedal walking in $(i-1)^{th}$ and i^{th} iteration. We assume the current iteration is $(i-1)^{th}$ iteration. Here, the iteration starts at the beginning of the SS phase of the left foot step and ends at the end of the DS phase from the right foot step to the next left foot step. Specifically, the robot's walking starts from the 0^{th} iteration. In Figure 3.1, $\boldsymbol{G}_0$ is the global coordinate frame, $\boldsymbol{G}_{i-1}$ and $\boldsymbol{G}_i$ are the local coordinate frames of $(i-1)^{th}$ and i^{th} iteration respectively. The orange lines between the feet's centers are the desired VRP trajectory, and the angle $\Delta\alpha$ represents the rotation from frame $\boldsymbol{G}_{i-1}$ to frame $\boldsymbol{G}_i$ along the Z axis. Since the stepping behavior is repetitive in the local frame, the rotation matrix $\boldsymbol{R}_\Delta$ that transforms the learned information from $\boldsymbol{G}_{i-1}$ to $\boldsymbol{G}_i$

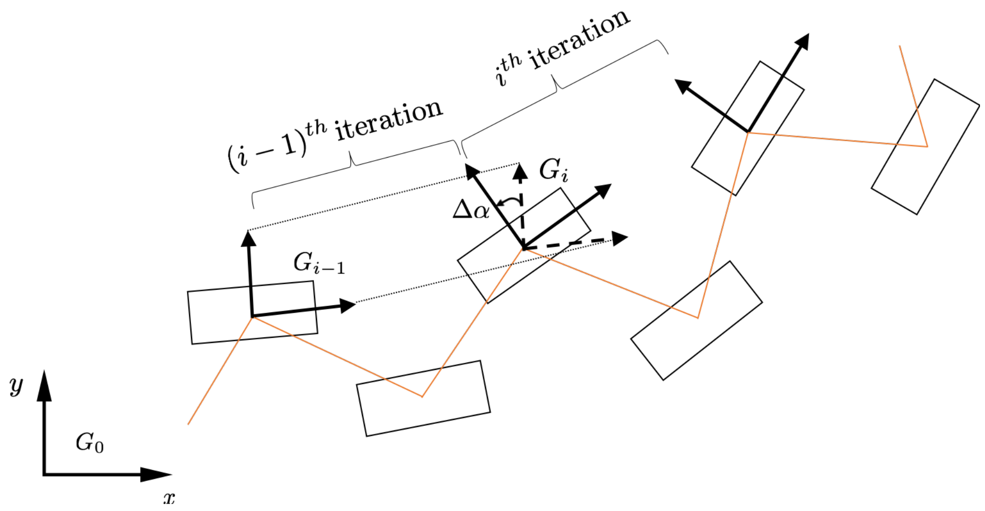


Figure 3.1: An example of two walking iterations.

can be expressed as:

$$\boldsymbol{R}_{\Delta} = \boldsymbol{R}_{Z}(\Delta\alpha) = \begin{bmatrix} \cos\Delta\alpha & -\sin\Delta\alpha & 0 \\ \sin\Delta\alpha & \cos\Delta\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.1}$$

3.1.2 Measurement-Error-Based Reference Trajectories Adaptation

Our learning framework is an online VRP-waypoint-based adaptation approach. The system adjusts the future VRP reference trajectory by using ILC to learn from the current VRP measurement error. In this subsection, our online ILC framework (VRP-OILC) is elaborated. Firstly, three essential definitions for the explanation of the VRP-OILC are shown in Definition 3.1.1 - 3.1.3:

Definition 3.1.1 (VRP Measurement Error)**.** A VRP measurement error is defined as the difference between the desired and measured VRP, i.e., $\boldsymbol{v}_d - \boldsymbol{v}_m$.

Definition 3.1.2 (Learned VRP Waypoint)**.** A learned VRP waypoint is a discrete reference VRP point on the learned VRP reference trajectory. This reference trajectory is learned using the ILC updating law (3.2).

Definition 3.1.3 (ILC Update Interval)**.** An ILC update interval Δt_{ILC} is defined as the time interval between two ILC update cycles. For instance, if $\Delta t_{ILC} = 0.01$ s, the VRP-OILC framework obtains a learned VRP waypoint every 0.01 s. Note that the relation between Δt_{ILC} and the system sampling time[1] Δt_s should be: $\Delta t_{ILC} \geq \Delta t_s$.

[1] The sampling time of controlling the whole robot system.

Algorithm 1: Pseudocode of VRP-OILC

```
Input: Step Information.
Output: Adjusted VRP and DCM Trajectory ($\boldsymbol{v}_{l,i}(t)$ and $\boldsymbol{\xi}_{l,i}(t)$).
$i \leftarrow \lfloor t_g / T_{iter} \rfloor$
$t \leftarrow t_g$ rem $T_{iter}$
if $i = 0$ and $t = 0$ then
    Initialize $\boldsymbol{V}_l$: $\boldsymbol{V}_l \leftarrow [\boldsymbol{v}_{d,0}(0),\ \boldsymbol{v}_{d,0}(\Delta t_{ILC}),\ \dots,\ \boldsymbol{v}_{d,0}(T_{iter} - \Delta t_{ILC})]$
    $\boldsymbol{v}_{l,i}(t) \leftarrow \boldsymbol{V}_l(1)$
    $\boldsymbol{\xi}_{l,i}(t) \leftarrow \boldsymbol{\xi}_{d,0}(0)$
else
    $k \leftarrow \lfloor t / \Delta t_{ILC} \rfloor$
    $t_{in\_w} \leftarrow t$ rem $k \Delta t_{ILC}$
    if $t_{in\_w} = 0$ then
        if $\varphi_c + 4 \leq n_\varphi$ then
            $\boldsymbol{v}_{l,i}(t) \leftarrow \boldsymbol{V}_l(2)$
            $\boldsymbol{v}_{l,i+1}(t) \leftarrow \boldsymbol{v}_{d,i+1}(t) + k_f \boldsymbol{R}_\Delta \left(\boldsymbol{v}_{l,i}(t) - \boldsymbol{v}_{d,i}(t)\right) + k_l \boldsymbol{R}_\Delta \left(\boldsymbol{v}_{d,i}(t) - \boldsymbol{v}_{m,i}(t)\right)$
            $\boldsymbol{V}_l' \leftarrow [\boldsymbol{V}_l(2 : end),\ \boldsymbol{v}_{l,i+1}(t)]$
        else
            $\boldsymbol{V}_l' \leftarrow [\boldsymbol{V}_l(2 : end),\ \boldsymbol{V}_l(end)]$
        $\boldsymbol{V}_l \leftarrow \boldsymbol{V}_l'$
    $\boldsymbol{v}_{l,i}(t) \leftarrow (1 - f_{ILC}(t_{in\_w}))\, \boldsymbol{V}_l(1) + f_{ILC}(t_{in\_w}) \boldsymbol{V}_l(2)$
    if $\varphi_c + 4 \leq n_\varphi$ then
        $\boldsymbol{\xi}_f \leftarrow \boldsymbol{\xi}_{d,i+1}(k \Delta t_{ILC})$
    else
        $\boldsymbol{\xi}_f \leftarrow \boldsymbol{V}_l(end)$
    $\boldsymbol{\xi}_{mid} \leftarrow \boldsymbol{\xi}_f$
    for $j \leftarrow \frac{T_{iter}}{\Delta t_{ILC}} : -1 : 3$ do
        $\boldsymbol{\xi}_{mid} \leftarrow \alpha_{ILC,\xi}(0) \boldsymbol{V}_l(j-1) + \beta_{ILC,\xi}(0) \boldsymbol{V}_l(j) + \gamma_{ILC,\xi}(0) \boldsymbol{\xi}_{mid}$
    $\boldsymbol{\xi}_{l,i}(t) \leftarrow \alpha_{ILC,\xi}(t_{in\_w}) \boldsymbol{V}_l(1) + \beta_{ILC,\xi}(t_{in\_w}) \boldsymbol{V}_l(2) + \gamma_{ILC,\xi}(t_{in\_w}) \boldsymbol{\xi}_{mid}$
$t_g \leftarrow t_g + \Delta t_s$
```

General Framework Procedure

In the VRP-OILC, a learned VRP waypoint list is used, which can be used as a buffer to save the learned VRP waypoints and calculate the adjusted DCM reference trajectory. The learned VRP waypoints' number in the list is $\frac{T_{iter}}{\Delta t_{ILC}}$, where T_{iter} is an iteration duration and Δt_{ILC} is an ILC update interval. For example, if the iteration duration is $T_{iter} = 2.4$ s and the ILC updating interval is $\Delta t_{ILC} = 0.01$ s, the learned VRP waypoint list always has 240 learned VRP waypoints. The $\boldsymbol{V}_l$ is used as the learned VRP waypoint list before updated by the ILC updating law, and

$\boldsymbol{V}_l'$ represents the updated learned VRP waypoint list after using the ILC updating law. A pseudocode of the VRP-OILC is presented in Algorithm 1.

Explanation of Algorithm 1: The general idea of VRP-OILC is elaborated line by line as follows:

Lines 1 - 2: The iteration index i and the local time t within an iteration are calculated. Note that $t_g \in \mathbb{R}$ is a global time elapsed from the start of the VRP-OILC. Here, the iteration index is expressed using a floor operator $\lfloor\cdot\rfloor$, while the local time is obtained by a remainder operator **rem**. For instance, $\lfloor 2.5 \rfloor = 2$, and $5 \textbf{ rem } 2 = 1$.

Lines 3 - 6: When $i = 0$ and $t = 0$, it is at the start time step of VRP-OILC (or 0^{th} iteration). At this time step, the learned VRP waypoint list $\boldsymbol{V}_l$ is initialized by using the 0^{th} iteration's VRP original reference trajectory. Particularly, the trajectory is discretized by using the ILC updating interval Δt_{ILC} as sampling time, then place all the sampled VRP reference points $[\boldsymbol{v}_{d,0}(0),\ \boldsymbol{v}_{d,0}(\Delta t_{ILC}),\ \ldots,\ \boldsymbol{v}_{d,0}(T_{iter} - \Delta t_{ILC})]$ into $\boldsymbol{V}_l$. After the $\boldsymbol{V}_l$ is initialized, the first waypoint of the $\boldsymbol{V}_l$ (i.e., $\boldsymbol{v}_{d,0}(0)$) is selected as the current VRP reference point. Also, the original DCM reference point $\boldsymbol{\xi}_{d,0}(0)$ is treated as the current DCM reference point. Here, the current VRP and DCM reference point mean the adjusted VRP and DCM reference trajectory (i.e., $\boldsymbol{v}_{l,i}(t)$ and $\boldsymbol{\xi}_{l,i}(t)$) at $t = 0$ and $i = 0$, which are the adjusted reference trajectories the robot needs to track in DCM-based walking. In short, the adjusted VRP and DCM reference trajectory are equivalent to their original reference trajectory when $t = 0$ and $i = 0$.

Lines 7 - 26: These lines represent the trajectory adjustment case. Specifically, lines 8 - 9 show the calculation of the learned waypoint index k and the elapsed local time t_{in_w} between two learned waypoints. Since the system sampling time Δt_s and the ILC updating interval Δt_{ILC} might be different, the index k and $k+1$ are used to determine which two learned waypoints the current position is between, while the time $t_{in\_w} \in [0, \Delta t_{ILC})$ denotes how long it elapsed after the last ILC update (i.e., the local time between current and next waypoint). For instance, if $i = 1$, $t = 0.032$ s, $\Delta t_{ILC} = 0.01$ s and $\Delta t_s = 0.001$ s, then $k = 3$ and $t_{in\_w} = 0.002$ s. For the VRP trajectory, it means that the current adjusted VRP trajectory is between the 3^{rd} VRP learned waypoint $\boldsymbol{v}_{l,i}(3\Delta t_{ILC})$ and the 4^{th} waypoint $\boldsymbol{v}_{l,i}(4\Delta t_{ILC})$, and 0.002 s has passed inside the period between the 3^{rd} and 4^{th} waypoint (i.e., since the last ILC update).

Lines 10 - 17 denotes the update process of the learned VRP waypoint list at $t_{in\_w} = 0$. Particularly, lines 11 - 14 present the learning process when the robot will keep walking within the 4 transition phases in the future (i.e., $\varphi_c + 4 \leq n_\varphi$, with current transition phase index φ_c and total transition phase's number n_φ). As shown in lines 12 - 13, the next iteration's reference VRP waypoint $\boldsymbol{v}_{l,i+1}(t)$ is adjusted by:

$$\boldsymbol{v}_{l,i+1}(t) = \boldsymbol{v}_{d,i+1}(t) + k_f \boldsymbol{R}_\Delta \left(\boldsymbol{v}_{l,i}(t) - \boldsymbol{v}_{d,i}(t)\right) + k_l \boldsymbol{R}_\Delta \left(\boldsymbol{v}_{d,i}(t) - \boldsymbol{v}_{m,i}(t)\right), \tag{3.2}$$

where $\boldsymbol{v}_{m,i}(t)$ is the measured VRP trajectory, $0 < k_f \leq 1$ is the forgetting factor and $k_l > 0$ is the learning gain. In line 14, the $\boldsymbol{V}_l'$ is obtained by discarding the first waypoint, then inserting the learned VRP future waypoint $\boldsymbol{v}_{l,i+1}(t)$ into the end of the waypoint list. In this way, the learned future waypoint is reused to calculate the adjusted VRP reference trajectory when it entries the next iteration at the same local time step t. When the robot stops walking within the 4 transition phases in the future, the waypoint list is updated by replicating the last waypoint of the waypoint list, as shown in line 16.

After the waypoint list is updated, the current VRP reference trajectory $\boldsymbol{v}_{l,i}(t)$ is adjusted by the linear interpolation between $\boldsymbol{V}_l(1)$ and $\boldsymbol{V}_l(2)$ in line 18, where $f_{ILC}(\cdot)$ uses the ILC updating interval Δt_{ILC} as the time duration in the polynomial. Lines 19 - 26 show the process of obtaining the current adjusted DCM reference trajectory $\boldsymbol{\xi}_{l,i}(t)$. In particular, lines 19 - 22 determine the terminal DCM point for the calculation of $\boldsymbol{\xi}_{l,i}(t)$ bases on whether the robot will stop within the next 4 transition phases, while lines 23 - 26 elaborate the backward calculation from terminal DCM point to the current adjusted DCM reference point by using the DCM trajectory calculation (2.8) and all the learned VRP waypoints in $\boldsymbol{V}_l$. Again, the $\alpha_{ILC,\xi}(\cdot)$, $\beta_{ILC,\xi}(\cdot)$ and $\gamma_{ILC,\xi}(\cdot)$ in lines 25 - 26 replace the transition phase duration T_φ in (2.8) with the ILC updating interval Δt_{ILC}. In the end, the $\boldsymbol{v}_{l,i}(t)$ and $\boldsymbol{\xi}_{l,i}(t)$ are the outputs of the VRP-OILC.

The outputs of the VRP-OILC are the inputs to the DCM-controller. Here, the DCM control law (2.14) is modified by substituting the adjusted reference trajectories for the original desired trajectories as follows:

$$\boldsymbol{v}_c = \boldsymbol{v}_l + (1 + k_\xi b)\,(\boldsymbol{\xi} - \boldsymbol{\xi}_l). \tag{3.3}$$

In summary, the VRP-OILC framework learns the future VRP waypoint from the VRP error at the current iteration. This future waypoint is stored in the learned waypoint list $\boldsymbol{V}_l$ and reused when it entries the corresponding future iteration at the same local time step. In this way, on the one hand, the current adjusted VRP reference trajectory is not affected by the newly learned future waypoint, and on the other hand, the current adjusted DCM reference trajectory is smoother than the one in the failed attempt described in "*Learning from Previous Iteration*“ of Section 3.1.5. These properties of the VRP-OILC guarantee the stability of the whole robot system.

Pre-learning Procedure for Fast Walking

After the VRP-OILC framework is used, the robot's walking behavior is wobbling during the 0^{th} iteration's two steps for fast walking[2]. Inspired by the precompensation procedure for the ZMP adaptation in [HOL16], a pre-learning procedure for the initialization of the learned VRP waypoint list $\boldsymbol{V}_l$ at the beginning of the 0^{th} iteration is designed to enhance the walking robustness in this iteration for fast walking.

Specifically, a database of the pre-learned knowledge is constructed for different walking parameters firstly. To learn the database, walking trials under a designed sample walking parameter vector set $\boldsymbol{\Theta} = \{\boldsymbol{\theta}_i | i = 1, \ldots, m\}$ are conducted, where m is the number of the different parameter vectors, and parameter vector $\boldsymbol{\theta}_i$ consists of n walking parameters, e.g., the SS time, DS time and the step distance in sagittal direction are examples of the walking parameter. During the walking experiment under each parameter vector $\boldsymbol{\theta}_i$, the compensative VRP trajectory is extracted only at the beginning of the 1^{st} iteration (i.e., at the moment of $t = 0$ and $i = 1$). The compensative VRP trajectory $\Delta\boldsymbol{V}$ is defined as:

$$\Delta\boldsymbol{V} = \boldsymbol{R}_1^T \left[\boldsymbol{V}_{l,1} - \boldsymbol{V}_{d,1}\right], \tag{3.4}$$

where

$$\boldsymbol{V}_{l,1} = [\boldsymbol{v}_{l,1}(0),\ \boldsymbol{v}_{l,1}(\Delta t_{ILC,i}),\ \boldsymbol{v}_{l,1}(2\Delta t_{ILC,i}),\ \ldots,\ \boldsymbol{v}_{l,1}(T_{iter,i} - \Delta t_{ILC,i})], \tag{3.5}$$

$$\boldsymbol{V}_{d,1} = [\boldsymbol{v}_{d,1}(0),\ \boldsymbol{v}_{d,1}(\Delta t_{ILC,i}),\ \boldsymbol{v}_{d,1}(2\Delta t_{ILC,i}),\ \ldots,\ \boldsymbol{v}_{d,1}(T_{iter,i} - \Delta t_{ILC,i})], \tag{3.6}$$

and $\boldsymbol{R}_1$ is the rotation matrix from the global frame to local frame $\boldsymbol{G}_1$. Moreover, the $\Delta t_{ILC,i}$ and $T_{iter,i}$ in (3.5) and (3.6) represent the ILC updating interval and iteration duration for the walking under the parameter vector $\boldsymbol{\theta}_i$ respectively. The $\boldsymbol{V}_{l,1}$ is the learned VRP waypoint list extracted at the moment of $t = 0$ and $i = 1$, so all the waypoints in the list are the learned VRP waypoints only from the 1^{st} iteration. To obtain the proper compensative VRP trajectory $\Delta\boldsymbol{V}$, the $\boldsymbol{V}_{d,1}$ contains the corresponding original VRP reference waypoint from the 1^{st} iteration, as shown in (3.6). The compensative VRP trajectory is marked as $\Delta\boldsymbol{V}(\boldsymbol{\theta}_i)$ according to the parameter vector $\boldsymbol{\theta}_i$ and saved in the database. In the end, the database can be considered as a lookup table of the compensative VRP trajectories corresponding to different walking parameter vectors.

Assume the robot is running under a new walking parameter vector $\boldsymbol{\theta}^* = [\theta_1^*, \ldots, \theta_n^*]$. A k-nearest neighbors algorithm (k-NN) is used to predict a pre-learned knowledge $\Delta\boldsymbol{V}(\boldsymbol{\theta}^*)$ for the initialization of the $\boldsymbol{V}_l$ in line 4 in Algorithm 1. The prediction of the pre-learned knowledge $\Delta\boldsymbol{V}(\boldsymbol{\theta}^*)$ is conducted as follows:

[2] The DLR humanoid robot TORO cannot walk with a SS time less than 0.9 s and a DS time less than 0.3 s without applying the VRP-OILC framework. Fast walking means the case where the SS time is less than 0.9 s and the DS time is less than 0.3 s.

1) The k nearest parameter vectors from the database are selected based on the Mahalanobis distance in the following form [HOL16]:

$$d\left(\boldsymbol{\theta}_i, \boldsymbol{\theta}^*\right) = \sqrt{\left(\boldsymbol{\theta}_i - \boldsymbol{\theta}^*\right)^T \boldsymbol{\Sigma} \left(\boldsymbol{\theta}_i - \boldsymbol{\theta}^*\right)}, \tag{3.7}$$

where $\boldsymbol{\Sigma}$ denotes the covariance matrix of the walking parameter vector set $\boldsymbol{\Theta}$.

2) The weight of each selected data is computed as:

$$\omega\left(\boldsymbol{\theta}_i\right) = \frac{1/d\left(\boldsymbol{\theta}_i, \boldsymbol{\theta}^\star\right)}{\sum_{j=1}^{k} 1/d\left(\boldsymbol{\theta}_j, \boldsymbol{\theta}^\star\right)}, \ i = 1, \dots, k \tag{3.8}$$

3) To match the size of the learned VRP waypoint list of the walking under the new walking parameter vector $\boldsymbol{\theta}^*$, the candidate pre-learned knowledge $\Delta\boldsymbol{V}(\boldsymbol{\theta}_i)$ must be modified as:

$$\Delta_m \boldsymbol{V}(j; \boldsymbol{\theta}_i) = \Delta\boldsymbol{V}(\lfloor \frac{jT_{iter,i}}{T^*_{iter}} \rfloor; \boldsymbol{\theta}_i), \ j = 1, \dots, \frac{T^*_{iter}}{\Delta t^*_{ILC}}, \tag{3.9}$$

where $\Delta_m \boldsymbol{V}(j; \boldsymbol{\theta}_i)$ denotes the j^{th} entry of the modified pre-learned knowledge, T^*_{iter} and Δt^*_{ILC} are the iteration duration and ILC updating interval under the parameter vector $\boldsymbol{\theta}^*$ respectively.

4) The pre-learned knowledge of the parameter vector $\boldsymbol{\theta}^*$ is predicted as:

$$\Delta\boldsymbol{V}(\boldsymbol{\theta}^*) = \sum_{i=1}^{k} \omega\left(\boldsymbol{\theta}_i\right) \Delta_m \boldsymbol{V}(\boldsymbol{\theta}_i), \tag{3.10}$$

After the pre-learned knowledge $\Delta\boldsymbol{V}(\boldsymbol{\theta}^*)$ is predicted, the VRP waypoint list $\boldsymbol{V}_l$ for the step described in line 4 in Algorithm 1 is initialized by:

$$\boldsymbol{V}_l = \boldsymbol{V}_{d,0}(\boldsymbol{\theta}^*) + \boldsymbol{R}_0 \, \Delta\boldsymbol{V}(\boldsymbol{\theta}^*), \tag{3.11}$$

where

$$\boldsymbol{V}_{d,0}(\boldsymbol{\theta}^*) = \left[\boldsymbol{v}_{d,0}(0), \ \boldsymbol{v}_{d,0}(\Delta t^*_{ILC}), \ \boldsymbol{v}_{d,0}(2\Delta t^*_{ILC}), \ \dots, \ \boldsymbol{v}_{d,0}(T^*_{iter} - \Delta t^*_{ILC})\right], \tag{3.12}$$

and $\boldsymbol{R}_0$ is the rotation matrix from the global frame to $\boldsymbol{G}_0$.

3.1.3 Commanded-Error-Based Reference Trajectories Adaptation

Line 13 of the Algorithm 1 in Section 3.1.2 shows the ILC updating law learning from the VRP measurement error. Unfortunately, the FT sensors of the humanoid robot TORO were not available for the experimental phase of this thesis. As an alternative solution, the VRP measurement based on the DCM model described in Section 2.1.4 was used for the ILC updating law. The results in Section 4.2 shows that this alternative method made the robot always fall during the experiment. The reasons for the falling are the noisy measurements and the overuse of the low pass filters, as discussed in Chapter 5. So the alternative measurement solution cannot be used further. Instead of the VRP measurement error, the VRP commanded error is used as a compromise. The VRP commanded error is defined as:

Definition 3.1.4 (VRP Commanded Error)**.** A VRP commanded error is the difference between the desired and commanded VRP, i.e., $\boldsymbol{v}_d - \boldsymbol{v}_c$.

Specifically, the VRP measurement error is replaced by the VRP commanded error in line 13 of the Algorithm 1 as a proposed commanded-error-based reference trajectories adaptation of the VRP-OILC framework. It means that the ILC updating law in commanded-error-based framework becomes as:

$$\begin{aligned}\boldsymbol{v}_{l,i+1}(t-\Delta t_s) &= \boldsymbol{v}_{d,i+1}(t-\Delta t_s) + k_f \boldsymbol{R}_\Delta \left(\boldsymbol{v}_{l,i}(t-\Delta t_s) - \boldsymbol{v}_{d,i}(t-\Delta t_s)\right) \\ &\quad + k_l \boldsymbol{R}_\Delta \left(\boldsymbol{v}_{d,i}(t-\Delta t_s) - \boldsymbol{v}_{c,i}(t-\Delta t_s)\right), \ t \in [\Delta t_s, T_{iter}], \ i \geq 0.\end{aligned} \tag{3.13}$$

Notice that all the VRP quantities used in (3.13) are from the last time step (i.e., time step $t - \Delta t_s$). The reason is that the current commanded VRP $\boldsymbol{v}_{c,i}(t)$ is only obtained after the adjusted VRP and DCM reference trajectory $\boldsymbol{v}_{l,i}(t)$ and $\boldsymbol{\xi}_{l,i}(t)$ are calculated by VRP-OILC and inserted as inputs to the DCM-controller. Since $\boldsymbol{v}_{l,i+1}(t - \Delta t_s)$ is the learned future waypoint, it is used to replace the $\boldsymbol{v}_{l,i+1}(t)$ in line 14 of Algorithm 1 for the framework. The ILC updating law (3.13) and the time step where the learning is executed are the only differences between the commanded-error-based and measurement-error-based reference trajectory adaptation. Moreover, we also prove the stability of the commanded-error-based framework in Section 3.2.

3.1.4 Convergence Condition

Hu et al. [HOL16] evaluated their system convergence by calculating and comparing the average ZMP deviation of each iteration. If the error between two iterations is smaller than a threshold value, then the system converges. Inspired by this idea, an average VRP error is used to evaluate the performance of the VRP-OILC framework. Since there are two reference trajectories adaptation implementations as aforementioned, two kinds of average VRP error are defined: average measured and commanded VRP error. The average measured VRP error $e_{m,i}$ of the i^{th} iteration is expressed as:

$$e_{m,i} = \frac{1}{T_{iter}} \int_0^{T_{iter}} \left|\boldsymbol{v}_{d,i}(t) - \boldsymbol{v}_{m,i}(t)\right| dt. \tag{3.14}$$

And the average commanded VRP error $e_{c,i}$ is calculated as:

$$e_{c,i} = \frac{1}{T_{iter}} \int_0^{T_{iter}} |\boldsymbol{v}_{d,i}(t) - \boldsymbol{v}_{c,i}(t)| \, dt. \tag{3.15}$$

3.1.5 Failed Design

Before we came up with the current VRP-OILC framework, we designed various frameworks with many different failed trials. Here, two important failed designs are introduced, which are the milestones on the path to the current framework.

Learning from Previous Iteration

Assume the current iteration index is i, and the current transition phase is φ. Inspired by the ILC learning control law in [HOL16], the VRP reference trajectory is adjusted as:

$$\boldsymbol{v}_{l,i}(t) = \boldsymbol{v}_{d,i}(t) + k_f \boldsymbol{R}_{i-1}^{i} \left(\boldsymbol{v}_{l,i-1}(t) - \boldsymbol{v}_{d,i-1}(t)\right) + k_l \boldsymbol{R}_{i-1}^{i} \left(\boldsymbol{v}_{d,i-1}(t) - \boldsymbol{v}_{m,i-1}(t)\right), \ i \geq 1, \tag{3.16}$$

where $\boldsymbol{R}_{i-1}^{i}$ here is the rotation matrix from the frame $\boldsymbol{G}_{i-1}$ to $\boldsymbol{G}_i$. The adjusted VRP reference trajectory $\boldsymbol{v}_{l,i}(t)$ is learned from the last iteration's VRP error $\boldsymbol{v}_{d,i-1}(t) - \boldsymbol{v}_{m,i-1}(t)$. The VRP difference trajectory $\Delta\boldsymbol{v}(t)$ between $\boldsymbol{v}_{l,i}(t)$ and $\boldsymbol{v}_{d,i}(t)$ can be derived from (3.16) as:

$$\begin{aligned} \Delta\boldsymbol{v}(t) = \boldsymbol{v}_{l,i}(t) - \boldsymbol{v}_{d,i}(t) = k_f \boldsymbol{R}_{i-1}^{i} \left(\boldsymbol{v}_{l,i-1}(t) - \boldsymbol{v}_{d,i-1}(t)\right) \\ + k_l \boldsymbol{R}_{i-1}^{i} \left(\boldsymbol{v}_{d,i-1}(t) - \boldsymbol{v}_{m,i-1}(t)\right), \ i \geq 1. \end{aligned} \tag{3.17}$$

The $\Delta\boldsymbol{v}(t)$ is then used to adjust the start VRP waypoint $\boldsymbol{v}_{d,i,\varphi}^{s}$ and the end VRP waypoint $\boldsymbol{v}_{d,i,\varphi}^{e}$ of the current transition phase φ as:

$$\boldsymbol{v}_{l,i,\varphi}^{s}(t) = \boldsymbol{v}_{d,i,\varphi}^{s} + \Delta\boldsymbol{v}(t), \tag{3.18}$$

$$\boldsymbol{v}_{l,i,\varphi}^{e}(t) = \boldsymbol{v}_{d,i,\varphi}^{e} + \Delta\boldsymbol{v}(t), \tag{3.19}$$

where $\boldsymbol{v}_{l,i,\varphi}^{s}(t)$ and $\boldsymbol{v}_{l,i,\varphi}^{e}(t)$ denote the learned VRP start and end waypoint of the φ^{th} transition phase which is at the i^{th} iteration, respectively. Note that the $\boldsymbol{v}_{l,i,\varphi}^{s}(t)$ and $\boldsymbol{v}_{l,i,\varphi}^{e}(t)$ are both functions of time because $\Delta\boldsymbol{v}(t)$ is a function of time. It means that the learned start and end VRP waypoint of the phase φ are not constant. In this way, the $\boldsymbol{v}_{l,i}(t)$ always stays on the linear interpolation between $\boldsymbol{v}_{l,i,\varphi}^{s}(t)$ and $\boldsymbol{v}_{l,i,\varphi}^{e}(t)$. However, the non-constant learned start and end VRP waypoint cause a discontinuity of the learned DCM reference trajectory $\boldsymbol{\xi}_{l,i}(t)$ and commanded VRP trajectory, which leads to VRP and DCM deviation and unstable walking behavior. Therefore, adjusting the current VRP reference trajectory by learning from the VRP error of the previous iteration is not an appropriate method.

Figure 3.2 illustrates a DCM-based walking simulated on a point-mass model. A constant force for $30N$ acts as an external disturbance on the CoM in the lateral direction consistently. Note that the deviation of the measured VRP $\boldsymbol{v}_m$ and the measured DCM $\boldsymbol{\xi}_m$ at the beginning of each transition phase become larger and larger, which means this learning strategy cannot successfully bring the VRP back to the originally desired trajectory, and the walking robustness is not enhanced.

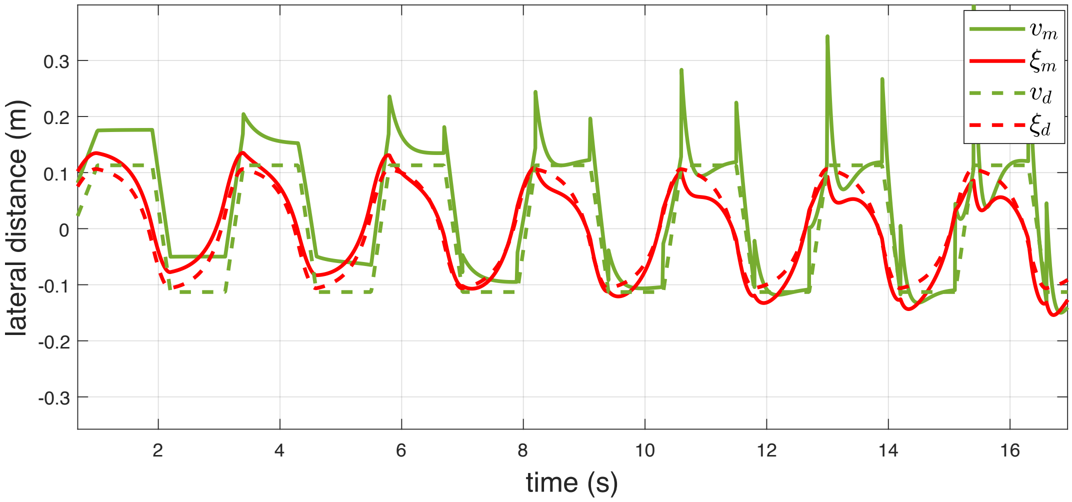


Figure 3.2: Lateral VRP and DCM deviation by ILC learning from the previous iteration.

Start Waypoint Learning for Future Iteration

Since the ILC learning from the VRP error of the previous iteration is not suitable for the DCM-based walking, the method is improved by learning from the VRP error of the current iteration to adjust the start waypoint of the corresponding transition phase of the next iteration. Specifically, the ILC approach is only applied at the beginning of the transition phase to learn the adjustment of the future transition phase's start VRP waypoint. Assume the current transition phase is φ, and the current iteration is i. The future VRP waypoint is adjusted as:

$$\boldsymbol{v}^s_{l,i+1,\varphi+4} = \boldsymbol{v}^s_{d,i+1,\varphi+4} + k_f \boldsymbol{R}_i^{i+1} \left(\boldsymbol{v}^s_{l,i,\varphi} - \boldsymbol{v}^s_{d,i,\varphi}\right) + k_l \boldsymbol{R}_i^{i+1} \left(\boldsymbol{v}^s_{d,i,\varphi} - \boldsymbol{v}^s_{m,i,\varphi}\right), \; i \geq 0, \tag{3.20}$$

where $\boldsymbol{v}^s_{l,i+1,\varphi+4}$ is the learned start VRP waypoint of the global transition phase $\varphi+4$ in the next iteration $i+1$, and $\boldsymbol{v}^s_{d,i+1,\varphi+4}$ represents the original reference start VRP waypoint. Here, the global transition phases $\varphi+4$ and φ present the same local transition phase in the future iteration $i+1$ and current iteration i respectively.

Figure 3.3 shows a simulation's performance of this learning design applied on the TORO robot in OpenHRP3 [KHK04]. When the waypoint learning method was not applied, the measured VRP trajectory, i.e., the solid green line in the zoomed-in figure, deviated from the desired trajectory. In comparison, our learning design brought the measured VRP trajectory (the red line) closer to the desired trajectory. The start and end of the measured VRP trajectory were already back to the desired trajectory. However, the middle of the trajectory cannot perfectly track the desired one. The reason for that is: we only focused on the learning for the start VRP waypoint but ignored the learning inside each transition phase. This motivated us to use a learned VRP waypoint list to save the whole adjusted future VRP trajectory and reuse it later, as described in Algorithm 1.

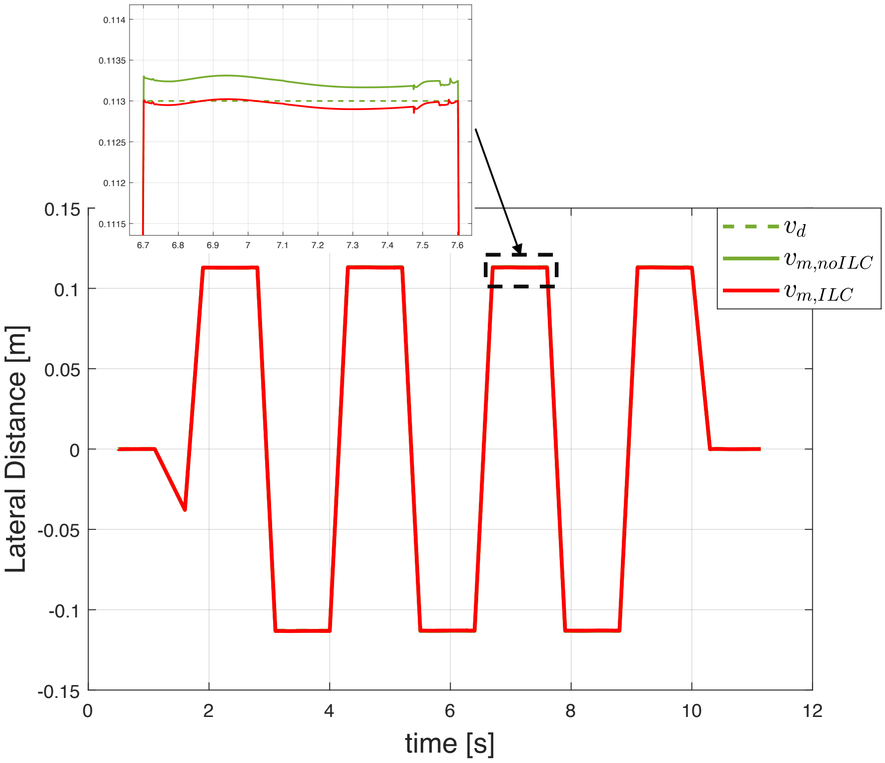


Figure 3.3: Lateral VRP trajectories by start waypoint learning for future iteration.

3.2 Stability Proof

To prove the stability of the VRP-OILC, we first simplify the model for the analysis by assuming there is only one VRP and DCM waypoint per iteration, as described in Section 3.2.1. Section 3.2.2 shows the analysis for the case of having two waypoints per iteration, and Section 3.2.3 introduces the general case. In each section, the stability of the VRP-OILC regarding both the measurement-error-based and commanded-error-based reference trajectories adaptation is analyzed, respectively.

3.2.1 Simple One Waypoint Case

Assume that there is only one waypoint in each iteration. This assumption aims to simplify the stability proof by reducing the waypoint number in each iteration. Since there is only one waypoint per iteration, the notation for the waypoint is also becoming simple. For instance, the single desired VRP waypoint at the i^{th} iteration is noted as $\boldsymbol{v}_{d,i}$. Moreover, the only waypoint is represented as the point at the iteration's local time step of $t = 0$. For example, the desired VRP waypoint $\boldsymbol{v}_{d,i}$ is equivalent to the $\boldsymbol{v}_{d,i}(0)$.

From now on, the i^{th} iteration is always considered as the current iteration ($i \geq 0$). So all the analysis is based on this consideration. Moreover, the robot's straightforward walking is first concerned. It means that there is not any turning behavior. To distinguish the duration T between two waypoints from the iteration duration T_{iter}, the relation between them is: $T_{iter} = n_w T$, with n_w is the waypoint number in each iteration. In this section, since there is only one waypoint per iteration, so $n_w = 1$. Section 3.2.2 and Section 3.2.3 have $n_w = 2$ and $n_w = \frac{T_{iter}}{\Delta t_{ILC}}$, respectively.

Measurement-Error-Based Adaptation Framework

Calculation of $\boldsymbol{v}_{l,i+1}$ For the stability proof, we start with the calculation of $\boldsymbol{v}_{l,i+1}$. For the calculation of the reference trajectories, the first-order polynomial $f_\varphi(t) = t/T_\varphi$ is used for the interpolation. As aforementioned in Algorithm 1, since we have the current learned VRP waypoint $\boldsymbol{v}_{l,i}$, the learned DCM is calculated as:

$$\boldsymbol{\xi}_{l,i} = \underbrace{\left(1 - \frac{b}{T} + \frac{b}{T} e^{-\frac{T}{b}}\right)}_{\alpha} \boldsymbol{v}_{l,i} + \underbrace{\left(\frac{b}{T} - \left(1 + \frac{b}{T}\right) e^{-\frac{T}{b}}\right)}_{\beta} \boldsymbol{v}_{d,i+1} + \underbrace{e^{-\frac{T}{b}}}_{\gamma} \boldsymbol{\xi}_{d,i+1}. \tag{3.21}$$

Then, the commanded VRP $\boldsymbol{v}_{c,i}$ at the current iteration is expressed as:

$$\boldsymbol{v}_{c,i} = \boldsymbol{v}_{l,i} + (1 + k_\xi b)\left(\boldsymbol{\xi}_i - \boldsymbol{\xi}_{l,i}\right). \tag{3.22}$$

The ILC updating law for $\boldsymbol{v}_{l,i+1}$ is:

$$\boldsymbol{v}_{l,i+1} = \boldsymbol{v}_{d,i+1} + k_f\left(\boldsymbol{v}_{l,i} - \boldsymbol{v}_{d,i}\right) + k_l\left(\boldsymbol{v}_{d,i} - \boldsymbol{v}_{m,i}\right). \tag{3.23}$$

According to the force error modeling in (41) of [EOAS15], when model inaccuracies and external forces exist, the total force error $\Delta \boldsymbol{F}(t)$ on the CoM can be modeled as:

$$\Delta \boldsymbol{F}(t) = \boldsymbol{F}_m(t) - \boldsymbol{F}_c(t), \tag{3.24}$$

where $\boldsymbol{F}_m(t)$ is the measured CoM total force and $\boldsymbol{F}_c(t)$ is the commanded force. As defined in (2.5), $\boldsymbol{F}_m(t)$ and $\boldsymbol{F}_c(t)$ can be computed by $\boldsymbol{F}_m(t) = \frac{m}{b^2}(\boldsymbol{x}(t) - \boldsymbol{v}_m(t))$ and $\boldsymbol{F}_c(t) = \frac{m}{b^2}(\boldsymbol{x}(t) - \boldsymbol{v}_c(t))$ respectively. Therefore, the VRP disturbance error $\boldsymbol{d}(t)$ in the i^{th} iteration can be modeled similar to (3.24) as:

$$\boldsymbol{d}(t) = \boldsymbol{v}_{m,i}(t) - \boldsymbol{v}_{c,i}(t). \tag{3.25}$$

Since there is only one waypoint in each iteration, (3.25) can be adapted to this case as:

$$\boldsymbol{d}(0) = \boldsymbol{v}_{m,i} - \boldsymbol{v}_{c,i}. \tag{3.26}$$

Inserting (3.21), (3.22) and (3.26) into the ILC updating law (3.23):

$$\begin{aligned}
\boldsymbol{v}_{l,i+1} &= \boldsymbol{v}_{d,i+1} + k_f\boldsymbol{v}_{l,i} - k_f\boldsymbol{v}_{d,i} + k_l\boldsymbol{v}_{d,i} - k_l\boldsymbol{v}_{l,i} - k_l\underbrace{(1 + k_\xi b)}_{\zeta}(\boldsymbol{\xi}_i - \boldsymbol{\xi}_{l,i}) - k_l\boldsymbol{d}(0)\\
&= (k_f - k_l)\boldsymbol{v}_{l,i} + \boldsymbol{v}_{d,i+1} + (k_l - k_f)\boldsymbol{v}_{d,i} - k_l\zeta\boldsymbol{\xi}_i + k_l\zeta\boldsymbol{\xi}_{l,i} - k_l\boldsymbol{d}(0)\\
&= [k_f + k_l(\zeta\alpha - 1)]\,\boldsymbol{v}_{l,i} + (-k_l\zeta)\boldsymbol{\xi}_i\\
&\quad + \underbrace{(k_l\zeta\beta + 1)\boldsymbol{v}_{d,i+1} + (k_l - k_f)\boldsymbol{v}_{d,i} + k_l\zeta\gamma\boldsymbol{\xi}_{d,i+1}}_{\text{Feedforward Term}\,(\boldsymbol{FFT}_1)} - k_l\boldsymbol{d}(0)\\
&= \underbrace{[k_f + k_l(\zeta\alpha - 1)]}_{a_4}\,\boldsymbol{v}_{l,i} + \underbrace{(-k_l\zeta)}_{a_3}\,\boldsymbol{\xi}_i + \boldsymbol{FFT}_1 - k_l\boldsymbol{d}(0),
\end{aligned} \tag{3.27}$$

where $\boldsymbol{FFT}_1$ denotes the sum of all the original desired terms. To simplify the equations in the following stability proof, the feedforward term's symbol $\boldsymbol{FFT}$ is always used to represent the original desired terms[3]. Note that there is a DCM state $\boldsymbol{\xi}_i$ in the equation (3.27). Basically, the idea for the stability proof is to treat the $\boldsymbol{v}_{l,i}$ and $\boldsymbol{\xi}_i$ as the system states, and subsequently build a state-space representation and analyze the stability of the representation. To build the system state-space representation, it is necessary to find out the relation between $\boldsymbol{\xi}_{i+1}$, $\boldsymbol{\xi}_i$ and $\boldsymbol{v}_{l,i}$.

Calculation of $\boldsymbol{\xi}_{i+1}$ There is a discontinuity of the $\boldsymbol{\xi}_{l,i}$ caused by adjusting the reference VRP point from the original desired VRP waypoint to the learned VRP waypoint, and also adjusting the reference DCM from the original desired DCM waypoint to the learned DCM waypoint after the future waypoints are updated using the ILC in the framework. More specifically, the desired VRP waypoint $\boldsymbol{v}_{d,i+1}$ and the desired DCM waypoint $\boldsymbol{\xi}_{d,i+1}$ from the equation (3.21) are adjusted to $\boldsymbol{v}_{l,i+1}$ and $\boldsymbol{\xi}_{l,i+1}$ respectively after the waypoints' update conducted by the VRP-OILC. The calculation of $\boldsymbol{\xi}_{l,i+1}$ is expressed by substituting $i + 1$ for i in (3.21):

$$\boldsymbol{\xi}_{l,i+1} = \alpha\boldsymbol{v}_{l,i+1} + \boldsymbol{FFT}_2. \tag{3.28}$$

So after the adjustment, the equation (3.21) becomes to the following equation based on the equation (3.27) and (3.28):

$$\begin{aligned}
\boldsymbol{\xi}^{\star}_{l,i} &= \alpha\boldsymbol{v}_{l,i} + \beta\boldsymbol{v}_{l,i+1} + \gamma\boldsymbol{\xi}_{l,i+1}\\
&= \alpha\boldsymbol{v}_{l,i} + (\beta + \gamma\alpha)\,\boldsymbol{v}_{l,i+1} + \boldsymbol{FFT}_3\\
&= \alpha\boldsymbol{v}_{l,i} + (\beta + \gamma\alpha)\,a_4\boldsymbol{v}_{l,i} + (\beta + \gamma\alpha)\,a_3\boldsymbol{\xi}_i - k_l\,(\beta + \gamma\alpha)\,\boldsymbol{d}(0) + \boldsymbol{FFT}_4\\
&= [\alpha + (\beta + \gamma\alpha)\,a_4]\,\boldsymbol{v}_{l,i} + (\beta + \gamma\alpha)\,a_3\boldsymbol{\xi}_i - k_l\,(\beta + \gamma\alpha)\,\boldsymbol{d}(0) + \boldsymbol{FFT}_4,
\end{aligned} \tag{3.29}$$

[3] E.g., $\boldsymbol{v}_{d,i+1}$ and $\boldsymbol{\xi}_{d,i+1}$. Basically all the terms with a subscript "d"

where $\boldsymbol{\xi}^{\star}_{l,i}$ represents the learned DCM waypoint at the i^{th} iteration after the adjustment.

To find the relation between $\boldsymbol{\xi}_{i+1}$ and $\boldsymbol{\xi}_i$, it is vital to start from the DCM dynamics. The continue-time DCM dynamics in i^{th} iteration is expanded by inserting the equations (3.22) and (3.25) as:

$$\begin{aligned}
\dot{\boldsymbol{\xi}}_i(t) &= \frac{1}{b}\left(\boldsymbol{\xi}_i(t) - \boldsymbol{v}_{m,i}(t)\right) \\
&= \frac{1}{b}\left(\boldsymbol{\xi}_i(t) - \boldsymbol{v}_{c,i}(t) - \boldsymbol{d}(t)\right) \\
&= \frac{1}{b}\left(\boldsymbol{\xi}_i(t) - \boldsymbol{v}_{l,i}(t) - (1 + k_\xi b)\left(\boldsymbol{\xi}_i(t) - \boldsymbol{\xi}_{l,i}(t)\right) - \boldsymbol{d}(t)\right) \\
&= \frac{1}{b}\left(\cancel{\boldsymbol{\xi}_i(t)} - \cancel{\boldsymbol{\xi}_i(t)} + \underbrace{\boldsymbol{\xi}_{l,i}(t) - \boldsymbol{v}_{l,i}(t)}_{=b\,\dot{\boldsymbol{\xi}}_{l,i}(t)} - k_\xi b\left(\boldsymbol{\xi}_i(t) - \boldsymbol{\xi}_{l,i}(t)\right) - \boldsymbol{d}(t)\right) \\
&= \dot{\boldsymbol{\xi}}_{l,i}(t) - k_\xi\left(\boldsymbol{\xi}_i(t) - \boldsymbol{\xi}_{l,i}(t)\right) - \frac{\boldsymbol{d}(t)}{b}.
\end{aligned} \tag{3.30}$$

Moving the $\dot{\boldsymbol{\xi}}_{l,i}(t)$ to the left side of the equation (3.30), a DCM closed-loop dynamics with the disturbance term can be obtained as:

$$\underbrace{\dot{\boldsymbol{\xi}}_i(t) - \dot{\boldsymbol{\xi}}_{l,i}(t)}_{\dot{\boldsymbol{\epsilon}}_i(t)} = -k_\xi \underbrace{\left(\boldsymbol{\xi}_i(t) - \boldsymbol{\xi}_{l,i}(t)\right)}_{\boldsymbol{\epsilon}_i(t)} - \frac{\boldsymbol{d}(t)}{b}, \tag{3.31}$$

which can derive the DCM error $\boldsymbol{\epsilon}_i(t)$ as:

$$\begin{aligned}
\underbrace{\boldsymbol{\xi}_i(t) - \boldsymbol{\xi}_{l,i}(t)}_{\boldsymbol{\epsilon}_i(t)} &= e^{-k_\xi t}\left(\underbrace{\boldsymbol{\xi}_i(0) - \boldsymbol{\xi}^{\star}_{l,i}(0)}_{\boldsymbol{\epsilon}_i(0)} - \frac{1}{b}\int_0^t e^{k_\xi \tau}\boldsymbol{d}(\tau)d\tau\right) \\
&= e^{-k_\xi t}\cdot\boldsymbol{\xi}_i - e^{-k_\xi t}\cdot\boldsymbol{\xi}^{\star}_{l,i} - \frac{1}{b}\underbrace{e^{-k_\xi t}\int_0^t e^{k_\xi \tau}\boldsymbol{d}(\tau)d\tau}_{\boldsymbol{D}(t)},
\end{aligned} \tag{3.32}$$

where $\boldsymbol{\xi}_i(0) = \boldsymbol{\xi}_i$ and $\boldsymbol{\xi}^{\star}_{l,i}(0) = \boldsymbol{\xi}^{\star}_{l,i}$ because the only waypoint in each iteration is equivalent to the point at the beginning of the iteration, as mentioned in the beginning of this section. Inserting (3.27), (3.28) and (3.29) into (3.32), the relation between $\boldsymbol{\xi}_{i+1}$, $\boldsymbol{\xi}_i$ and $\boldsymbol{v}_{l,i}$ can be found as:

$$\begin{aligned}
\boldsymbol{\xi}_{i+1} = \boldsymbol{\xi}_i(T) &= \boldsymbol{\xi}_{l,i}(T) + e^{-k_\xi T}\cdot\boldsymbol{\xi}_i - e^{-k_\xi T}\cdot\boldsymbol{\xi}^{\star}_{l,i} - \frac{\boldsymbol{D}(T)}{b} \\
&= \boldsymbol{\xi}_{l,i+1} + e^{-k_\xi T}\cdot\boldsymbol{\xi}_i - e^{-k_\xi T}\cdot\boldsymbol{\xi}^{\star}_{l,i} - \frac{\boldsymbol{D}(T)}{b} \\
&= \underbrace{\left[\alpha a_4 - e^{-k_\xi T}\left[\alpha + a_4\left(\beta+\gamma\alpha\right)\right]\right]}_{a_2}\boldsymbol{v}_{l,i} \\
&\quad + \underbrace{\left[e^{-k_\xi T} + \alpha a_3 + e^{-k_\xi T}k_l\zeta\left(\beta+\gamma\alpha\right)\right]}_{a_1}\boldsymbol{\xi}_i \\
&\quad + \underbrace{\left[e^{-k_\xi T}k_l(\beta+\gamma\alpha) - \alpha k_l\right]}_{a_5}\boldsymbol{d}(0) - \frac{\boldsymbol{D}(T)}{b} + \boldsymbol{FFT}_5.
\end{aligned} \tag{3.33}$$

Stability Analysis of State-Space Representation According to the (3.27) and (3.33), $\left[\boldsymbol{\xi}_i^T, \boldsymbol{v}_{l,i}^T\right]^T$ is considered a system state vector. So the system state-space representation can be built as:

$$\begin{bmatrix}\boldsymbol{\xi}_{i+1}\\ \boldsymbol{v}_{l,i+1}\end{bmatrix} = \underbrace{\begin{bmatrix}a_1 & a_2\\ a_3 & a_4\end{bmatrix}}_{A}\begin{bmatrix}\boldsymbol{\xi}_i\\ \boldsymbol{v}_{l,i}\end{bmatrix} + \begin{bmatrix}a_5\boldsymbol{d}(0) - \frac{\boldsymbol{D}(T)}{b}\\ -k_l\boldsymbol{d}(0)\end{bmatrix} + \begin{bmatrix}\boldsymbol{FFT}_5\\ \boldsymbol{FFT}_1\end{bmatrix}. \tag{3.34}$$

Because all the terms in the state-space representation (3.34) are not functions of time, this state-space representation is linear. For a linear system, it is possible to prove the system stability by analyzing the eigenvalues of matrix A. Since (3.34) is a discrete-time model, the system is stable when all the absolute eigenvalues of A are smaller than 1.

The entries a_1 - a_4 are composed of five essential parameters: the ILC forgetting factor k_f, the ILC learning gain k_l, the duration T between two waypoints, the time constant b, and the DCM controller gain k_ξ. In the robot system, the b and k_ξ are not changed any more since they have been set to optimal values. Here, $b \approx 0.3112$ and $k_\xi = 4$ are used not only for the eigenvalue analysis but also for the simulation and experiment. By this, only the impact of the k_f, k_l, and T on the eigenvalues need to be analyzed.

In this section, the waypoint duration T is always equal to the iteration duration T_{iter} because there is only one waypoint per iteration. For the eigenvalue analysis, we start from the following two cases of the waypoint duration T:

1) $e^{-k_\xi T} \approx 0$: The entry a_1 and a_2 in the matrix A include a term "$e^{-k_\xi T}$". For normal speed walking during the simulations and experiments, a SS time for 0.9 s and a DS time for 0.3 s are always used. In this case, the waypoint

duration is $T = 2.4$ s, which causes $e^{-k_\xi T} \approx 0$. For fast walking after applying the VRP-OILC framework, the shortest SS and DS time we tried is 0.7 s and 0.2 s respectively, which means the $T = 1.8$ s. Still, this shortest T also leads to $e^{-k_\xi T} \approx 0$. It guarantees the $e^{-k_\xi T}$ is always approximately equal to 0 for all the simulations and experiments we conducted. In this case, the matrix A can be expressed as:

$$A \approx A' = \begin{bmatrix} \alpha a_3 & \alpha a_4 \\ a_3 & a_4 \end{bmatrix}, \tag{3.35}$$

where matrix A' sets the terms $e^{-k_\xi T}$ to be 0 for its entries. To calculate the eigenvalues, the determinant of $A' - \lambda \boldsymbol{I}$ is first calculated:

$$\begin{aligned} |A' - \lambda \boldsymbol{I}| &= \begin{vmatrix} \alpha a_3 - \lambda & \alpha a_4 \\ a_3 & a_4 - \lambda \end{vmatrix} \\ &= (\alpha a_3 - \lambda)(a_4 - \lambda) - \alpha a_3 a_4 \\ &= \lambda^2 - (\alpha a_3 + a_4)\lambda \\ &= \lambda^2 - (k_f - k_l)\lambda \\ &= 0 \end{aligned} \tag{3.36}$$

From (3.36), the eigenvalues of the matrix A' can be solved as:

$$\lambda_1(A') = k_f - k_l; \quad \lambda_2(A') = 0. \tag{3.37}$$

The eigenvalues of the matrix A are:

$$\lambda_1(A) \approx \lambda_1(A') = k_f - k_l; \quad \lambda_2(A) \approx \lambda_2(A') = 0. \tag{3.38}$$

Under the condition of $e^{-k_\xi T} \approx 0$, one of the eigenvalues of the matrix A is only relative to the k_l and k_f, where the other is close to 0 all the time. Therefore, the stability condition of the system (3.34) becomes as follows:

$$|k_f - k_l| < 1. \tag{3.39}$$

2) $e^{-k_\xi T} \not\approx 0$: The SS time and DS time used in the simulations and experiments ensure the condition $e^{-k_\xi T} \approx 0$. Here, the relation between the T and the eigenvalues with $e^{-k_\xi T} \not\approx 0$ is investigated. To analyze the eigenvalues numerically, the waypoint duration T varies from 0.01 s to 2.4 s, and the k_l and k_f are set to the combinations of $\{0.1, 0.5, 1.0\}$. These combinations ensure $|k_f - k_l| < 1$. Figure 3.4 illustrates the relation between the eigenvalues and the T for different k_l and k_f. The subfigures in the same row show the relations under the same k_l value. Likewise, the subfigures in the same column have the same k_f value. In each single subfigure, the X-axis represents the duration T, where the Y-axis denotes the absolute eigenvalue $|\lambda_i|$. From Figure 3.4, three important observations can be obtained: i) *one of the absolute eigenvalue becomes greater than* 1 *for* $k_f = 1$ *and* $T < 0.2$ s. Except for this k_f value and

duration range, the absolute eigenvalues are always less than 1, which ensures the stability condition for the system. Basically, $T < 0.2$ s is an extremely short duration and unrealistic for a real robot. As known so far, there is not any bipedal robot that can walk with such a short duration (i.e., the sum of the SS time and DS time is less than 0.1 s). As aforementioned, the shortest iteration duration T_{iter} applied in the DLR humanoid robot TORO is 1.8 s (i.e., the waypoint duration $T = 1.8$ s for one waypoint case). It concludes that in the robot system (i.e., $T_{iter} \geq 1.8$ s), the phenomenon with an eigenvalue greater than 1 as mentioned before does not occur; ii) *As the waypoint duration T increases, one absolute eigenvalue converges to* 0*, where the other one converges to* $|k_f - k_l|$. The reason for the convergence is that when T is increasing, the $e^{-k_\xi T}$, where $k_\xi = 4$, is then approaching 0. As analyzed in (3.38), when $e^{-k_\xi T} \rightarrow 0$, $|\lambda_1(A)|$ converges to $|k_f - k_l|$, and $|\lambda_2(A)|$ converges to 0. Figure 3.4 shows that both absolute eigenvalues are already converged when T is equal to or greater than 1.2 s (i.e., equivalent to $T_{iter} \geq 1.2$ s). We conclude that when $T \geq 1.2$ s (or $T_{iter} \geq 1.2$ s), the eigenvalue analysis can be simplified by analyzing the $|k_f - k_l| < 1$; iii) *In the range of* $0.2\ s \leq T < 1.2\ s$*, although the absolute eigenvalues are not converged and need to be determined by numerical analysis, they are still smaller than* 1 *with the pre-condition of* $|k_f - k_l| < 1$. This concludes that when $|k_f - k_l| < 1$ and $0.2\ s \leq T < 1.2\ s$, the system is still stable, but the eigenvalues need to be calculated numerically for the analysis.

After analyzing the case of $e^{-k_\xi T} \approx 0$ and $e^{-k_\xi T} \not\approx 0$, we can conclude that the system is stable for $T > 0.2$ s (i.e., $T_{iter} > 0.2$ s) and $|k_f - k_l| < 1$. The stability analysis can particularly be simplified to only focus on the condition $|k_f - k_l| < 1$ because one of the absolute eigenvalues converges to $|k_f - k_l|$ when $T \geq 1.2$ s. Since the robot system cannot achieve an iteration duration T_{iter} less than 1.8 s, the $|k_f - k_l| < 1$ becomes the only condition that needs to be considered for stability.

As mentioned in the beginning, the stability analysis so far is based on the robot's straight walking. For a walking with turning, the ILC updating law (3.23) becomes:

$$\boldsymbol{v}_{l,i+1} = \boldsymbol{v}_{d,i+1} + k_f \boldsymbol{R}_\Delta \left(\boldsymbol{v}_{l,i} - \boldsymbol{v}_{d,i}\right) + k_l \boldsymbol{R}_\Delta \left(\boldsymbol{v}_{d,i} - \boldsymbol{v}_{m,i}\right). \tag{3.40}$$

As known, the absolute eigenvalues of the rotation matrix $|\lambda(\boldsymbol{R}_\Delta)|$ are always equal to 1. It means that the rotation matrix $\boldsymbol{R}_\Delta$ does not affect the stability proof until now. The turning behavior is ignored to simplify all the analysis below.

Commanded-Error-Based Adaptation Framework

As aforementioned, the turning behavior does not affect the stability analysis. So the straight walking is also used for the analysis of the commanded-error-based adaptation. Here, the ILC updating law becomes:

$$\boldsymbol{v}_{l,i+1} = \boldsymbol{v}_{d,i+1} + k_f \left(\boldsymbol{v}_{l,i} - \boldsymbol{v}_{d,i}\right) + k_l \left(\boldsymbol{v}_{d,i} - \boldsymbol{v}_{c,i}\right). \tag{3.41}$$

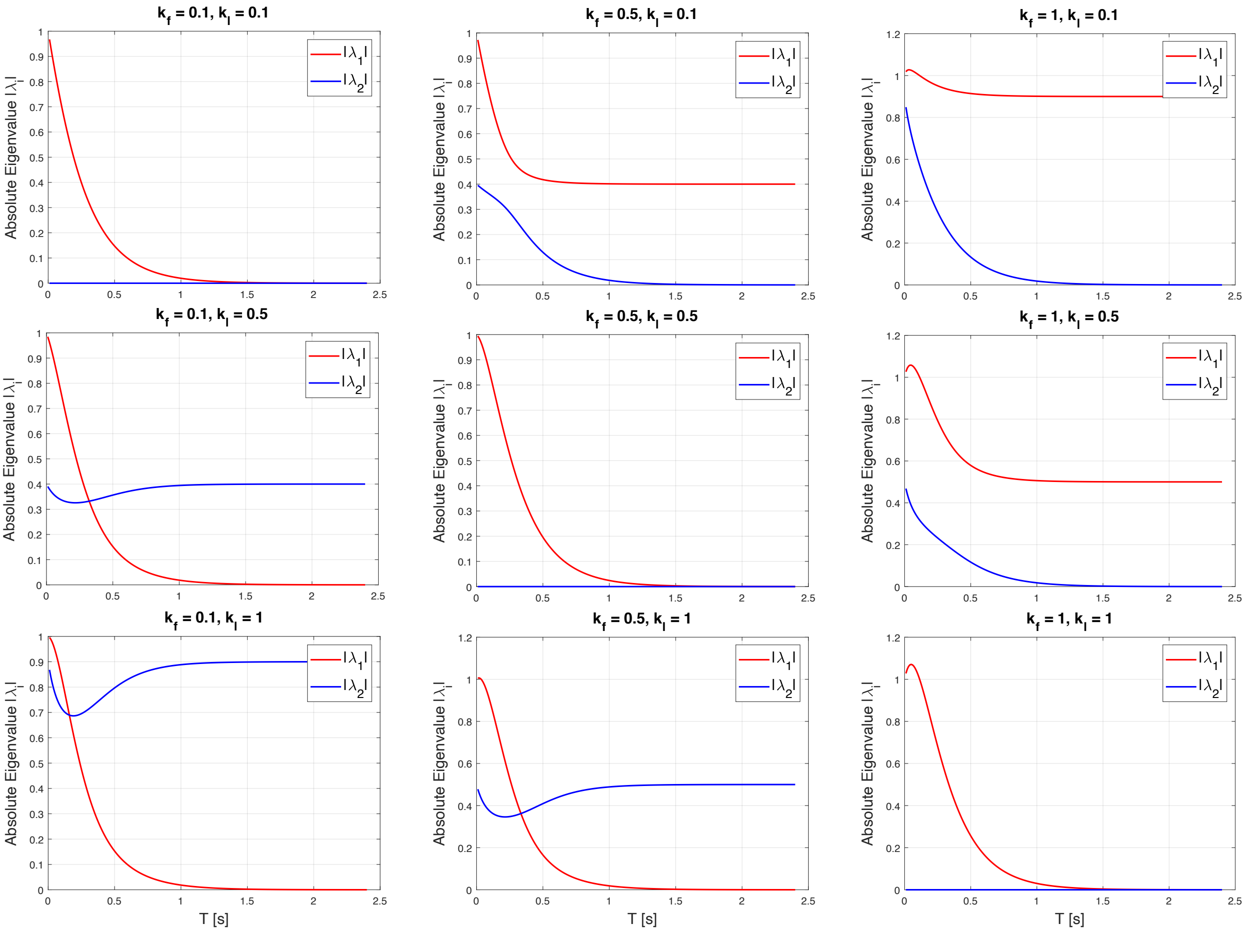


Figure 3.4: Relation between the eigenvalues and the waypoint duration T under different k_f and k_l (one waypoint case).

Note that (3.41) is the same as (3.27) without the disturbance term $\boldsymbol{d}(0)$. Therefore, (3.41) can be expanded from (3.27) to:

$$\boldsymbol{v}_{l,i+1} = a_4 \boldsymbol{v}_{l,i} + a_3 \boldsymbol{\xi}_i + \boldsymbol{FFT}_1. \tag{3.42}$$

Consequently, the relation between $\boldsymbol{\xi}_{i+1}$, $\boldsymbol{\xi}_i$ and $\boldsymbol{v}_{l,i}$ can be obtained by following the calculation ideas of (3.28) - (3.34) as:

$$\boldsymbol{\xi}_{i+1} = a_2 \boldsymbol{v}_{l,i} + a_1 \boldsymbol{\xi}_i - \frac{\boldsymbol{D}(T)}{b} + \boldsymbol{FFT}_5. \tag{3.43}$$

The final state space representation is built from (3.42) and (3.43) as:

$$\begin{bmatrix} \boldsymbol{\xi}_{i+1} \\ \boldsymbol{v}_{l,i+1} \end{bmatrix} = \underbrace{\begin{bmatrix} a_1 & a_2 \\ a_3 & a_4 \end{bmatrix}}_{A} \begin{bmatrix} \boldsymbol{\xi}_i \\ \boldsymbol{v}_{l,i} \end{bmatrix} + \begin{bmatrix} -\frac{\boldsymbol{D}(T)}{b} \\ \boldsymbol{0} \end{bmatrix} + \begin{bmatrix} \boldsymbol{FFT}_5 \\ \boldsymbol{FFT}_1 \end{bmatrix}. \tag{3.44}$$

Since the A matrix of (3.44) is exactly the same as the one in (3.34), the stability analysis of the measurement-error-based adaptation framework is also applicable to the commanded-error-based adaptation.

3.2.2 Simple Two Waypoints Case

This section analyzes the stability of a more complex system: two waypoints system. Since there are two waypoints per iteration, the superscript "1" and "2" are used to denote the first and second waypoints. For instance, the first learned VRP waypoint of the i^{th} iteration is $\boldsymbol{v}_{l,i}^1$, and the second one is $\boldsymbol{v}_{l,i}^2$ (note that $\boldsymbol{v}_{l,i}^2$ is not a square, i.e., $\boldsymbol{v}_{l,i}^2 \neq \boldsymbol{v}_{l,i} \cdot \boldsymbol{v}_{l,i}$). These two waypoints are the discrete points at the moment of the iteration's local time $t = 0$ and $t = \frac{T_{iter}}{2}$ of the corresponding trajectory. For example, $\boldsymbol{v}_{l,i}^1 = \boldsymbol{v}_{l,i}(0)$ and $\boldsymbol{v}_{l,i}^2 = \boldsymbol{v}_{l,i}(\frac{T_{iter}}{2})$ are presented.

Assume that the i^{th} iteration is the current iteration ($i \geq 0$). As known, the waypoint number in each iteration is $n_w = 2$. So the waypoint duration is: $T = \frac{T_{iter}}{n_w} = \frac{T_{iter}}{2}$.

Measurement-Error-Based Adaptation Framework

Calculation of $\boldsymbol{v}_{l,i+1}^1$ To calculate the first VRP learned waypoint $\boldsymbol{v}_{l,i+1}^1$ in the next iteration, we start from the calculation of the second DCM learned waypoint $\boldsymbol{\xi}_{l,i}^2$ at the current iteration as:

$$\boldsymbol{\xi}_{l,i}^2 = \alpha \boldsymbol{v}_{l,i}^2 + \beta \boldsymbol{v}_{d,i+1}^1 + \gamma \boldsymbol{\xi}_{d,i+1}^1, \tag{3.45}$$

where the first-order polynomial is applied for the linear interpolation. Here, the $\boldsymbol{\xi}_{l,i}^2$ represents the learned DCM waypoint before the future VRP waypoint is updated. The first DCM learned waypoint can be derived from (3.45) as:

$$\begin{aligned} \boldsymbol{\xi}_{l,i}^1 &= \alpha \boldsymbol{v}_{l,i}^1 + \beta \boldsymbol{v}_{l,i}^2 + \gamma \boldsymbol{\xi}_{l,i}^2 \\ &= \alpha \boldsymbol{v}_{l,i}^1 + (\beta + \alpha\gamma) \boldsymbol{v}_{l,i}^2 + \boldsymbol{FFT}_6. \end{aligned} \tag{3.46}$$

The first commanded VRP waypoint $\boldsymbol{v}_{c,i}^1$ is expressed as:

$$\boldsymbol{v}_{c,i}^1 = \boldsymbol{v}_{l,i}^1 + \zeta \left(\boldsymbol{\xi}_i^1 - \boldsymbol{\xi}_{l,i}^1 \right). \tag{3.47}$$

The disturbance error of the first and second VRP waypoint is modeled respectively as:

$$\boldsymbol{d}(0) = \boldsymbol{v}_{m,i}^1 - \boldsymbol{v}_{c,i}^1. \tag{3.48}$$

$$\boldsymbol{d}(T) = \boldsymbol{v}_{m,i}^2 - \boldsymbol{v}_{c,i}^2. \tag{3.49}$$

The ILC updating law can be expressed and expanded by inserting the equations (3.46) - (3.48) as:

$$\begin{aligned} \boldsymbol{v}^1_{l,i+1} &= \boldsymbol{v}^1_{d,i+1} + k_f \boldsymbol{v}^1_{l,i} - k_f \boldsymbol{v}^1_{d,i} + k_l \boldsymbol{v}^1_{d,i} - k_l \boldsymbol{v}^1_{l,i} - k_l \zeta \left(\boldsymbol{\xi}^1_i - \boldsymbol{\xi}^1_{l,i} \right) - k_l \boldsymbol{d}(0) \\ &= \underbrace{\left[k_f + k_l(\zeta\alpha - 1) \right]}_{b_5} \boldsymbol{v}^1_{l,i} + \underbrace{k_l \zeta(\beta + \alpha\gamma)}_{b_6} \boldsymbol{v}^2_{l,i} + \underbrace{(-k_l\zeta)}_{b_4} \boldsymbol{\xi}^1_i + \boldsymbol{FFT}_7 - k_l \boldsymbol{d}(0). \end{aligned} \tag{3.50}$$

Calculation of $\boldsymbol{v}^2_{l,i+1}$ For the learning of the second VRP reference waypoint in the next iteration, we start from calculating the first DCM learned waypoint in the $(i+1)^{th}$ iteration:

$$\boldsymbol{\xi}^1_{l,i+1} = \alpha \boldsymbol{v}^1_{l,i+1} + \boldsymbol{FFT}_8. \tag{3.51}$$

As aforementioned, the second DCM learned waypoint $\boldsymbol{\xi}^2_{l,i}$ at the current iteration has a discontinuity behavior attributed to the future reference VRP waypoint's update. Here, the first VRP reference VRP waypoint in the next iteration $\boldsymbol{v}^1_{d,i+1}$ is updated to $\boldsymbol{v}^1_{l,i+1}$, so the $\boldsymbol{\xi}^2_{l,i}$ becomes to $\boldsymbol{\xi}^{2\star}_{l,i}$, shown as:

$$\boldsymbol{\xi}^{2\star}_{l,i} = \alpha \boldsymbol{v}^2_{l,i} + \beta \boldsymbol{v}^1_{l,i+1} + \gamma \boldsymbol{\xi}^1_{l,i+1}. \tag{3.52}$$

And the current iteration's first DCM learned waypoint is calculated after the future VRP waypoint's update as:

$$\boldsymbol{\xi}^{1\star}_{l,i} = \alpha \boldsymbol{v}^1_{l,i} + \beta \boldsymbol{v}^2_{l,i} + \gamma \boldsymbol{\xi}^{2\star}_{l,i}. \tag{3.53}$$

According to (3.32), the second measured DCM waypoint $\boldsymbol{\xi}^2_i$ can be derived from the first measured waypoint $\boldsymbol{\xi}^1_i$ as:

$$\boldsymbol{\xi}^2_i = \boldsymbol{\xi}_i(T) = \boldsymbol{\xi}^{2\star}_{l,i} + e^{-k_\xi T} \cdot \boldsymbol{\xi}^1_i - e^{-k_\xi T} \cdot \boldsymbol{\xi}^{1\star}_{l,i} - \frac{\boldsymbol{D}(T)}{b}. \tag{3.54}$$

The second commanded VRP waypoint $\boldsymbol{v}^2_{c,i}$ is calculated as:

$$\boldsymbol{v}^2_{c,i} = \boldsymbol{v}^2_{l,i} + \zeta \left(\boldsymbol{\xi}^2_i - \boldsymbol{\xi}^{2\star}_{l,i} \right). \tag{3.55}$$

The ILC updating law of the $\boldsymbol{v}^2_{l,i+1}$ can be expanded by using the error model (3.49) and the equations (3.50) - (3.55) as:

$$\begin{aligned} \boldsymbol{v}^2_{l,i+1} = b_8 \boldsymbol{v}^1_{l,i} + b_9 \boldsymbol{v}^2_{l,i} + b_7 \boldsymbol{\xi}^1_i + \underbrace{\left[- \left[\zeta k_l^2 e^{-k_\xi T} (\beta + \alpha\gamma)\gamma \right] \right]}_{b_{13}} \boldsymbol{d}(0) \\ - k_l \boldsymbol{d}(T) + \frac{k_l \zeta}{b} \boldsymbol{D}(T) + \boldsymbol{FFT}_9, \end{aligned} \tag{3.56}$$

where the parameters b_7 - b_9 are not expanded here because they are too long. With the *Symbolic Math Toolbox* of *Matlab* [TM19], these parameters can be analyzed in symbolic form for stability proof.

Calculation of $\boldsymbol{\xi}^1_{i+1}$ To obtain the measured DCM waypoint $\boldsymbol{\xi}^1_{i+1}$ of the next iteration, we start from the calculation of $\boldsymbol{\xi}^2_{l,i+1}$:

$$\boldsymbol{\xi}^2_{l,i+1} = \alpha \boldsymbol{v}^2_{l,i+1} + \boldsymbol{FFT}_{10}. \tag{3.57}$$

Since the second VRP reference waypoint in the next iteration is updated to $\boldsymbol{v}^2_{l,i+1}$, the fisrt DCM learned waypoint $\boldsymbol{\xi}^1_{l,i+1}$ in the next iteration is changed to $\boldsymbol{\xi}^{1\star}_{l,i+1}$ as:

$$\boldsymbol{\xi}^{1\star}_{l,i+1} = \alpha \boldsymbol{v}^1_{l,i+1} + \beta \boldsymbol{v}^2_{l,i+1} + \gamma \boldsymbol{\xi}^2_{l,i+1}. \tag{3.58}$$

The second DCM learned waypoint $\boldsymbol{\xi}^{2\star}_{l,i}$, which is the discontinue waypoint after the first future VRP reference waypoint is updated to $\boldsymbol{v}^1_{l,i+1}$, is changed again to $\boldsymbol{\xi}^{2\star\star}_{l,i}$ when the second VRP reference waypoint of the $(i+1)^{th}$ iteration $\boldsymbol{v}^2_{d,i+1}$ is updated to $\boldsymbol{v}^2_{l,i+1}$. The $\boldsymbol{\xi}^{2\star\star}_{l,i}$ is therefore expressed as:

$$\boldsymbol{\xi}^{2\star\star}_{l,i} = \alpha \boldsymbol{v}^2_{l,i} + \beta \boldsymbol{v}^1_{l,i+1} + \gamma \boldsymbol{\xi}^{1\star}_{l,i+1}. \tag{3.59}$$

Based on (3.32), (3.54), and (3.56) - (3.59), the first measured DCM $\boldsymbol{\xi}^1_{i+1}$ of the next iteration is calculated as:

$$\begin{aligned}
\boldsymbol{\xi}^1_{i+1} = \boldsymbol{\xi}_i(2T) &= \boldsymbol{\xi}^{1\star}_{l,i+1} + e^{-k_\xi T} \cdot \boldsymbol{\xi}^2_i - e^{-k_\xi T} \cdot \boldsymbol{\xi}^{2\star\star}_{l,i} - \frac{1}{b}\left[\boldsymbol{D}(2T) - \boldsymbol{D}(T)\right] \\
&= b_2 \boldsymbol{v}^1_{l,i} + b_3 \boldsymbol{v}^2_{l,i} + b_1 \boldsymbol{\xi}^1_i + b_{10} \boldsymbol{D}(T) - \frac{1}{b}\boldsymbol{D}(2T) \\
&\quad + b_{11}\boldsymbol{d}(0) + b_{12}\boldsymbol{d}(T) + \boldsymbol{FFT}_{11},
\end{aligned} \tag{3.60}$$

where the parameters b_1 - b_3 and b_{10} - b_{12} are not expanded because of their length and complexity.

It is redundant to investigate the second measured DCM waypoint $\boldsymbol{\xi}^2_{i+1}$ because it can be derived from the first waypoint $\boldsymbol{\xi}^1_{i+1}$ by following the similar idea of (3.54). For the measured DCM waypoint, the equation of $\boldsymbol{\xi}^1_{i+1}$ as shown in (3.60) is already sufficient for the stability analysis.

Stability Analysis of State-Space Representation From (3.50), (3.56) and (3.60), a state-space representation for the two waypoint case is built as:

$$\begin{aligned}
\begin{bmatrix} \boldsymbol{\xi}^1_{i+1} \\ \boldsymbol{v}^1_{l,i+1} \\ \boldsymbol{v}^2_{l,i+1} \end{bmatrix} = \underbrace{\begin{bmatrix} b_1 & b_2 & b_3 \\ b_4 & b_5 & b_6 \\ b_7 & b_8 & b_9 \end{bmatrix}}_{B} \begin{bmatrix} \boldsymbol{\xi}^1_i \\ \boldsymbol{v}^1_{l,i} \\ \boldsymbol{v}^2_{l,i} \end{bmatrix} &+ \begin{bmatrix} b_{10}\boldsymbol{D}(T) - \frac{1}{b}\boldsymbol{D}(2T) + b_{11}\boldsymbol{d}(0) + b_{12}\boldsymbol{d}(T) \\ -k_l \boldsymbol{d}(0) \\ \frac{k_l \zeta}{b}\boldsymbol{D}(T) + b_{13}\boldsymbol{d}(0) - k_l \boldsymbol{d}(T) \end{bmatrix} \\
&+ \begin{bmatrix} \boldsymbol{FFT}_{11} \\ \boldsymbol{FFT}_7 \\ \boldsymbol{FFT}_9 \end{bmatrix}.
\end{aligned} \tag{3.61}$$

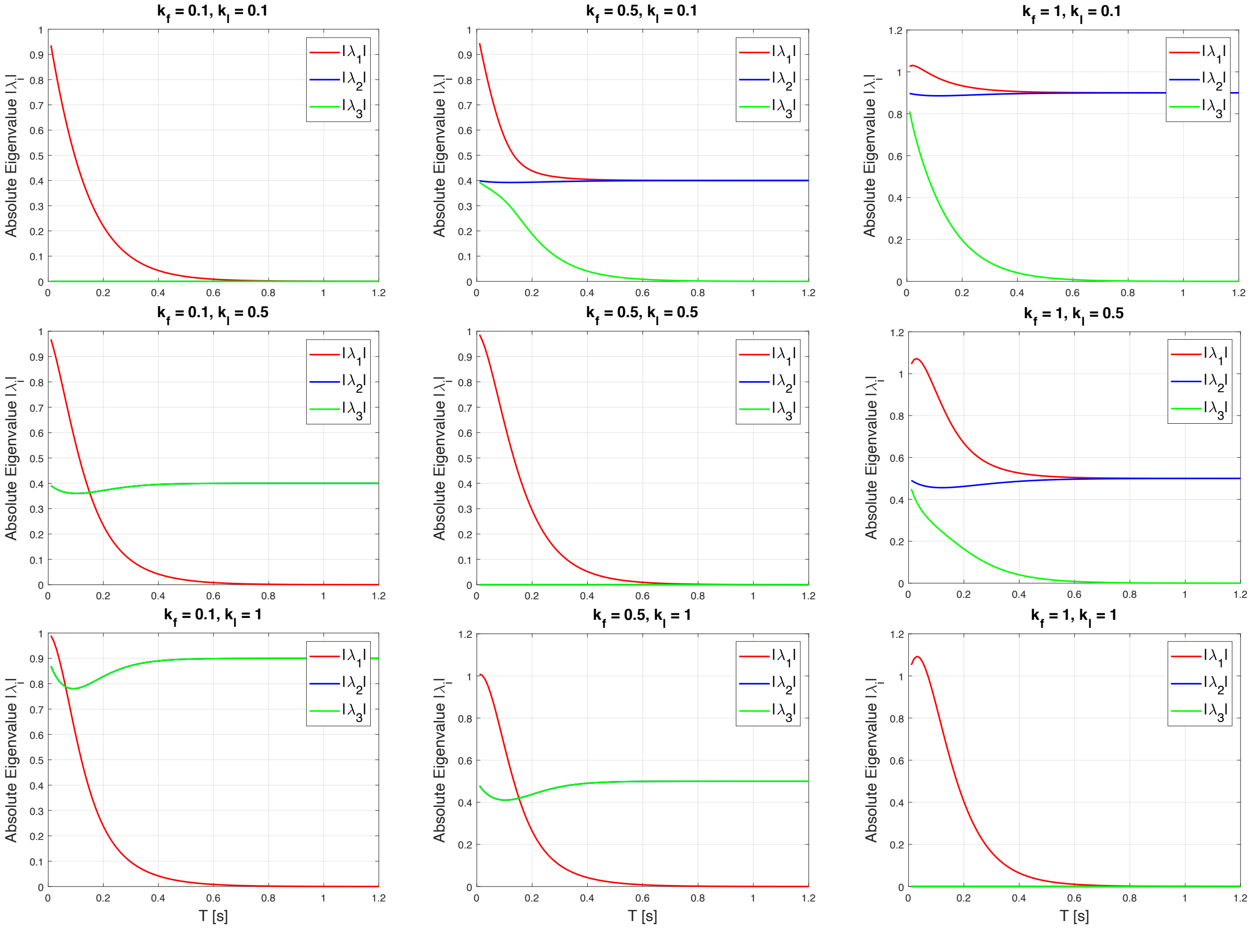


Figure 3.5: Relation between the eigenvalues and the waypoint duration T under different k_f and k_l (two waypoints case).

Here, the *Symbolic Math Toolbox* of *Matlab* is used to analyze the eigenvalues of the matrix B. Specifically, k_l and k_f are set to the values selected from the different combinations of $\{0.1, 0.5, 1.0\}$, set $b \approx 0.3112$ and $k_\xi = 4$ as aforementioned, and then change the waypoint duration T from 0.01 s to 1.2 s with the interval 0.01 s (i.e., the iteration duration T_{iter} is changing from 0.02 s to 2.4 s) to analyze the eigenvalues' responses. Figure 3.5 shows the absolute eigenvalues' plots response to the changing of T. Here, the same conclusions are obtained as concluding from the Figure 3.4: i) when waypoint duration T is greater than 0.1 s (i.e., the iteration duration T_{iter} is greater than 0.2 s) under the condition of $|k_f - k_l| < 1$, all the three absolute eigenvalues are smaller than 1; ii) when $T_{iter} \geq 1.2$ s, one of the absolute eigenvalue converges to 0, where the others converge to $|k_f - k_l|$. Because the T_{iter} in the system is not less than 1.8 s, this assures the system is always stable for $|k_f - k_l| < 1$.

Inspired by the root locus analysis method, a maximum eigenvalue locus is drawn as an auxiliary analysis method in the complex plane. Figure 3.6 illustrates an example of the maximum eigenvalue locus with a varying k_l. In particular, the k_f is fixed to 1, the SS time T_{SS} is set to 0.9 s and the DS time T_{DS} is 0.3 s, and k_l is varying from -1.2 to 2.2. The blue circle denotes a unit circle in the complex plane, and the red curve λ_{max} is the eigenvalue locus which has the greatest absolute value within all the eigenvalues of the state matrix B. When k_l changes from -1.2 to 2.2, the eigenvalue locus starts from the right side and ends at the left side of the unit circle. The locus crosses the circle at $k_l = 0$ and $k_l = 2$, where the equality $|k_f - k_l| = 1$ holds. When $|k_f - k_l| < 1$, the maximum eigenvalue stays inside the unit circle, ensuring the system is stable. Again, this firms the conclusion we mentioned before: when $T_{iter} \geq 1.2$ s, the system's stability condition can be simplified to the inequality of $|k_f - k_l| < 1$.

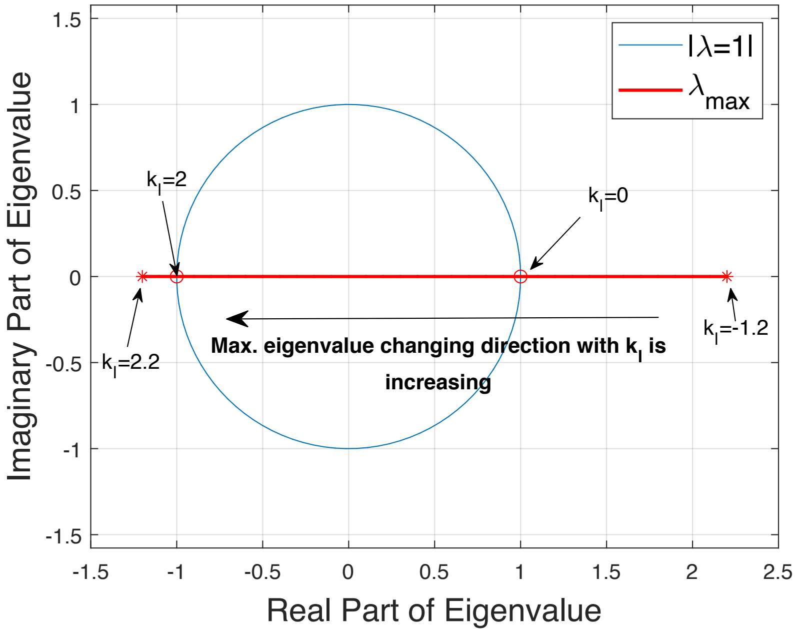


Figure 3.6: Maximum eigenvalue locus of two waypoints case for fixed parameters: $k_f = 1$, $T_{iter} = 2.4$ s, $b \approx 0.3112$ and $k_\xi = 4$. The k_l is varying from -1.2 to 2.2.

Commanded-Error-Based Adaptation Framework

The commanded-error-based adaptation framework replaces the VRP measurement error with the commanded error in the ILC updating law. The disturbance error terms related to $\boldsymbol{d}(0)$ and $\boldsymbol{d}(T)$ in (3.50), (3.56) and (3.60) disappear. So the state-space model of the commanded-error-based adaptation framework becomes:

$$\begin{bmatrix} \boldsymbol{\xi}^1_{i+1} \\ \boldsymbol{v}^1_{l,i+1} \\ \boldsymbol{v}^2_{l,i+1} \end{bmatrix} = \underbrace{\begin{bmatrix} b_1 & b_2 & b_3 \\ b_4 & b_5 & b_6 \\ b_7 & b_8 & b_9 \end{bmatrix}}_{B} \begin{bmatrix} \boldsymbol{\xi}^1_{i} \\ \boldsymbol{v}^1_{l,i} \\ \boldsymbol{v}^2_{l,i} \end{bmatrix} + \begin{bmatrix} b_{10}\boldsymbol{D}(T) - \frac{1}{b}\boldsymbol{D}(2T) \\ \boldsymbol{0} \\ \frac{k_l\zeta}{b}\boldsymbol{D}(T) \end{bmatrix} + \begin{bmatrix} \boldsymbol{FFT}_{11} \\ \boldsymbol{FFT}_{7} \\ \boldsymbol{FFT}_{9} \end{bmatrix}. \tag{3.62}$$

Comparing (3.61) and (3.62), the state matrices are the same. This guarantees that they have the same stability condition.

3.2.3 Generalized Case

In this section, the stability of a general system is further analyzed. The general system means that the waypoint number in one iteration is the same as the one applied in the simulations and experiments. It is no longer a simplified waypoint model described in Section 3.2.1 and 3.2.2. In particular, $n_w = 240$ is used for the analysis. This waypoint number is the same as walking with the parameters $T_{SS} = 0.9$ s, $T_{DS} = 0.3$ s and $\Delta t_{ILC} = 0.01$ s. Moreover, we also proved in Section 3.2.1 and 3.2.2 that the commanded-error-based adaptation framework has the same stability property as the measured-error-based framework. So the analysis in this section only focuses on the measured-error-based framework.

Because of the huge dimension of the waypoint per iteration, the *Symbolic Math Toolbox* of *Matlab* is applied to compute the state-space model by following the calculation ideas shown in Section 3.2.1 and 3.2.2. Figure 3.7 shows an example of the relation between the eigenvalues of the state-space model and the waypoint duration T. In this case, the $k_f = 0.8$, and $k_l = 1$. The red curve in the figure is the eigenvalue that always converges to 0 as T increases, where the blue curves denote all the rest eigenvalues. When T_{iter} is greater than 0.2 s, all the eigenvalues are smaller than 1. Particularly, the blue curves converge to $|k_f - k_l| = 0.2$ for $T_{iter} \geq 1.2$ s (i.e., $T \geq 0.005$ s).

Another example of the eigenvalue analysis is shown in Figure 3.8. Here, the same parameters setting used in Figure 3.6 is applied to draw the maximum eigenvalue locus. As result, the loci of Figure 3.6 and 3.8 are exactly the same.

From the results shown in Figure 3.7 and 3.8, we can conclude that whatever the waypoint number in one iteration is, the stability proof of a general system is identical to the proof of the simplified waypoint model shown in Section 3.2.1 and 3.2.2. Since the walking iteration duration in simulations and experiments was not less than 1.8 s, the inequality $|k_f - k_l| < 1$ is the only condition that needs to be considered for the stability of the VRP-OILC framework.

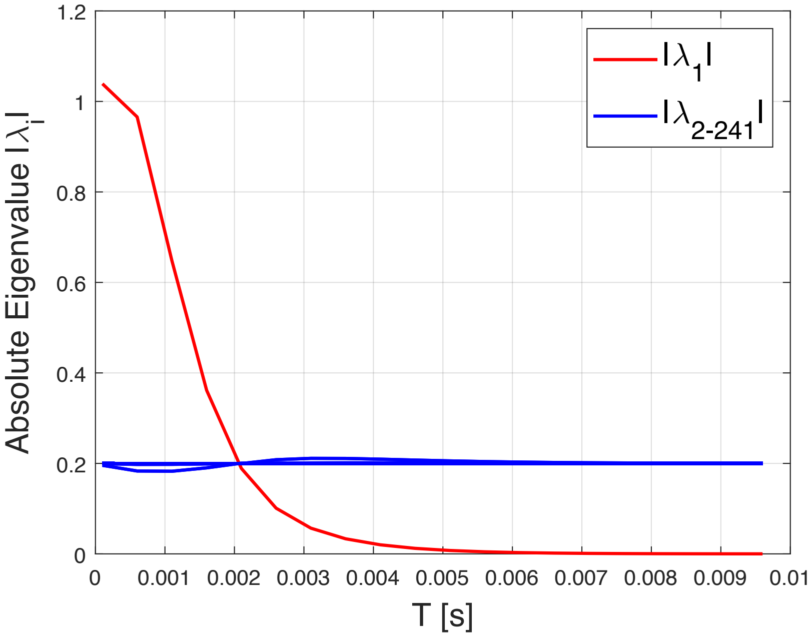


Figure 3.7: Relation between the eigenvalues and the waypoint duration T for $k_f = 0.8$ and $k_l = 1$ (general 240 waypoints case). λ_i denotes the i^{th} eigenvalue.

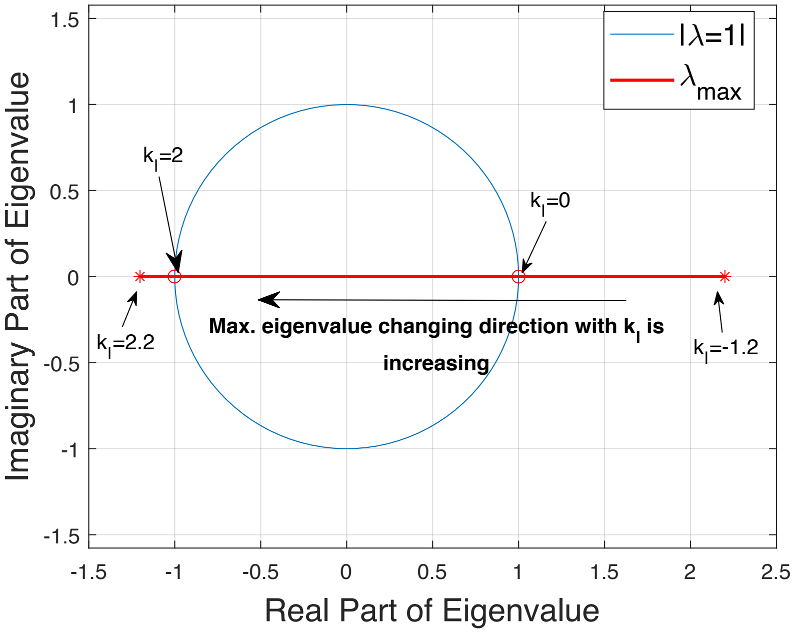


Figure 3.8: Maximum eigenvalue locus of general 240 waypoints case for fixed parameters: $k_f = 1$, $T_{iter} = 2.4$ s, $b \approx 0.3112$ and $k_\xi = 4$. The k_l is varying from -1.2 to 2.2.

3.3 Convergence Analysis

Section 3.2 proved the stability of the VRP-OILC framework. In this section, the convergence property of the system is further analyzed. To simplify the model, the one waypoint model described in Section 3.2.1 is used. To further simplify the analysis, walking in place behavior is considered. The local coordinate frames coincide with the global frame, where all the trajectories can be represented within the same frame.

Assume that the robot system is stable under a certain parameters' setting. Since the system is stable and no external force exists, all the VRP and DCM trajectories (including adjusted trajectory, original desired trajectory and measurement trajectory for both VRP and DCM, and commanded trajectory only for VRP) converge locally in each iteration. Because only the walking in place behavior is considered, it means that as the iteration index i increases to ∞, the convergence of the VRP and DCM terms is shown as:

$$\begin{gathered}
\lim_{i\to\infty} \boldsymbol{v}_{d,i+1} = \lim_{i\to\infty} \boldsymbol{v}_{d,i} = \boldsymbol{v}_d; \quad \lim_{i\to\infty} \boldsymbol{v}_{l,i+1} = \lim_{i\to\infty} \boldsymbol{v}_{l,i} = \boldsymbol{v}_l; \\
\lim_{i\to\infty} \boldsymbol{\xi}_{d,i+1} = \lim_{i\to\infty} \boldsymbol{\xi}_{d,i} = \boldsymbol{\xi}_d; \quad \lim_{i\to\infty} \boldsymbol{\xi}_{l,i+1} = \lim_{i\to\infty} \boldsymbol{\xi}_{l,i} = \boldsymbol{\xi}_l; \\
\lim_{i\to\infty} \boldsymbol{v}_{c,i} = \boldsymbol{v}_c; \quad \lim_{i\to\infty} \boldsymbol{v}_{m,i} = \boldsymbol{v}_m; \quad \lim_{i\to\infty} \boldsymbol{\xi}_i = \boldsymbol{\xi}; \qquad (3.63)
\end{gathered}$$

According to (3.26), the error model between commanded and measured VRP after the convergence is:

$$\boldsymbol{d}(0) = \boldsymbol{v}_m - \boldsymbol{v}_c. \qquad (3.64)$$

Measurement-Error-Based Adaptation Framework

In the measurement-error-based adaptation framework, we investigate where the measured VRP converges to. Firstly, we start from the calculation of learned DCM $\boldsymbol{\xi}_l$ before the future VRP waypoint is updated as:

$$\boldsymbol{\xi}_l = \alpha \boldsymbol{v}_1 + \beta \boldsymbol{v}_d + r \boldsymbol{\xi}_d. \qquad (3.65)$$

When all the VRP and DCM terms converge, the DCM control law becomes:

$$\boldsymbol{v}_c = \boldsymbol{v}_l + \zeta \left(\boldsymbol{\xi} - \boldsymbol{\xi}_l \right). \qquad (3.66)$$

The ILC updating equation of the $\boldsymbol{v}_l$ is expressed and expanded by inserting (3.64) - (3.66) as:

$$\begin{aligned}
\boldsymbol{v}_l &= \boldsymbol{v}_d + k_f \left(\boldsymbol{v}_l - \boldsymbol{v}_d \right) + k_l \left(\boldsymbol{v}_d - \boldsymbol{v}_m \right) \\
&= \left(k_f - k_l + \alpha k_l \zeta \right) \boldsymbol{v}_l - k_l \boldsymbol{d}(0) + \left[k_l(\beta\zeta + 1) - k_f + 1 \right] \boldsymbol{v}_d - k_l \zeta \boldsymbol{\xi} + \gamma k_l \zeta \boldsymbol{\xi}_d.
\end{aligned} \qquad (3.67)$$

There is a measured DCM waypoint $\boldsymbol{\xi}$ on the right hand side of (3.67) which is still unknown. To calculate the $\boldsymbol{\xi}$, the learned DCM waypoint $\boldsymbol{\xi}_l^\star$ is first computed after the future VRP waypoint is updated:

$$\boldsymbol{\xi}_l^\star = \alpha \boldsymbol{v}_l + \beta \boldsymbol{v}_l + \gamma \boldsymbol{\xi}_l. \tag{3.68}$$

According to the calculation in (3.33), the $\boldsymbol{\xi}$ is expressed as:

$$\boldsymbol{\xi} = \boldsymbol{\xi}_l + e^{-k_\xi T}(\boldsymbol{\xi} - \boldsymbol{\xi}_l^\star) - \frac{\boldsymbol{D}(T)}{b}. \tag{3.69}$$

Inserting the (3.65) and (3.68) into (3.69), the $\boldsymbol{\xi}$ finally becomes:

$$\boldsymbol{\xi} = \underbrace{\frac{\left[\alpha - e^{-k_\xi T}(1 - \gamma + \alpha\gamma)\right]}{1 - e^{-k_\xi T}}}_{\phi} \boldsymbol{v}_l + \underbrace{\frac{\left(\beta - e^{-k_\xi T}\beta\gamma\right)}{1 - e^{-k_\xi T}}}_{\kappa} \boldsymbol{v}_d + \underbrace{\frac{\gamma - e^{-k_\xi T}\gamma^2}{1 - e^{-k_\xi T}}}_{\psi} \boldsymbol{\xi}_d - \underbrace{\frac{\boldsymbol{D}(T)}{b(1 - e^{-k_\xi T})}}_{\varpi}. \tag{3.70}$$

Using (3.70) to replace the $\boldsymbol{\xi}$ in (3.67), the $\boldsymbol{v}_l$ is finally calculated as:

$$\boldsymbol{v}_l = \underbrace{\frac{-\left[k_f - k_l(\beta\zeta + 1) + k_l\kappa\zeta - 1\right]}{1 - \eta}}_{\varrho} \boldsymbol{v}_d + \underbrace{\frac{\gamma k_l \zeta - k_l \psi \zeta}{1 - \eta}}_{\digamma} \boldsymbol{\xi}_d - \frac{k_l \boldsymbol{d}(0)}{1 - \eta} + \frac{k_l \varpi \zeta}{1 - \eta}, \tag{3.71}$$

where $\eta = (k_f - k_l + \alpha k_l \zeta - k_l \phi \zeta) \neq 1$. Finally, the formula of the measured VRP $\boldsymbol{v}_m$ can be derived from (3.67) and (3.71) as:

$$\begin{aligned} \boldsymbol{v}_m &= \frac{1 - k_f + k_l}{k_l} \boldsymbol{v}_d + \frac{k_f - 1}{k_l} \boldsymbol{v}_l \\ &= \frac{k_l + (k_f - 1)(\varrho - 1)}{k_l} \boldsymbol{v}_d + \frac{(k_f - 1)\digamma}{k_l} \boldsymbol{\xi}_d + \frac{(\boldsymbol{d}(0) - \varpi\zeta) \cdot (k_f - 1)}{\eta - 1}. \end{aligned} \tag{3.72}$$

From (3.72), we conclude that the measured VRP $\boldsymbol{v}_m$ converges to different trajectories by setting the parameters differently. Specifically, the measured VRP $\boldsymbol{v}_m$ equals the original desired VRP $\boldsymbol{v}_d$ when $k_f = 1$. This theoretically proves that the framework can successfully bring the measured VRP trajectory back to the original desired VRP trajectory when the forgetting factor k_f is set to 1. This convergence analysis can also be applied to other walking behaviors, e.g., forward walking.

Commanded-Error-Based Adaptation Framework

By looking closely at the formulas (3.65) - (3.72), the disturbance term $\boldsymbol{d}(0)$ does not exist in the analysis of the commanded-error-based adaptation framework because the $\boldsymbol{v}_c$ is directly used in the ILC updating law. Therefore, the formula of the commanded VRP $\boldsymbol{v}_c$ can be expressed as:

$$\boldsymbol{v}_c = \frac{k_l + (k_f - 1)(\varrho - 1)}{k_l} \boldsymbol{v}_d + \frac{(k_f - 1)\digamma}{k_l} \boldsymbol{\xi}_d + \frac{\varpi\zeta \cdot (k_f - 1)}{1 - \eta}. \tag{3.73}$$

Again, the $\boldsymbol{v}_c$ converges to $\boldsymbol{v}_d$ when k_f equals 1.

Chapter 4

Evaluation

This chapter shows the performance of the VRP-OILC framework. In Section 4.1, the results of the simulation are informed. Specifically, we simulated model inaccuracies on the robot in OpenHRP3, then applied the measurement-error-based adaptation framework. The measurement method used in the simulation was based on FT sensors. The performance of the framework was first compared with the framework included a varying forgetting factor design. After that, the performance was evaluated for different k_l. Furthermore, the performance between the framework with and without using the pre-learning procedure was compared. Section 4.2 presents the experimental results of the humanoid robot TORO (see Figure 4.1). In particular, the measurement-error-based adaptation framework, where the VRP measurement was based on the DCM model, was first tested on TORO. Secondly, the commanded-error-based VRP-OILC for normal speed and fast speed walking was applied. Lastly, the pre-learning design was tested on TORO for fast walking. Section 4.3 summarizes all the simulation and experimental results.

For the setup of VRP-OILC, there are two important design decisions:

1) All the results shown in Section 4.1 - 4.2 are based on the 2-dimensional VRP-OILC framework. It means that the ILC updating law only learns from the VRP error in the X and Y direction. The reason for giving up the learning in the Z direction is: only the VRP in X and Y direction needs to be inside the support polygon to avoid falling.

2) The ILC updating interval Δt_{ILC} was always set to 0.01 s by default among all the simulations and experiments, except for specified declarations. This time value is chosen to avoid the problems caused by the measurement noise.

4.1 Simulation Results

Because the FT sensors in the feet of the TORO were not available for performing experiments, it is impossible to measure the VRP by the FT-sensor-based method

Figure 4.1: DLR humanoid robot TORO.

in the experiments. Therefore, the measurement-error-based adaptation framework with this measurement method was tested in simulation.

To simulate model inaccuracies, almost all the robot link masses were changed with the principle of maintaining the total robot's mass to be 77.5 kg. Appendix A.1 lists the link's name whose mass was modified (the first column), the corresponding original mass (the second column), and the modified mass (the third column).

4.1.1 General Framework

In the general framework, the pre-learning procedure was not applied. The straight-forward walkings with SS time $T_{SS} = 0.9$ s, DS time $T_{DS} = 0.3$ s and sagittal step distance 0.15 m were conducted with respect to different learning parameters k_f and k_l.

VRP Discontinuity and Varying Forgetting Factor Method

Figure 4.2 illustrates a comparison of the VRP trajectory with and without using VRP-OILC for constant $k_f = k_l = 1$. The VRP measurement trajectory $\boldsymbol{v}_{m,ILC}$ obtained by using VRP-OILC converged to the desired trajectory during the SS phases. However, a discontinuity of the measured VRP during the DS phases caused by the foot landing and lifting can be noticed. A zoomed-in discontinuity can be seen in the upper graph of Figure 4.3. The VRP-OILC with constant forgetting factor $k_f = 1$ did not suppress the discontinuity of the measured VRP and even exacerbated the VRP oscillation during the DS phases. This exacerbated oscillation is why the average measured VRP error $\mathrm{e}_{\mathrm{ILC,\,c}}$ (c stands for using constant k_f) increased from the 2^{nd} iteration in Figure 4.4. Inspired by Hu et al. [HOL16], a varying forgetting factor method was applied during the DS phases. In particular, this method changed the k_f from 1 to 0.5 during the DS phase and changed it back to 1 at the beginning of the next SS phase. The lower graph of Figure 4.3 shows that the discontinuity of the measured VRP was well suppressed by the varying forgetting factor method. So as shown in Figure 4.4, using the VRP-OILC with the varying forgetting factor method not only reduced the average VRP error but also kept the learning result not to drift compared with the constant forgetting factor case. All the simulations and experiments below were based on the varying forgetting factor method.

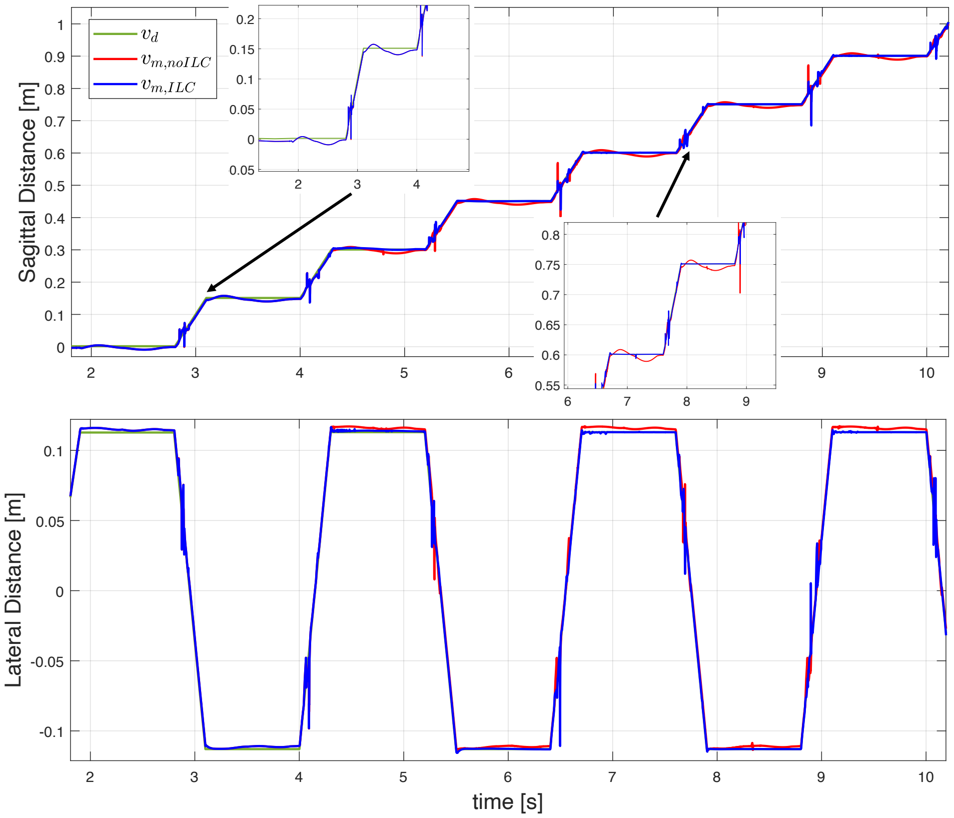


Figure 4.2: Simulation result of a straightforward walking with SS time $T_{SS} = 0.9$ s, DS time $T_{DS} = 0.3$ s, sagittal step distance 0.15 m, and learning parameters $k_l = 1$, $k_f = 1$.

Convergence Speed of Different Learning Gain

To investigate the effect of learning gain on convergence speed, different learning gains on the same walking behavior were used. As for the varying forgetting factor, the k_f was set to 1 during the SS phases and to 0.5 during the DS phases. As

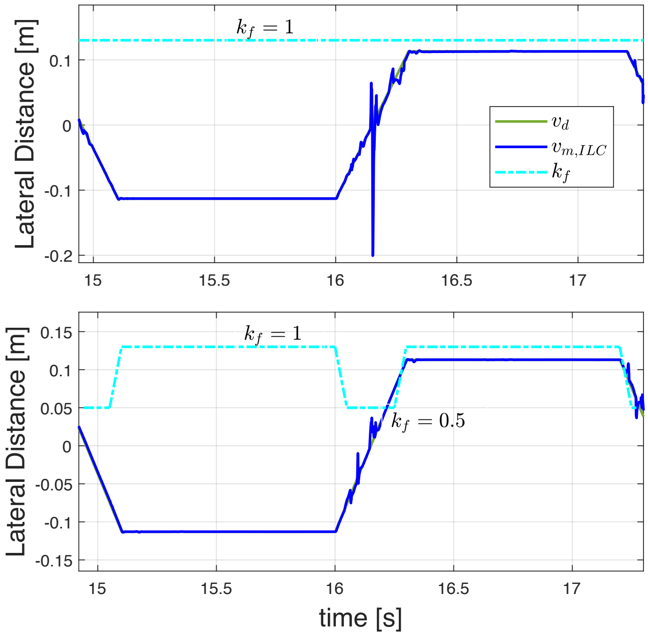


Figure 4.3: Comparison of using constant and varying forgetting factor. The upper plot shows a discontinuity of the measured VRP caused by the foot impact and lifting; The lower plot shows the varying forgetting factor method. Here the k_f varied from 1 to 0.5 during the DS phases, and remained to 1 during the SS phases.

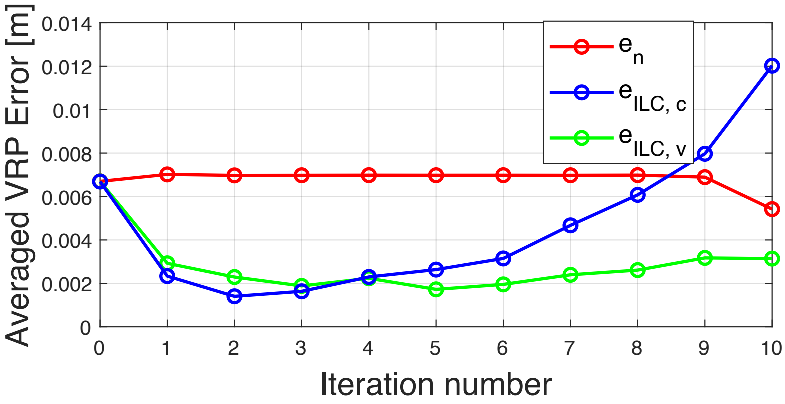


Figure 4.4: 2D average measured VRP error of the straightforward walking with SS time $T_{SS} = 0.9$ s, DS time $T_{DS} = 0.3$ s, sagittal step distance 0.15 m. $\mathrm{e_n}$ denotes the average error without using the VRP-OILC, where $\mathrm{e_{ILC,\,c}}$ is the average error using the framework with a constant k_f, and $\mathrm{e_{ILC,\,v}}$ represents the average error with the varying k_f method. The frameworks of both $\mathrm{e_{ILC,\,c}}$ and $\mathrm{e_{ILC,\,v}}$ used $k_l = 1$.

discussed in Section 3.2, the framework should converge for $0 < k_l < 1.5$ when $0.5 \leq k_f \leq 1$. Regarding the different k_l from 0.4 to 1.4, Figure 4.5 shows the 2D average VRP errors in each iteration. The VRP-OILC converged faster with a higher k_l value but ended up with a larger average VRP error. The framework started to diverge for $k_l = 1.4$ in simulation.

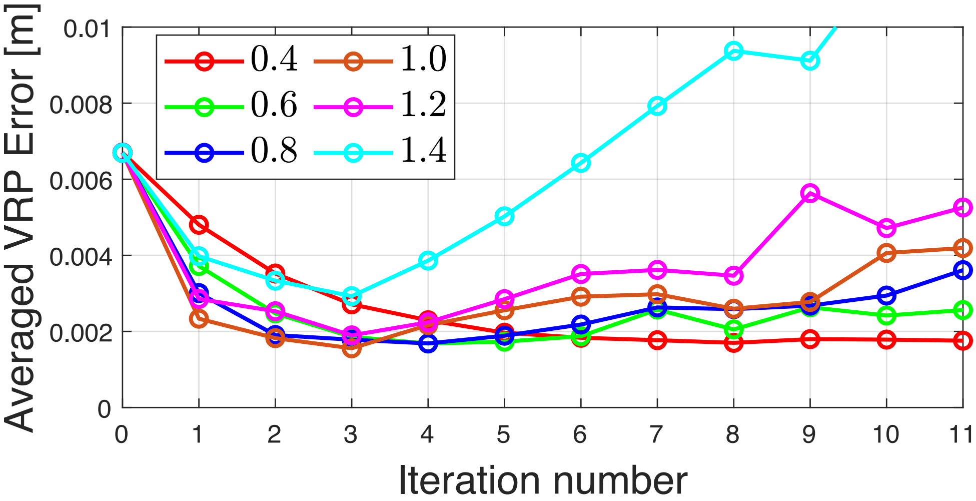


Figure 4.5: 2D average measured VRP error of the straightforward walking with SS time $T_{SS} = 0.9$ s, DS time $T_{DS} = 0.3$ s, sagittal step distance 0.15 m with respect to different learning gains.

4.1.2 Framework with Pre-Learning Procedure

To construct the pre-learned knowledge database, we sampled from the sagittal straightforward (SSW), lateral straightforward (LSW) and circle walking under the specified parameter vector sets shown in Appendix A.2. Specifically in Table A.2, the $T_{iter} = 2.0$ s means $T_{SS} = 0.8$ s and $T_{DS} = 0.2$ s, where $T_{iter} = 2.4$ s denotes $T_{SS} = 0.9$ s and $T_{DS} = 0.3$ s. For each sample, we set $k_{l,sa} = 1$ in the sagittal direction and $k_{l,la} = 0.5$ in the lateral direction. The k_f was varying from 1 to 0.5 during the DS phases and remaining at 1 during the SS phases. To test the performance of the pre-learned knowledge database, fast straightforward walking with $T_{SS} = 0.7$ s, $T_{DS} = 0.2$ s, sagittal step distance 0.15 m was conducted with respect to $k_{l,sa} = k_{l,la} = 0.4$ and k_f varied in the range of $[0.5, 1]$. Figure 4.6 compares the average VRP error using and without using the pre-learned knowledge. By using the pre-learning procedure, the average error $e_{\text{ILC, p}}$ converged faster than the error $e_{\text{ILC, np}}$ without using the procedure. Moreover, the pre-learning procedure yielded a lower average error at the 0^{th} iteration (i.e., at the beginning of the VRP-OILC being used).

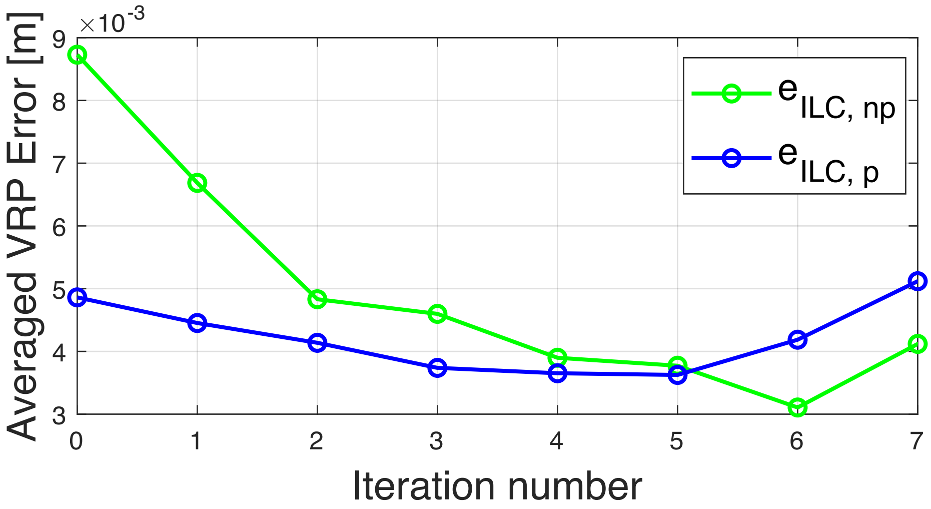


Figure 4.6: 2D average measured VRP error of using and without using the pre-learning database in simulation. $e_{\mathrm{ILC,\ np}}$ means that no pre-learning procedure was applied, where $e_{\mathrm{ILC,\ p}}$ denotes that the pre-learned knowledge was used.

4.2 Experimental Results

4.2.1 Measurement-Error-Based Reference Trajectories Adaptation

Since the FT sensors in the feet of the robot were not available, the measurement-error-based adaptation framework that measured the VRP by calculating from the DCM dynamics model was used, as shown in 2.1.4. An experiment of walking in place was conducted with $T_{SS} = 0.9$ s, $T_{DS} = 0.3$ s, $k_{l,sa} = 1$, $k_{l,la} = 0.5$, varying k_f from 1 to 0.5, and $\Delta t_{ILC} = 0.1$ s. Figure 4.7 illustrates the resulted measured VRP trajectory $\boldsymbol{v}_{m,ILC}$ and the corresponding adjusted reference trajectory $\boldsymbol{v}_l$. Both of them showed a tendency of divergence and oscillation, which exceeded the torque limits of the left foot and led to the robot's fall at $t = 12.2$ s. Therefore, this VRP measurement method cannot be a successful alternative to the FT-sensor-based method.

4.2.2 Commanded-Error-Based Reference Trajectories Adaptation

As an alternative solution for the measurement-error-based reference trajectories adaptation, the commanded-error-based reference trajectories adaptation was used as a compromise shown in Section 3.1.3.

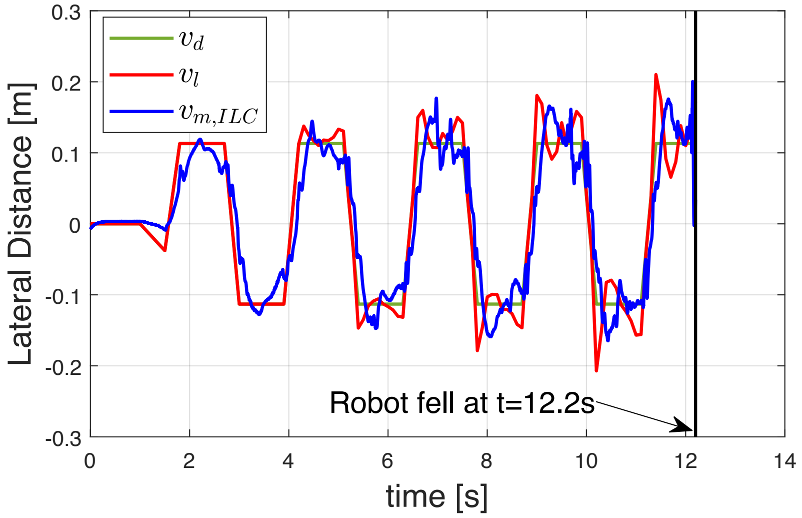
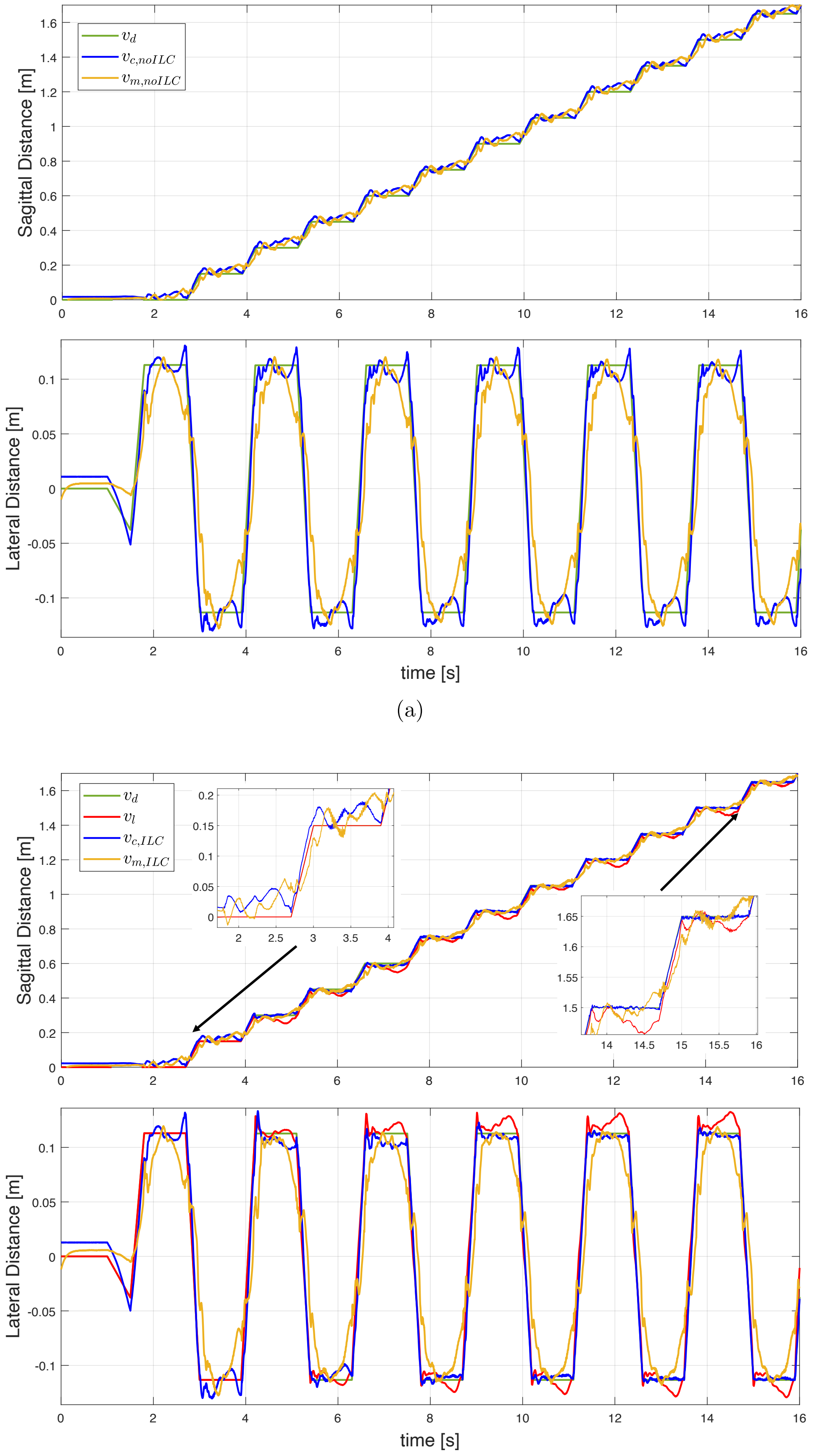


Figure 4.7: The lateral result of the experiment applying measurement-error-based VRP-OILC for walking in place behavior with respect to SS time $T_{SS} = 0.9$ s, DS time $T_{DS} = 0.3$ s, $k_{l,sa} = 1$, $k_{l,la} = 0.5$, k_f varying from 1 to 0.5 and ILC updating interval $\Delta t_{ILC} = 0.1$ s.

General Framework

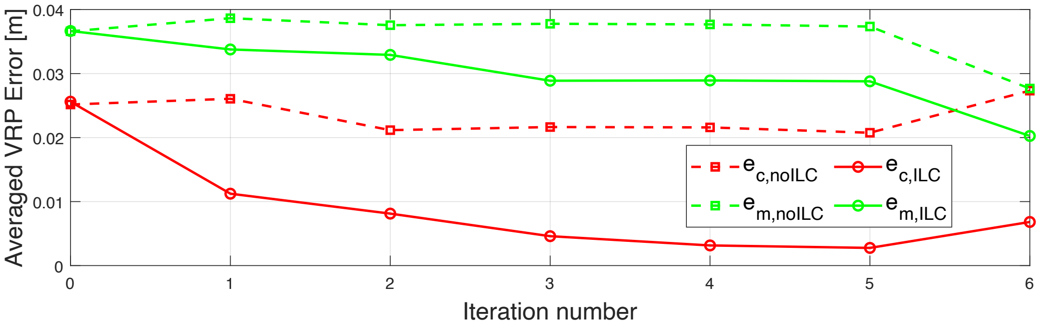
As aforementioned, the pre-learning procedure was not applied yet in a general framework. Here, two different types of walking were conducted: normal speed and fast speed walking. In the normal speed walking, the SS time was $T_{SS} = 0.9$ s and DS time was $T_{DS} = 0.3$ s, where they were set to $T_{SS} = 0.7$ s and $T_{DS} = 0.2$ s respectively in the fast speed walking.

- **Normal Speed Walking:** Figure 4.8 shows a comparison of the performance between using and without using the commanded-error-based adaptation framework for a straightforward walking with sagittal step length 0.15 m. The measured VRP trajectory $\boldsymbol{v}_{m,noILC}$ and commanded VRP trajectory $\boldsymbol{v}_{c,noILC}$ without using the framework are illustrated in Figure 4.8(a). After using the commanded-error-based VRP-OILC as shown in Figure 4.8(b), the commanded VRP converged to the original reference trajectory $\boldsymbol{v}_d$ in both sagittal and lateral direction, while the measured VRP was closer to the original reference trajectory as well. The convergence of the commanded and measured VRP can be reflected by the reduced average VRP error illustrated in Figure 4.8(c), where the average commanded error was reduced from 25 mm to a value less than 5 mm and the average measured error decreased from around 40 mm to 30 mm. Figure 4.9 compares the walking robustness before and after using the VRP-OILC framework. It demonstrates that the VRP-OILC improved the walking robustness by suppressing the heel-out behavior.


Sagittal Distance [m]
v_d
$v_{c,noILC}$
$v_{m,noILC}$
Lateral Distance [m]
time [s]
Sagittal Distance [m]
v_d
v_l
$v_{c,ILC}$
$v_{m,ILC}$
Lateral Distance [m]
time [s]

(a)

(b)



(c)

Figure 4.8: Performance comparison of using and without using commanded-error-based VRP-OILC for normal speed walking. a) VRP trajectories without using VRP-OILC in the sagittal direction (above) and in the lateral direction (below); b) VRP trajectories with using VRP-OILC in the sagittal direction (above) and in the lateral direction (below) regarding to parameter sets $k_{l,sa} = 1$, $k_{l,la} = 0.5$ and varying k_f in the range from 0.5 to 1; c) 2D average VRP error of using and without using VRP-OILC. $\mathrm{e}_{\mathrm{c,\ noILC}}$ and $\mathrm{e}_{\mathrm{m,\ noILC}}$ represent the average commanded and measured VRP error respectively without using VRP-OILC, where $\mathrm{e}_{\mathrm{c,\ ILC}}$ and $\mathrm{e}_{\mathrm{m,\ ILC}}$ denote the average errors when the VRP-OILC was applied.

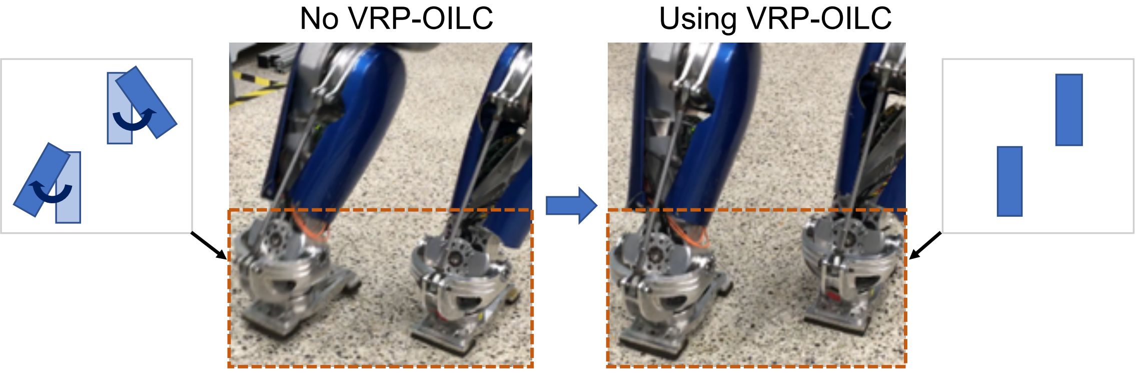


Figure 4.9: Walking robustness comparison of using and without using commanded-error-based VRP-OILC for normal speed walking. The picture on the left shows a heel-out sliding behavior during the walking without using VRP-OILC, while the picture on the right denotes that the VRP-OILC suppressed the heel-out behavior and enhanced the walking robustness.

• **Fast Speed Walking:** Without using the VRP-OILC framework, the robot did not finish the fast walking and fell at the global time 4.6 s, as shown in Figure 4.10(a). On the contrary, the framework enabled the robot to walk successfully under a fast speed walking SS time for 0.7 s and a DS time for 0.2 s, which are less than the SS and DS lower time limit[1] for walking without VRP-OILC. Figure 4.10(b) illustrates the successful fast straightforward walking with respect to a sagittal step distance 0.15 m and the learning parameters $k_{l,sa} = 1$, $k_{l,la} = 0.5$, and varying k_f in the range from 0.5 to 1. In this figure, the commanded VRP trajectory converged to the original reference trajectory again, while the measured VRP trajectory in the sagittal direction was also getting closer to the reference trajectory. A more intuitive representation of the VRP convergence can be shown in Figure 4.10(c): the average commanded VRP error decreased from 30 mm to a value less than 10 mm, and the average measured error was reduced from 50 mm to around 40 mm simultaneously. Moreover, an unexpected toe-out sliding behavior occurred in the later stage after the VRP-OILC was applied in the case of fast walking, where there was not sliding on the feet in the early stage, as shown in Figure 4.11.

Framework with Pre-Learning Procedure

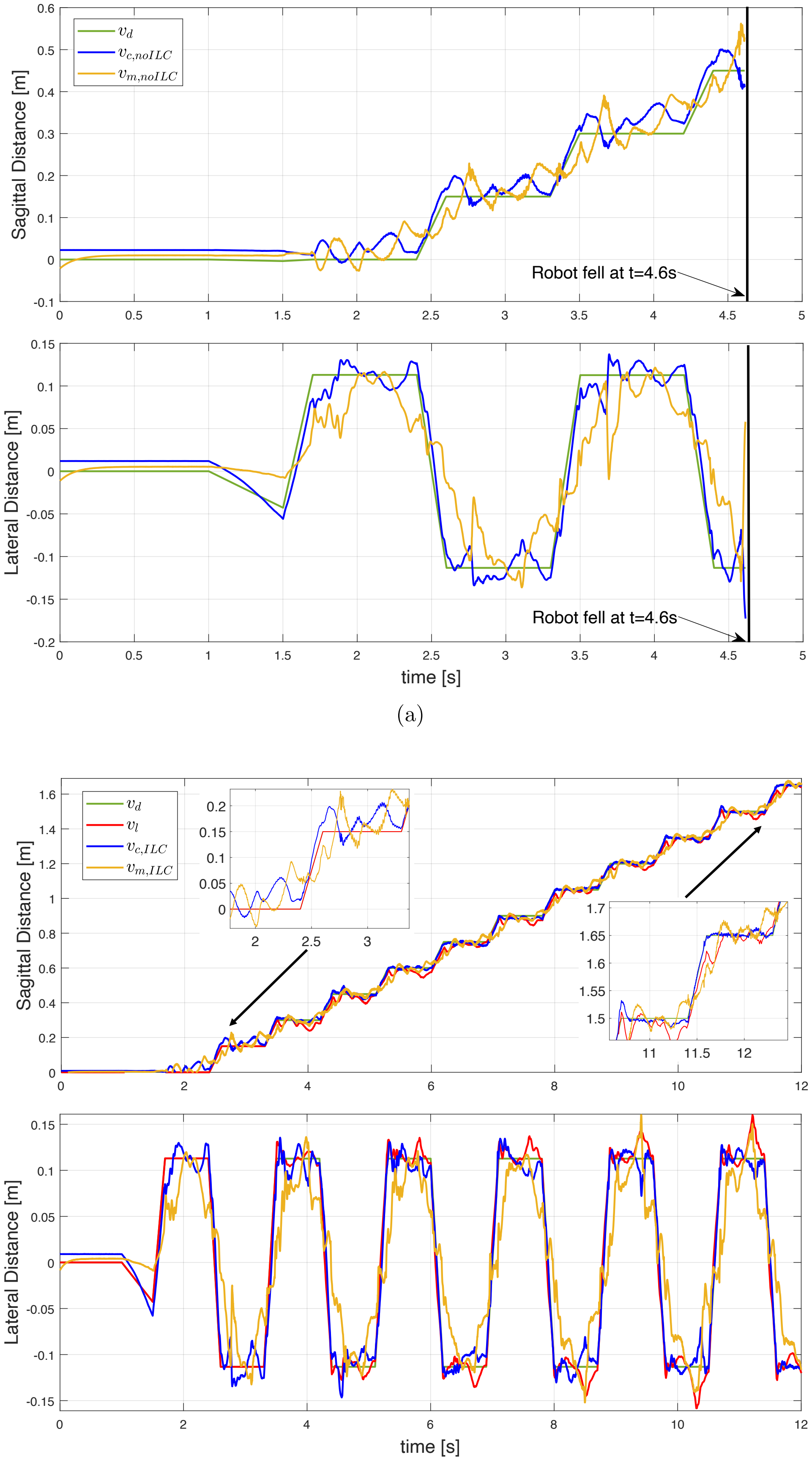
The robot's walking behavior is wobbling during the 0^{th} iteration's two steps for fast walking. As aforementioned, the pre-learning procedure aims to stabilize these two steps of an initial iteration. To learn the pre-learned knowledge database, the experiments were conducted by using different walking parameter sets of Appendix A.2. We set the $k_{l,sa} = 1$, $k_{l,la} = 0.5$ and k_f varying from 1 to 0.5 for each sample walking. As result, we visually observed that the fast walking robustness of the initial two steps was improved. Figure 4.12 shows the average commanded and measured VRP error before and after using pre-learned knowledge. The average commanded error of using the pre-learning procedure yielded 10 mm less than without using it at the initial iteration, while the average measured error of using the pre-learning procedure was reduced by about 5 mm. The toe-out behavior also occurred in the later stage of the fast walking even though the pre-learning procedure was used.

4.3 Result Summary

Based on the results of all the simulations and experiments, we can make the following conclusions:

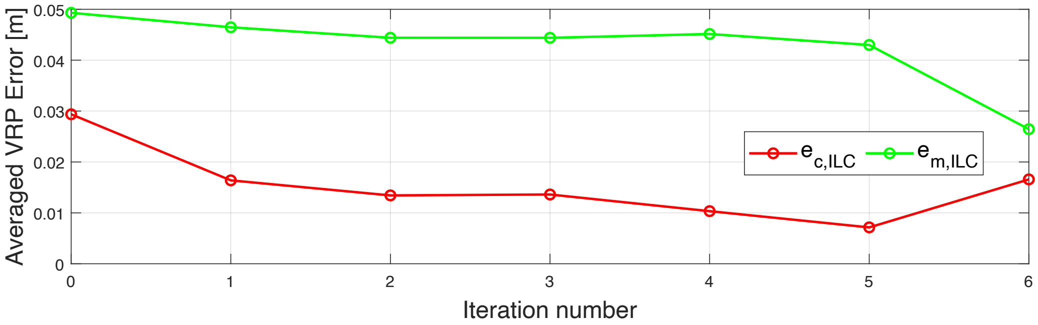
1. The VRP-OILC can improve the walking robustness by bringing the VRP trajectory back to its original reference trajectory. Specifically, a varying forgetting factor method can also suppress the VRP trajectory's oscillation during the DS.

[1] The shortest SS and DS time of TORO without using VRP-OILC is 0.9 s and 0.3 s respectively.


v_d
$v_{c,noILC}$
$v_{m,noILC}$
Sagittal Distance [m]
Robot fell at t=4.6s
Lateral Distance [m]
Robot fell at t=4.6s
time [s]
v_d
v_l
$v_{c,ILC}$
$v_{m,ILC}$
Sagittal Distance [m]
Lateral Distance [m]
time [s]

(a)

(b)



(c)

Figure 4.10: Performance comparison of using and without using commanded-error-based VRP-OILC for fast speed walking. a) VRP trajectories without using VRP-OILC in the sagittal direction (above) and in the lateral direction (below). The robot fell at 4.6 s; b) VRP trajectories with using VRP-OILC in the sagittal direction (above) and in the lateral direction (below) regarding to parameter sets $k_{l,sa} = 1$, $k_{l,la} = 0.5$ and varying k_f in the range from 0.5 to 1; c) 2D average VRP error of using VRP-OILC. $\mathrm{e}_{\mathrm{c,\ ILC}}$ and $\mathrm{e}_{\mathrm{m,\ ILC}}$ denote the average commanded and measured error respectively.

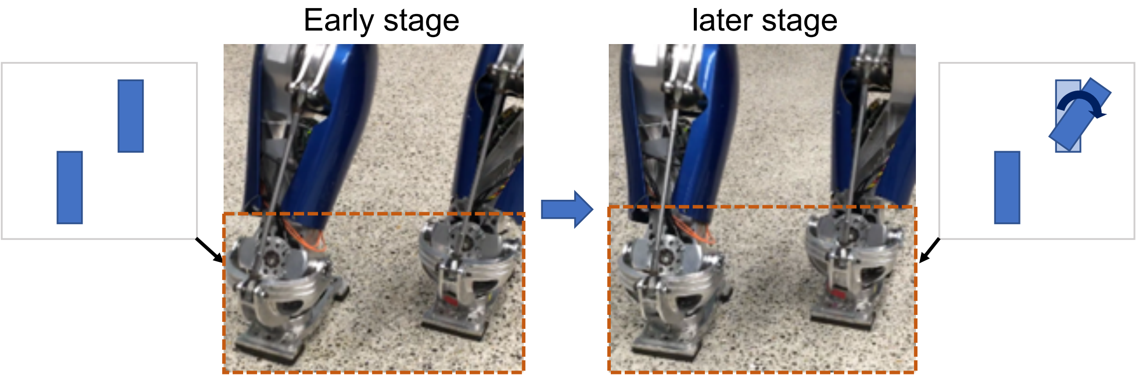


Figure 4.11: Walking robustness comparison of early and later stage after the commanded-error-based VRP-OILC was applied for fast speed walking.

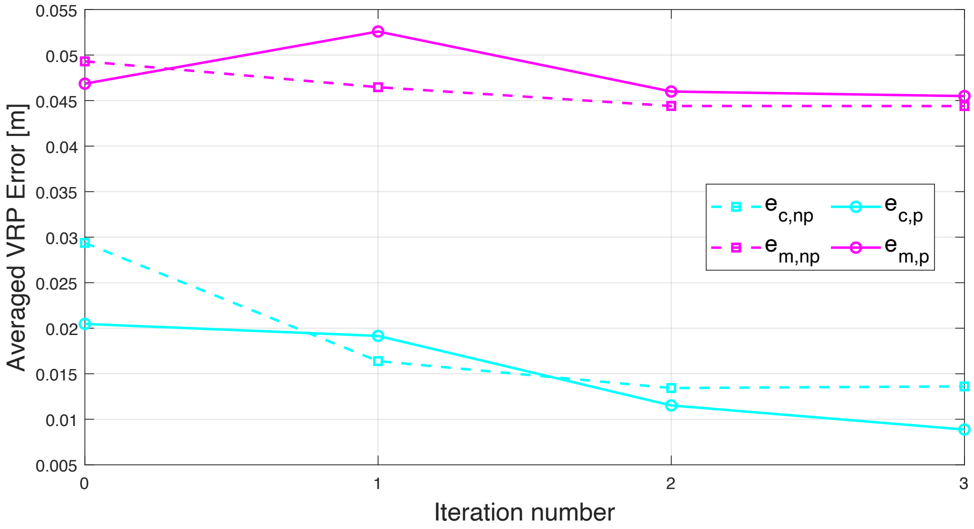


Figure 4.12: 2D average commanded and measured VRP error of using and without using the pre-learning database in experiment. $\mathrm{e}_{\mathrm{c,\ np}}$ and $\mathrm{e}_{\mathrm{m,\ np}}$ represent the average commanded and measured VRP error when no pre-learning procedure was applied, where $\mathrm{e}_{\mathrm{c,\ p}}$ and $\mathrm{e}_{\mathrm{m,\ p}}$ denote that the pre-learned knowledge was used.

2. The learning gain k_l determines the convergence speed of the VRP trajectory. It is crucial to choose the k_l and k_f such that the system is stable.

3. In the experiment, the measurement-error-based reference trajectories adaptation, whose VRP is measured using the DCM dynamics model, is not usable for the humanoid robot's walking. The commanded-error-based VRP-OILC can be an alternative to improve the walking robustness. The improved walking robustness can be represented by the suppression of the heel-out behavior in normal speed walking.

4. Moreover, the commanded-error-based VRP-OILC can also enable the robot to achieve fast walking with the SS and DS time less than 0.9 s and 0.3 s respectively, which are the shortest SS and DS time limit in case that no VRP-OILC is applied. However, an unexpected toe-out behavior will occur in the later stage of fast speed walking after the VRP-OILC is used.

5. The pre-learning procedure can make the walking more stable at the initial iteration of the fast walking. The reduced average VRP error can reflect this at the 0^{th} iteration.

Chapter 5

Discussion

As shown in Section 3.3, the VRP trajectory converges to the desired trajectory for a constant forgetting factor $k_f = 1$ theoretically. The VRP-OILC framework successfully brought the measured VRP back to the desired trajectory during the SS phases with a constant forgetting factor. We notice that a discontinuity of the measured VRP occurred when the foot was landing and the other foot was lifting during the walking (i.e., during the DS phases). Unfortunately, our framework with constant k_f failed to compensate for the discontinuity during the DS phases because the discontinuity did not occur in the same way repetitively during the DS phases inside the corresponding local iteration. According to [WGDI09], traditional ILC cannot deal well with non-repetitive behavior. As result, the learned VRP trajectory $\boldsymbol{v}_l$ was accumulated for a constant $k_f = 1$ and grew larger among the iteration, which led to the oscillation of the measured VRP during the DS phases. The oscillation was reflected by the increasing average VRP error (the blue curve) in Figure 4.4. On the contrary, the varying k_f method forgot some part of the learned VRP trajectory from the last iteration to update the current iteration's learned trajectory. In this way, the accumulation (or divergence) of the learned VRP trajectory was suppressed. Consequently, the oscillation of the measured VRP was not worsened and was even suppressed in the long term. This indicates that the varying forgetting factor method can adapt the traditional ILC to a non-repetitive circumstance.

As shown in Section 4.2.1, it failed to use the VRP measurement calculated from the DCM dynamics model for the learning. In fact, this is exactly the result we expected. As introduced in Section 2.1.4, the measured VRP based on the DCM model is calculated by the measured DCM position and velocity. Actually, the DCM position and velocity were strongly filtered signals because the joint velocity $\dot{\boldsymbol{q}}$ was filtered by a strong low pass filter before calculating the CoM velocity $\dot{\boldsymbol{x}}$ by $\dot{\boldsymbol{x}} = \boldsymbol{J}\dot{\boldsymbol{q}}$, and the CoM velocity was filtered by a low pass filter again before calculating the DCM position by $\boldsymbol{\xi} = \boldsymbol{x} + b\,\dot{\boldsymbol{x}}$. A strong low pass filter was used again on the resulted DCM velocity before calculating the measured VRP by equation (2.17). All these low pass filters were designed to filter out the measurement noise during the experiment. However, strong filters also led to a deformation of the output signal. Therefore, the calculated measurement VRP was severely deformed and became not reliable for the ILC method.

The commanded-error-based VRP-OILC improved the walking robustness during the experiments. Without using the VRP-OILC framework on a straightforward walking at normal speed, the measured VRP was almost higher than the original reference VRP in sagittal direction during the SS phases, as shown in the upper graph of Figure 4.8(a). It means that the 2D projection of the measured VRP was located on the front of the support foot, which made the friction between the hindfoot and the ground less than the one of the front foot. As known, the whole-body dynamics requires a compensating torque to be applied at the point of contact during the swing leg motion. The compensating torque caused the heel-out sliding behavior since the friction difference existed during the SS phases. After using our framework, Figure 4.8(b) illustrates that the measured VRP in sagittal direction during the SS phases was closer to the desired VRP (i.e., the center of the foot), which reduced the friction difference between the front and hindfoot. Thus, the heel-out behavior was suppressed after using our framework in the normal speed walking.

Without using our framework, TORO cannot achieve fast walking because the VRP deviation was exacerbated by the effect of the model inaccuracies, as shown in Figure 4.10(a). Contrary, Figure 4.10(b) shows that the fast speed walking of TORO succeeded after applying VRP-OILC. However, an unexpected toe-out sliding occurred in the later stage of fast speed walking. As shown in the lower graph of Figure 4.10(b), the measured and learned VRP started to oscillate and diverge from the reference trajectory even during the SS phases in the later stage (i.e., from $t = 8$ s to the end of the walking), which produced again a friction difference between the inside and outside of the foot. The compensating torque in feet during the swing leg motion rotated the toe towards the outside. Besides, we can further conclude that the robot system produced a non-repetitive behavior in the lateral direction during the fast walking because the oscillation and divergence of the measured and learned VRP were caused by the non-repetitive system behavior, as mentioned before. To solve the toe-out behavior, we can either reduce the forgetting factor $k_{f,la}$ in the lateral direction or use an additional ILC to compensate for the foot torque, which causes a toe-out sliding behavior.

Chapter 6

Conclusion and Future Work

Model uncertainties are common problems that lead to an imperfect tracking of the reference trajectories. The imperfect tracking will limit the maximum walking speed and reduce the robustness of bipedal walking robots. In this thesis, an online VRP adaptation framework for biped walking based on the concept of ILC is proposed. In short, our framework learns the future iteration's VRP reference trajectory adaptation from the VRP error at the current iteration by using the ILC and saves the learned future adaptation into a VRP waypoint list for later reuse. The framework can be implemented in two different learning designs: measurement-error-based and commanded-error-based learning, where the former uses the VRP measured error and the latter uses the VRP commanded error for learning. Moreover, the framework is extended with a pre-compensation procedure, which predicts a compensative VRP reference trajectory from the database to stabilize the walking in the initial iteration and increase the VRP convergence speed among the whole walking sequence.

The stability of the framework is proved in the thesis. The stability condition can be simplified to the inequality of $|k_f - k_l| < 1$ when the iteration duration $T_{iter} > 1.2$ s. Additionally, we analyze the VRP convergence position theoretically. It is proved that the VRP converges to the original reference trajectory when $k_f = 1$ and the system is assumed to be stable.

All the simulation and experimental results demonstrate that the walking robustness was significantly improved after applying the VRP-OILC framework. Especially, the framework achieved fast walking, which is impossible without the framework. However, there are several limitations to our work:

1. The FT sensors were not available during the experiments. This makes the FT-sensor-based VRP measurement method impossible to be used in the framework.

2. The commanded-error-based adaptation framework cannot suppress the toe-out sliding behavior in the later stage of fast walking.

3. All the experiments were conducted on flat ground in the laboratory, where the external disturbances from the ground were subtle and repetitive. However, bipedal robot research's ultimate goal is to enable robots to walk robustly in different environments. Therefore, walking on complex terrains outside of the laboratory, where non-repetitive disturbances will be generated on the robot system, needs to be further studied.

Future work will be considered as follows:

1. We will test the measurement-error-based VRP adaptation framework, where the VRP measurement bases on the FT sensors, on TORO.

2. To suppress the toe-out behavior in the fast walking, we will try two different ideas in the future: i) reduce the value of the forgetting factor $k_{f,la}$ in the lateral direction; ii) apply an additional ILC method to compensate for the foot torque in yaw direction which causes the toe-out behavior.

3. Wang et al. [WGDI09] mentioned that the run-to-run control (R2R) is suitable for the case when the system has not good repetitive behaviors. For the robot walking in the complex environments outside the laboratory, an R2R-based VRP online adaptation framework can be developed to overcome the problems caused by the non-repetitive disturbances.

Appendix A

Parameter Setting in Simulation

A.1 Model Inaccuracies Modeling

Table A.1: Robot link masses before and after the modification for the model inaccuracies modeling in OpenHRP3.

Link name	**Original mass (kg)**	**Modified mass (kg)**
WAIST	19.6	21.6
NECK_LINK	0.2	1.2
HEAD_LINK	1.5	0.5
HIP_LINK	5.6	3.6
RARM_SHOULDER1	2.6	1.6
RARM_SHOULDER2	1.1	2.1
RARM_SHOULDER3	2.4	3.4
RARM_LOWERARM	2.2	1.2
LARM_SHOULDER1	2.6	1.6
LARM_SHOULDER2	1.1	2.1
LARM_SHOULDER3	2.4	3.4
LARM_LOWERARM	2.2	1.2
RLEG_HIP_R	3.2	4.2
RLEG_HIP_Y	2.6	0.6
RLEG_KNEE	5.7	6.7
LLEG_HIP_R	3.2	0.2
LLEG_HIP_P	2.6	8.6
LLEG_HIP_Y	2.6	0.6
LLEG_KNEE	5.7	4.7
LLEG_ANKLE_R	0.4	1.4
LLEG_ANKLE_P	1.4	0.4

A.2 Construction of Pre-Learned Knowledge Database

Table A.2: Walking parameter sets for constructing the pre-learned knowledge database.

Sagittal Straight Walking (SSW)	Lateral Straight Walking (LSW)	Circle Walking
$d_{sa} = \{-10, 15\}$cm	$d_{sa} = 0$ cm	$\Delta\alpha = \{-10, 10\}^\circ$
$d_{la} = 0$ cm	$d_{la} = \{-8, 8\}$cm	
$T_{iter} = \{2.0, 2.4\}$s	$T_{iter} = \{2.0, 2.4\}$s	$T_{iter} = \{2.0, 2.4\}$s

List of Figures

2.1 General humanoid robot dynamics. 9
2.2 Examples of 4-step VRP, DCM and CoM reference trajectory. 12

3.1 An example of two walking iterations. 22
3.2 Lateral VRP and DCM deviation by ILC learning from the previous iteration. 30
3.3 Lateral VRP trajectories by start waypoint learning for future iteration. 31
3.4 Relation between the eigenvalues and the waypoint duration T under different k_f and k_l (one waypoint case). 38
3.5 Relation between the eigenvalues and the waypoint duration T under different k_f and k_l (two waypoints case). 42
3.6 Maximum eigenvalue locus of two waypoints case for fixed parameters. 43
3.7 Relation between the eigenvalues and the waypoint duration T for $k_f = 0.8$ and $k_l = 1$ (general 240 waypoints case). 45
3.8 Maximum eigenvalue locus of general 240 waypoints case for fixed parameters. 45

4.1 DLR humanoid robot TORO. 50
4.2 Simulation result of a straightforward walking with SS time $T_{SS} = 0.9$ s, DS time $T_{DS} = 0.3$ s, sagittal step distance 0.15 m, and learning parameters $k_l = 1$, $k_f = 1$. 51
4.3 Comparison of using constant and varying forgetting factor 52
4.4 2D average measured VRP error of the straightforward walking with SS time $T_{SS} = 0.9$ s, DS time $T_{DS} = 0.3$ s, sagittal step distance 0.15 m. 52
4.5 2D average measured VRP error of the straightforward walking with SS time $T_{SS} = 0.9$ s, DS time $T_{DS} = 0.3$ s, sagittal step distance 0.15 m with respect to different learning gains. 53
4.6 2D average measured VRP error of using and without using the pre-learning database in simulation. 54
4.7 The lateral result of the experiment applying measurement-error-based VRP-OILC for walking in place behavior. 55

4.8 Performance comparison of using and without using commanded-error-based VRP-OILC for normal speed walking. 57
4.9 Walking robustness comparison of using and without using commanded-error-based VRP-OILC for normal speed walking. 57
4.10 Performance comparison of using and without using commanded-error-based VRP-OILC for fast speed walking. 60
4.11 Walking robustness comparison of early and later stage after the commanded-error-based VRP-OILC for fast speed walking was applied. 60
4.12 2D average commanded and measured VRP error of using and without using the pre-learning database in experiment. 61

Acronyms and Notations

DCM Divergent Component of Motion

VRP Virtual Repellent Point

CoM Center of Mass

ILC Iterative Learning Control

ZMP Zero-Moment Point

LIP Linear Inverted Pendulum

DF Dynamical Filter

ZMP-OILC ZMP-Based Online Iterative Learning Control Framework

CZMP Compensative ZMP

VRP-OILC VRP-Based Online Iterative Learning Control Framework

CoP Center of Pressure

eCMP Enhanced Centroidal Moment Pivot Point

CMP Centroidal Moment Pivot

SS Single Support

DS Double Support

FT Sensor Force-Torque Sensor

k-NN k-Nearest Neighbors Algorithm

SSW Sagittal Straightforward Walking

LSW Lateral Straightforward Walking

R2R Run-to-Run Control

Bibliography

[AKM84] Suguru Arimoto, Sadao Kawamura, and Fumio Miyazaki. Bettering operation of dynamic systems by learning: A new control theory for servomechanism or mechatronics systems. In *The 23rd IEEE Conference on Decision and Control*, pages 1064–1069. IEEE, 1984.

[BTA06] Douglas A Bristow, Marina Tharayil, and Andrew G Alleyne. A survey of iterative learning control. *IEEE control systems magazine*, 26(3):96–114, 2006.

[CKOY07] Youngjin Choi, Doik Kim, Yonghwan Oh, and Bum-Jae You. Posture/walking control for humanoid robot based on kinematic resolution of com jacobian with embedded motion. *IEEE Transactions on Robotics*, 23(6):1285–1293, 2007.

[EMO17] Johannes Englsberger, George Mesesan, and Christian Ott. Smooth trajectory generation and push-recovery based on divergent component of motion. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4560–4567. IEEE, 2017.

[EOAS15] 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.

[GS58] Ulf Grenander and Gabor Szegö. *Toeplitz forms and their applications.* Univ of California Press, 1958.

[HOL16] Kai Hu, Christian Ott, and Dongheui Lee. Learning and generalization of compensative zero-moment point trajectory for biped walking. *IEEE Transactions on Robotics*, 32(3):717–725, 2016.

[HRO16] Bernd Henze, Máximo A Roa, and Christian Ott. Passivity-based whole-body balancing for torque-controlled humanoid robots in multi-contact scenarios. *The International Journal of Robotics Research*, 35(12):1522–1543, 2016.

[KHK04] Fumio Kanehiro, Hirohisa Hirukawa, and Shuuji Kajita. Openhrp: Open architecture humanoid robotics platform. *The International Journal of Robotics Research*, 23(2):155–165, 2004.

[KKF$^+$85] Sadao Kawamura, Tatsuya Kawamura, Daisuke Fujino, Fumio Miyazaki, and Suguru Arimoto. Realization of biped locomotion by motion pattern learning. *Journal of the Robotics Society of Japan*, 3(3):177–187, 1985.

[KKK$^+$03] 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 *2003 IEEE International Conference on Robotics and Automation (Cat. No. 03CH37422)*, volume 2, pages 1620–1626. IEEE, 2003.

[KMM$^+$10] 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 *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4489–4496. IEEE, 2010.

[LTK93] Qinghua Li, Atsuo Takanishi, and Ichiro Kato. Learning control for a biped walking robot with a trunk. *Journal of the Robotics Society of Japan*, 11(7):1011–1019, 1993.

[MEG$^+$19] George Mesesan, Johannes Englsberger, Gianluca Garofalo, Christian Ott, and Alin Albu-Schäffer. Dynamic walking on compliant and uneven terrain using dcm and passivity-based whole-body control. In *2019 IEEE-RAS 19th International Conference on Humanoid Robots (Humanoids)*, pages 25–32. IEEE, 2019.

[MEOAS18] George Mesesan, Johannes Englsberger, Christian Ott, and Alin Albu-Schäffer. Convex properties of center-of-mass trajectories for locomotion based on divergent component of motion. *IEEE Robotics and Automation Letters*, 3(4):3449–3456, 2018.

[NG02] Mikael Norrlöf and Svante Gunnarsson. Time and frequency domain convergence properties in iterative learning control. *International Journal of Control*, 75(14):1114–1126, 2002.

[She66] Thomas B Sheridan. Three models of preview control. *IEEE Transactions on Human Factors in Electronics*, (2):91–102, 1966.

[SNI02] Tomomichi Sugihara, Yoshihiko Nakamura, and Hirochika Inoue. Realtime humanoid motion generation through zmp manipulation based on inverted pendulum control. In *Proceedings 2002 IEEE International*

Conference on Robotics and Automation (Cat. No. 02CH37292), volume 2, pages 1404–1409. IEEE, 2002.

[Tak89] Atsuo Takanishi. Dynamic biped walking stabilized with optimal trunk and waist motion. *Proceeding of IROS'89*, pages 187–192, 1989.

[TLTK90] Atsuo Takanishi, Hun-ok Lim, Masayuki Tsuda, and Ichiro Kato. Realization of dynamic biped walking stabilized by trunk motion on a sagittally uneven surface. In *IEEE International Workshop on Intelligent Robots and Systems, Towards a New Frontier of Applications*, pages 323–330. IEEE, 1990.

[TM19] Inc. The MathWorks. *Symbolic Math Toolbox.* Natick, Massachusetts, United State, 2019. URL: https://www.mathworks.com/help/symbolic/.

[Uch78] Masaru Uchiyama. Formation of high-speed motion pattern of a mechanical arm by trial. *Transactions of the Society of Instrument and Control Engineers*, 14(6):706–712, 1978.

[VS72] Miomir Vukobratović and J Stepanenko. On the stability of anthropomorphic systems. *Mathematical biosciences*, 15(1-2):1–37, 1972.

[WGDI09] Youqing Wang, Furong Gao, and Francis J Doyle III. Survey on iterative learning control, repetitive control, and run-to-run control. *Journal of Process Control*, 19(10):1589–1600, 2009.