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

Shengzhi Wang, George Mesesan, Johannes Englsberger, Dongheui Lee, Christian Ott

***Abstract*— We propose an online learning framework to reduce the effect of model inaccuracies and improve the robustness of the Divergent Component of Motion (DCM)-based walking algorithm. This framework uses the iterative learning control (ILC) theory for learning an adjusted Virtual Repellent Point (VRP) reference trajectory based on the current VRP error. The learned VRP reference waypoints are saved in a memory buffer and used in the subsequent walking iteration. Based on the availability of force-torque (FT) sensors, we propose two different implementations using different VRP error signals for learning: measurement-error-based and commanded-error-based framework. Both implementations reduce the average VRP errors and demonstrate improved walking robustness. The measurement-error-based framework has better reference trajectory tracking performance for the measured VRP.**

I. Introduction

Bipedal locomotion is a complex problem because the robots have nonlinear and hybrid system dynamics, and the contact stability constraints need to be fulfilled. Many studies used a mass concentrated model to reduce the complexity, focusing on the center of mass (CoM) dynamics. One of the most popular mass concentrated models is the linear inverted pendulum model (LIPM) [1]. Based on LIPM, the concept of the zero-moment point (ZMP) [2] was widely used in many early locomotion research for the bipedal robots [1], [3], [4]. However, the swing leg motion during walking and the model inaccuracies due to the inconsistency between the model and the real robot lead to an imperfect tracking of the reference trajectories. These tracking errors can cause the robot to fall if the ZMP reaches the edge of the support polygon. To prevent falling, feedback stabilization is widely used in many locomotion research [5], [6]. However, these balance methods may produce a persistent ZMP deviation from the reference trajectory.

Humans can learn a new skill or improve their motion performance by practicing repetitively and gaining experience from previous attempts. Inspired by human capabilities, Kawamura et al. [7] proposed a learning control algorithm for precise tracking of each joint reference trajectory. Yet, this method does not guarantee the robot walking stability as it does not use the concept of ZMP. In [8], a compensative trunk motion, which reduces the ZMP error, was learned iteratively from the ZMP error in the previous trial. However, this method is an offline learning framework for a complete walking trial. Contrary, Kajita et al. [4] proposed an online learning method: dynamic filter (DF), which adjusts the ZMP reference trajectory according to the current ZMP error prediction. Nevertheless, the DF method requires calculating the robot's multibody model and uses the preview control [9] to generate the CoM reference trajectory, which means it is model-based and computationally heavy. Hu et al. [10] proposed an Online Iterative Learning Control approach (ZMP-OILC) to learn a compensative ZMP trajectory (CZMP) based on the concept of iterative learning control (ILC). Compared with DF, ZMP-OILC is model-free, computationally lighter, and has better adaptation to constant external disturbances. Both DF and ZMP-OILC use the preview control to generate the CoM reference trajectory.

More recently, Englsberger et al. [11] presented the concepts of three dimensional Divergent Component of Motion (DCM) and Virtual Repellent Point (VRP) that decompose the second-order CoM dynamics into a first-order stable dynamics (CoM converges to DCM) and a first-order unstable dynamics (DCM diverges from VRP). Unlike ZMP, a 2-D point restricted to the surface of the ground, VRP is a 3-D point that is equivalent to the ZMP horizontally if its height above ground corresponds with the CoM average height Δz. This DCM-based algorithm generates a closed-form DCM and CoM reference trajectory based on the interpolation of a VRP waypoints sequence [12], which reduces the calculation load of the reference trajectory generation. Moreover, it has been shown that the DCM framework, combined with robust passivity based whole-body control algorithms, enables walking on uneven and compliant terrains [13].

Inspired by [10], [11], we aim to develop an online iterative learning control framework to reduce the VRP deviation caused by model inaccuracies for the DCM-based walking and call our framework VRP-OILC. We present two different implementations for the VRP-OILC depending on the availability of force-torque (FT) sensors in the feet: if the FT sensors are available, a measurement-error-based VRP-OILC will be used; otherwise, a commanded-error-based VRP-OILC will be applied. We demonstrate by simulations and experiments that both implementations yield improved walking robustness. Furthermore, the stability condition of VRP-OILC is analyzed. Compared with ZMP-OILC, our VRP-OILC uses the DCM-based algorithm, which is fully compatible with force-controlled robots and does not need any simplifying model assumptions as of the LIPM.

The paper is organized as follows. Section II introduces the DCM-based walking algorithm and the ILC theory. The

S. Wang and D. Lee are with the Department of Electrical and Computer Engineering, Technical University of Munich, Munich, Germany. Corresponding Author: S. Wang, email: shengzhi.wang@tum.de

All other authors are with the Institute of Robotics and Mechatronics, German Aerospace Center (DLR), Wessling, Germany.

D. Lee is also with Institute of Robotics and Mechatronics, German Aerospace Center (DLR).

framework design and the stability analysis are shown in Section III. In Section IV, the simulation and experimental results are evaluated. Finally, we conclude the paper in Section V.

II. Technical Background

A. Divergent Component of Motion Tracking Control

The concept of Divergent Component of Motion (DCM) and Virtual Repellent Point (VRP) can be found in [11]. Here, the DCM $\boldsymbol{\xi} \in \mathbb{R}^3$ is defined as:

$$\boldsymbol{\xi} = \boldsymbol{x} + b\,\dot{\boldsymbol{x}}, \tag{1}$$

where $\boldsymbol{x} \in \mathbb{R}^3$ is the CoM position, $\dot{\boldsymbol{x}} \in \mathbb{R}^3$ is the CoM velocity, $b = \sqrt{\Delta z / g}$ is the time constant of the DCM dynamics calculated by the CoM average height Δz and gravitational constant g.

According to [11], the total force $\boldsymbol{F} \in \mathbb{R}^3$ acting on the CoM of the robot can be encoded by the VRP $\boldsymbol{v} \in \mathbb{R}^3$ as:

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

where m is the robot's total mass. Differentiating (1), and inserting Newton's 2nd law $\ddot{\boldsymbol{x}} = \boldsymbol{F}/m$ with (2), the unstable DCM dynamics is obtained as:

$$\dot{\boldsymbol{\xi}} = \frac{1}{b}(\boldsymbol{\xi} - \boldsymbol{v}), \tag{3}$$

where the DCM $\boldsymbol{\xi}$ diverges away from the VRP.

A DCM-controller was introduced in [11], starting with the desired closed-loop dynamics:

$$\underbrace{\dot{\boldsymbol{\xi}} - \dot{\boldsymbol{\xi}}_d}_{\dot{\boldsymbol{e}}_\xi} = -k_\xi \underbrace{(\boldsymbol{\xi} - \boldsymbol{\xi}_d)}_{\boldsymbol{e}_\xi}, \tag{4}$$

where $\dot{\boldsymbol{\xi}}_d$ is the reference DCM velocity and $\boldsymbol{\xi}_d$ denotes the reference DCM position. Considering the dynamics in (3) the reference DCM trajectory implies a reference VRP $\boldsymbol{v}_d = \boldsymbol{\xi}_d - b\,\dot{\boldsymbol{\xi}}_d$. For DCM controller gains $k_\xi > 0$, $\boldsymbol{\xi}$ converges to $\boldsymbol{\xi}_d$ (asymptotically stable). The DCM control law can be expressed by inserting (3) and the DCM reference velocity into (4):

$$\boldsymbol{v} = \boldsymbol{v}_d + (1 + k_\xi b)\,(\boldsymbol{\xi} - \boldsymbol{\xi}_d), \tag{5}$$

where $\boldsymbol{v}$ is designed as a commanded VRP to realize the desired tracking behavior [11], i.e., $\boldsymbol{v} = \boldsymbol{v}_c$. The commanded VRP can be used to calculate a corresponding commanded force according to (2), to serve as an input to our passivity-based whole-body controller [13].

B. Reference Trajectory Generation

We split the total robot walking motion into a sequence of n_φ transition phases[1]. The reference trajectory calculation of VRP and DCM during a single transition phase φ is reported in this section. More details can be found in [14].

[1]Transition phase is a general term for single support (SS) and double support (DS).

VRP Trajectory: The VRP trajectory $\boldsymbol{v}_\varphi(t)$ for a transition phase local time t can be linearly interpolated between the start and end VRP waypoint ($\boldsymbol{v}_\varphi(0)$ and $\boldsymbol{v}_\varphi(T_\varphi)$) as:

$$\boldsymbol{v}_\varphi(t) = \left(1 - \frac{t}{T_\varphi}\right) \boldsymbol{v}_\varphi(0) + \frac{t}{T_\varphi}\,\boldsymbol{v}_\varphi(T_\varphi),\ t \in [0, T_\varphi] \tag{6}$$

where T_φ is a transition phase duration.

DCM Trajectory: The DCM trajectory is obtained by inserting (6) into (3) and then solving its differential equation:

$$\begin{aligned}\boldsymbol{\xi}_\varphi(t) = &\underbrace{\left(1 - \frac{t}{T_\varphi} - \frac{b}{T_\varphi} + e^{\frac{t-T_\varphi}{b}}\frac{b}{T_\varphi}\right)}_{\alpha_{\varphi,\xi}(t)} \boldsymbol{v}_\varphi(0) \\ &+ \underbrace{\left(\frac{t}{T_\varphi} + \frac{b}{T_\varphi} - e^{\frac{t-T_\varphi}{b}}(1 + \frac{b}{T_\varphi})\right)}_{\beta_{\varphi,\xi}(t)} \boldsymbol{v}_\varphi(T_\varphi) \\ &+ \underbrace{e^{\frac{t-T_\varphi}{b}}}_{\gamma_{\varphi,\xi}(t)}\,\boldsymbol{\xi}_\varphi(T_\varphi).\end{aligned} \tag{7}$$

The entire DCM trajectory terminates at the same position as the final VRP waypoint. From this terminal constraint, the whole DCM trajectory can be calculated backward recursively.

C. Measurement of Virtual Repellent Point

Based on Force-Torque (FT) Sensors: This method is suitable for robots with FT sensors in the feet. We assume no external forces act on the robot except for the gravity and contact force with the ground. Then, the measured total force $\boldsymbol{F}$ acting on the robot's CoM is expressed as:

$$\boldsymbol{F} = \sum_{p=L,R} \boldsymbol{R}^p_{CoM} \boldsymbol{F}_p - m\boldsymbol{g}, \tag{8}$$

where $\boldsymbol{R}^p_{CoM}$ is the rotation matrix from the CoM's coordinate frame to the FT sensor frame in the left or right foot, $\boldsymbol{F}_L \in \mathbb{R}^3$ or $\boldsymbol{F}_R \in \mathbb{R}^3$ denote the measured force from the left or right FT sensor, and $\boldsymbol{g} = [0, 0, -g]^T$ is the gravitational acceleration vector. Using (2) and (8), the measured VRP $\boldsymbol{v}_m$ is obtained as:

$$\boldsymbol{v}_m = \boldsymbol{x} - \frac{b^2}{m} \sum_{p=L,R} \boldsymbol{R}^p_{CoM} \boldsymbol{F}_p - b^2 \boldsymbol{g}. \tag{9}$$

The disadvantage of this measurement approach is that it is only applicable when there are no external forces except gravity and ground reaction force.

Based on DCM Dynamics Model: The measured VRP trajectory $\boldsymbol{v}_m$ can be obtained from the DCM dynamics (3) as:

$$\boldsymbol{v}_m = \boldsymbol{\xi} - b\,\dot{\boldsymbol{\xi}}, \tag{10}$$

where $\boldsymbol{\xi}$ is the measured DCM and $\dot{\boldsymbol{\xi}}$ is the measured DCM velocity. Equation (1) shows that the measured CoM position and velocity $\boldsymbol{x}$ and $\dot{\boldsymbol{x}}$ can be used to compute the measured DCM $\boldsymbol{\xi}$.

D. Iterative Learning Control

We assume there is an output tracking task with a control input u and output error $e = y_d - y$. According to [15], [16], the typical P-type ILC updating law can be expressed as:

$$u(t,k) = k_f u(t,k-1) + k_l e(t,k-1),\ t \in [0, T_{iter}] \quad (11)$$

where t is the local time moment which is reset to 0 at the beginning of a new iteration, k is the iteration index and T_{iter} is the iteration duration. The learning gain k_l and forgetting factor k_f have the following properties: $k_l \in [0, +\infty]$ and $k_f \in [0, 1]$.

III. Online Iterative Learning Control of VRP

As bipedal walking is a repetitive locomotion behavior, we can use ILC to improve the DCM-based walking algorithm. Our proposed VRP-OILC framework defines one iteration as one walking cycle, which starts with the first foot, goes to the second foot, and ends back on the first foot. It means that 2 SS phases and 2 DS phases are included in one iteration. Specifically, the learning iteration number starts from the 0^{th} iteration. Fig. 1 shows a bipedal walking in $(i-1)^{th}$ and i^{th} iteration, where $(i-1)^{th}$ iteration is considered as the current iteration. Here, $\boldsymbol{G}_0$ 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, and the angle $\Delta\alpha$ represents the rotation from frame $\boldsymbol{G}_{i-1}$ to frame $\boldsymbol{G}_i$ along the z-axis. The rotation matrix $\boldsymbol{R}_\Delta$ from $\boldsymbol{G}_{i-1}$ to $\boldsymbol{G}_i$ can be expressed as:

$$\boldsymbol{R}_\Delta = \boldsymbol{R}_Z(\Delta\alpha) = \begin{bmatrix} \cos\Delta\alpha & -\sin\Delta\alpha & 0 \\ \sin\Delta\alpha & \cos\Delta\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix}. \quad (12)$$

Note that the stepping behavior is only repetitive in the local coordinate frame of each iteration. Therefore, the rotation matrix $\boldsymbol{R}_\Delta$ will be used in our ILC framework design to transform the learned information between the local frames.

The rest of this section will propose two different versions for the learning framework design: learning based on the VRP measurement and commanded error. For robots in which the FT sensors are available, the measurement-error-based framework should be chosen. In general, the framework based on the VRP measured error is preferable because the goal of using ILC is to enhance the walking robustness by bringing the measured VRP closer to the desired trajectory. When the FT sensors are not available and the VRP measurement based on the DCM model is not applicable due to signal noise, the commanded-error-based framework can be used as an alternative solution.

A. Framework Design Based on Measurement Error

Our VRP-OILC is a framework using ILC to learn the future VRP reference trajectory based on the current VRP error. The learned trajectory will be saved in a waypoint list and reused in the next iteration. This method guarantees the continuity of the adjusted reference VRP and DCM trajectory, stabilizing the whole robot system during walking.

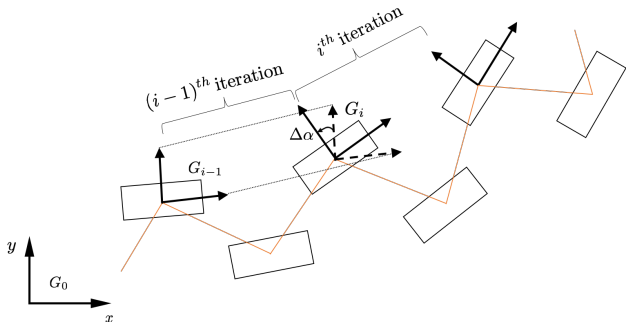

Fig. 1: An example of two walking iterations.

Definition 1 - 3 are essential for the elaboration of the framework:

Definition 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 2: (Learned VRP Waypoint) A learned VRP waypoint is a discrete reference VRP point adjusted using the ILC updating law.

Definition 3: (ILC Update Interval) An ILC update interval Δt_{ILC} is a time interval between two ILC update cycles. For example, our VRP-OILC framework uses $\Delta t_{ILC} = 0.01$ s. Note that the ILC update interval cannot be smaller than the system sampling time.

We use a learned VRP waypoint list as a buffer to save the learned VRP waypoints and then calculate the adjusted VRP and DCM reference trajectory. Algorithm 1 shows the details of the framework, where the whole process shown in this algorithm will be executed at each sample time step from the start of the initial iteration to the termination of the program:

• *Lines 1 - 2*: Calculation of the iteration index i and the iteration local time t. Here, t_g is the global time elapsed after the beginning of the initial iteration. The floor operator $\lfloor\cdot\rfloor$ and remainder operator $\mathbf{rem}$ are used to calculate the iteration index and the local time.

• *Lines 3 - 6*: Initialization of the learned VRP waypoint list $\boldsymbol{V}_l$ (i.e., when $i = 0$ and $t = 0$) by sampling the 0^{th} iteration's VRP original reference trajectory with a sample time Δt_{ILC}. The first waypoint of the $\boldsymbol{V}_l$ (i.e., $\boldsymbol{v}_{d,0}(0)$) is selected as the current adjusted VRP reference point $\boldsymbol{v}_{l,i}(0)$ and the original DCM reference point $\boldsymbol{\xi}_{d,0}(0)$ is treated as the current adjusted DCM reference point $\boldsymbol{\xi}_{l,i}(0)$. In short, the adjusted VRP and DCM reference trajectory are the same as their original reference trajectory at $t = 0$ and $i = 0$.

• *Lines 7 - 26*: These lines describe the trajectory adaptation case. The learned waypoint index k and the elapsed local time $t_{in\_w} \in [0, \Delta t_{ILC})$ are calculated in lines 8 - 9. Given the ILC updating interval Δt_{ILC} and the system sampling time Δt_s, the waypoint index k and $k+1$ determine the corresponding two learned waypoints for the current position, and t_{in_w} denotes the time elapsed since the last ILC update (or the local time between the current and next waypoint). For example, if $i = 1$, $t = 0.034$ s, $\Delta t_{ILC} = 0.01$ s and

Algorithm 1: Pseudocode of VRP-OILC

Input: Step Information.
Output: Adjusted VRP and DCM Trajectory ($\boldsymbol{v}_{l,i}(t)$ and $\boldsymbol{\xi}_{l,i}(t)$).

1 $i \leftarrow \lfloor t_g / T_{iter} \rfloor$
2 $t \leftarrow t_g$ **rem** T_{iter}
3 **if** $i = 0$ *and* $t = 0$ **then**
4 Initialize $\boldsymbol{V}_l$: $\boldsymbol{V}_l \leftarrow [\boldsymbol{v}_{d,0}(0),\ \boldsymbol{v}_{d,0}(\Delta t_{ILC}),\ \ldots,\ \boldsymbol{v}_{d,0}(T_{iter} - \Delta t_{ILC})]$
5 $\boldsymbol{v}_{l,i}(t) \leftarrow \boldsymbol{V}_l(1)$
6 $\boldsymbol{\xi}_{l,i}(t) \leftarrow \boldsymbol{\xi}_{d,0}(0)$
7 **else**
8 $k \leftarrow \lfloor t / \Delta t_{ILC} \rfloor$
9 $t_{in\_w} \leftarrow t$ **rem** $k \Delta t_{ILC}$
10 **if** $t_{in\_w} = 0$ **then**
11 **if** $\varphi_c + 4 \leq n_\varphi$ **then**
12 $\boldsymbol{v}_{l,i}(t) \leftarrow \boldsymbol{V}_l(2)$
13 $\boldsymbol{v}_{l,i+1}(t) \leftarrow \boldsymbol{v}_{d,i+1}(t) + k_f \boldsymbol{R}_\Delta (\boldsymbol{v}_{l,i}(t) - \boldsymbol{v}_{d,i}(t)) + k_l \boldsymbol{R}_\Delta (\boldsymbol{v}_{d,i}(t) - \boldsymbol{v}_{m,i}(t))$
14 $\boldsymbol{V}_l' \leftarrow [\boldsymbol{V}_l(2:end),\ \boldsymbol{v}_{l,i+1}(t)]$
15 **else**
16 $\boldsymbol{V}_l' \leftarrow [\boldsymbol{V}_l(2:end),\ \boldsymbol{V}_l(end)]$
17 $\boldsymbol{V}_l \leftarrow \boldsymbol{V}_l'$
18 $\boldsymbol{v}_{l,i}(t) \leftarrow (1 - f_{ILC}(t_{in\_w}))\, \boldsymbol{V}_l(1) + f_{ILC}(t_{in\_w}) \boldsymbol{V}_l(2)$
19 **if** $\varphi_c + 4 \leq n_\varphi$ **then**
20 $\boldsymbol{\xi}_f \leftarrow \boldsymbol{\xi}_{d,i+1}(k \Delta t_{ILC})$
21 **else**
22 $\boldsymbol{\xi}_f \leftarrow \boldsymbol{V}_l(end)$
23 $\boldsymbol{\xi}_{mid} \leftarrow \boldsymbol{\xi}_f$
24 **for** $j \leftarrow \frac{T_{iter}}{\Delta t_{ILC}} : -1 : 3$ **do**
25 $\boldsymbol{\xi}_{mid} \leftarrow \alpha_{ILC,\xi}(0) \boldsymbol{V}_l(j-1) + \beta_{ILC,\xi}(0) \boldsymbol{V}_l(j) + \gamma_{ILC,\xi}(0) \boldsymbol{\xi}_{mid}$
26 $\boldsymbol{\xi}_{l,i}(t) \leftarrow \alpha_{ILC,\xi}(t_{in\_w}) \boldsymbol{V}_l(1) + \beta_{ILC,\xi}(t_{in\_w}) \boldsymbol{V}_l(2) + \gamma_{ILC,\xi}(t_{in\_w}) \boldsymbol{\xi}_{mid}$
27 $t_g \leftarrow t_g + \Delta t_s$

$\Delta t_s = 0.001$ s, then $k = 3$ and $t_{in\_w} = 0.004$ s, which means that the current adjusted VRP reference point is on the interpolation 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 0.004 s have elapsed after the VRP passed the 3^{rd} waypoint (i.e., the last ILC update).

Lines 10 - 17 describe the update process of the learned VRP waypoint list, i.e., at $t_{in\_w} = 0$ s. Specifically, lines 11 - 14 show the learning process of the waypoint list when the robot does not stop walking in the next iteration (i.e., within the next 4 transition phases, which can be represented as $\varphi_c + 4 \leq n_\varphi$, with φ_c is the current transition phase index and n_φ is the total transition phase's amount). In line 13, the ILC updating law of VRP-OILC is applied to adjust the next iteration's reference VRP trajectory waypoint $\boldsymbol{v}_{l,i+1}(t)$. As shown in line 14, this learned VRP future reference waypoint is placed at the end of the waypoint list, and the first waypoint of the waypoint list is deleted so that the learned VRP waypoint list contains a fixed number of learned waypoints and acts like a sliding window. This ensures that the learned future waypoint would be used again for the adjusted reference trajectories calculation at the same local time in the next iteration and the adjusted VRP reference trajectory is continuous. When the robot stops walking within the next four transition phases, the ILC updating law is not applied, and the last waypoint of the waypoint list is duplicated, as shown in line 16.

After the waypoint list update, the current adjusted VRP reference trajectory $\boldsymbol{v}_{l,i}(t)$ is obtained by the linear interpolation between $\boldsymbol{V}_l(1)$ and $\boldsymbol{V}_l(2)$ in line 18, where Δt_{ILC} is the time duration for the polynomial $f_{ILC}(\cdot)$. Lines 19 - 26 show the calculation of the current adjusted DCM reference trajectory $\boldsymbol{\xi}_{l,i}(t)$. Particularly, the terminal DCM point is determined in lines 19 - 22 depending on whether the robot stops walking within the next four transition phases, while lines 23 - 26 use all the learned VRP waypoints in $\boldsymbol{V}_l$ and the DCM trajectory calculation (7) to calculate the $\boldsymbol{\xi}_{l,i}(t)$ from the terminal DCM backward recursively. Similar to $f_{ILC}(\cdot)$, the $\alpha_{ILC,\xi}(\cdot)$, $\beta_{ILC,\xi}(\cdot)$ and $\gamma_{ILC,\xi}(\cdot)$ in lines 25 - 26 use the Δt_{ILC} as time duration. Finally, the VRP-OILC outputs the $\boldsymbol{v}_{l,i}(t)$ and $\boldsymbol{\xi}_{l,i}(t)$ as the adjusted reference trajectories to the DCM-controller. Line 27 implies that the system enters the next sample time step.

B. Framework Design Based on Commanded Error

Similar to the definition of VRP measured error, the commanded error is defined as:

Definition 4: (VRP Commanded Error) A VRP commanded error is the difference between the desired and commanded VRP, i.e., $\boldsymbol{v}_d - \boldsymbol{v}_c$.

In the commanded-error-based VRP-OILC, the ILC updating law in line 13 of Algorithm 1 becomes:

$$\begin{aligned}\boldsymbol{v}_{l,i+1}(t - \Delta t_s) = \boldsymbol{v}_{d,i+1}(t - \Delta t_s) \\ + k_f \boldsymbol{R}_\Delta (\boldsymbol{v}_{l,i}(t - \Delta t_s) - \boldsymbol{v}_{d,i}(t - \Delta t_s)) \\ + k_l \boldsymbol{R}_\Delta (\boldsymbol{v}_{d,i}(t - \Delta t_s) - \boldsymbol{v}_{c,i}(t - \Delta t_s)).\end{aligned} \tag{13}$$

Note that (13) uses VRP quantities from the last time step (i.e., from $t - \Delta t_s$) because the current time step's commanded VRP is only obtained by the DCM-controller after the adjusted VRP and DCM reference trajectory are calculated. Moreover, the learned future waypoint $\boldsymbol{v}_{l,i+1}(t)$ in line 14 of Algorithm 1 is replaced by $\boldsymbol{v}_{l,i+1}(t - \Delta t_s)$ correspondingly. These are the only differences between the commanded-error-based and measurement-error-based VRP-OILC.

C. Convergence Condition

Since the goal of VRP-OILC is to improve the walking robustness by reducing the VRP deviation while tracking the reference trajectory, the framework's performance can be

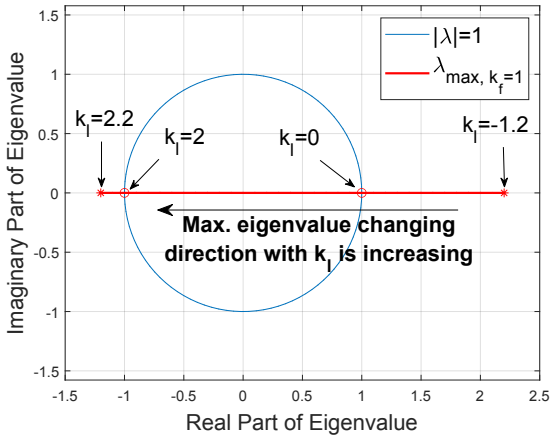

Fig. 2: Maximum eigenvalue locus with $k_f = 1$, $T_{iter} = 2.4$ s, $b \approx 0.3112$, $k_\xi = 4$, and k_l varying from -1.2 to 2.2.

represented by evaluating the convergence of the VRP trajectory. Inspired by [10], the VRP trajectory convergence can be further evaluated by the average VRP error during each iteration. Here, we define two average VRP errors: average measured and commanded error. The average measured error $e_{m,i}$ during the i^{th} iteration is denoted as:

$$e_{m,i} = \frac{1}{T_{iter}} \int_0^{T_{iter}} |\boldsymbol{v}_{d,i}(t) - \boldsymbol{v}_{m,i}(t)|\, dt. \tag{14}$$

The average commanded VRP error $e_{c,i}$ is expressed as:

$$e_{c,i} = \frac{1}{T_{iter}} \int_0^{T_{iter}} |\boldsymbol{v}_{d,i}(t) - \boldsymbol{v}_{c,i}(t)|\, dt. \tag{15}$$

The measurement-error-based framework uses the average measured error, while the commanded-error-based framework uses the average commanded error as a convergence criterion, respectively. Both frameworks' learning process is considered to converge if the average error difference between two iterations is small enough.

D. Stability Analysis

To analyze the stability of the VRP-OILC, we simplify the walking model by assuming there is only one waypoint per iteration, i.e., one VRP, DCM and CoM waypoint respectively, and considering only straightforward walking. The notation of the waypoints is also simplified by omitting the local time "t" inside the bracket, e.g., we use $\boldsymbol{v}_{d,i}$ to denote the i^{th} iteration's desired VRP waypoint. These waypoints are the waypoints at local time moment $t = 0$ s. Furthermore, we build a state-space representation of the measurement-error-based VRP-OILC as:

$$\begin{bmatrix} \boldsymbol{\xi}_{i+1} \\ \boldsymbol{v}_{l,i+1} \end{bmatrix} = \underbrace{\begin{bmatrix} a_1 & a_2 \\ a_3 & a_4 \end{bmatrix}}_{A} \begin{bmatrix} \boldsymbol{\xi}_i \\ \boldsymbol{v}_{l,i} \end{bmatrix} + \begin{bmatrix} a_5 \boldsymbol{d}(0) - \frac{\boldsymbol{D}(T)}{b} \\ -k_l \boldsymbol{d}(0) \end{bmatrix} + \begin{bmatrix} \boldsymbol{FFT}_4 \\ \boldsymbol{FFT}_1 \end{bmatrix}, \tag{16}$$

where $\boldsymbol{d}$ is the VRP disturbance error and $\boldsymbol{D}$ is the integration of the VRP disturbance error. The details about the derivations can be seen in Appendix V-A. Moreover, the state-space model of the commanded-error-based VRP-OILC is derived in Appendix V-B. Both models have the same state matrix A, whose entries are time-invariant constant values, so these models' stability can be analyzed via the eigenvalues of A.

Fig. 2 shows the maximum absolute eigenvalue locus (the red line) of the matrix A for fixed forgetting factor k_f and different learning gain k_l. The maximum eigenvalue is inside the unit circle when $|k_f - k_l| < 1$. This is the VRP-OILC's stability condition, which is the same as the one shown in [10]. Note that the presented stability analysis can be extended to an arbitrary number of waypoints. The same stability condition can be obtained from the analysis.

IV. Result Evaluation

The proposed VRP-OILC is implemented on the DLR humanoid robot TORO [17], which has a height of 1.74 m and a weight of 77.5 kg. The robot's whole control loop is executed at a sampling frequency of 1 kHz. For the VRP-OILC setup, four important design decisions were taken as follows:

1) We only used the 2-dimensional VRP-OILC framework for the learning, which means that the framework will not adjust the reference trajectory in Z direction. The reason is: we only concern about whether the VRP stays inside the support polygon in the X and Y direction.

2) To avoid problems caused by measurement noise, such as a jerky learned VRP trajectory, we set the ILC update interval Δt_{ILC} to 0.01 s for all simulations and experiments.

3) The measurement-error-based framework, which measured the VRP by FT sensors, was evaluated in simulation. Since the FT sensors were not available on the real robot and the measured VRP based on the DCM model is noisy, we used a commanded-error-based framework in the experiments.

4) To suppress the discontinuity of VRP measurement during the foot landing and lifting, we applied a variable forgetting factor method during the simulations and experiments inspired by [10]: the k_f remained to 1 in SS phases and changed to 0.5 in DS phases.

A. Simulation

The simulations were conducted in OpenHRP3 [18]. To simulate model inaccuracies, we changed the robot link masses while maintaining the total mass to 77.5 kg. The upper and middle graphs in Fig. 3 compares the measured VRP trajectory with and without using the VRP-OILC (i.e., $\boldsymbol{v}_{m,ILC}$ and $\boldsymbol{v}_{m,noILC}$, respectively) for straightforward walking. The zoomed-in areas show that the measured VRP converged to the desired trajectory after using VRP-OILC. The convergence of the measured VRP can also be observed in the lower graph of Fig. 3, where the average measured VRP error $e_{m,\,ILC}$ of VRP-OILC decreased when compared with the constant average error $e_{m,\,noILC}$ without using our framework, from about 7 mm to 2 mm.

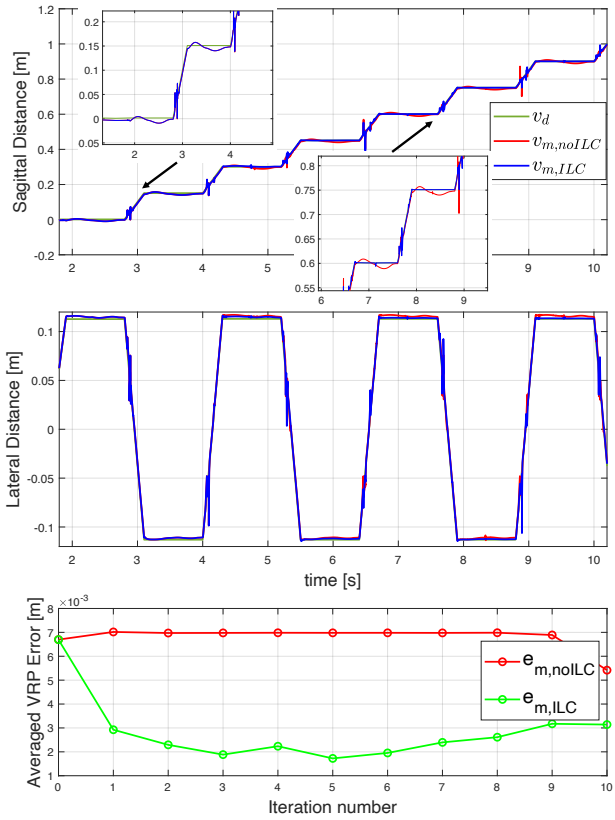

Fig. 3: Simulation results of a straightforward walking with SS time $T_{SS} = 0.9$ s, DS time $T_{DS} = 0.3$ s, and learning parameter $k_l = 1$. The upper and middle graphs are the VRP plots in sagittal and lateral direction respectively. The lower graph shows the average VRP measured error with and without using our framework.

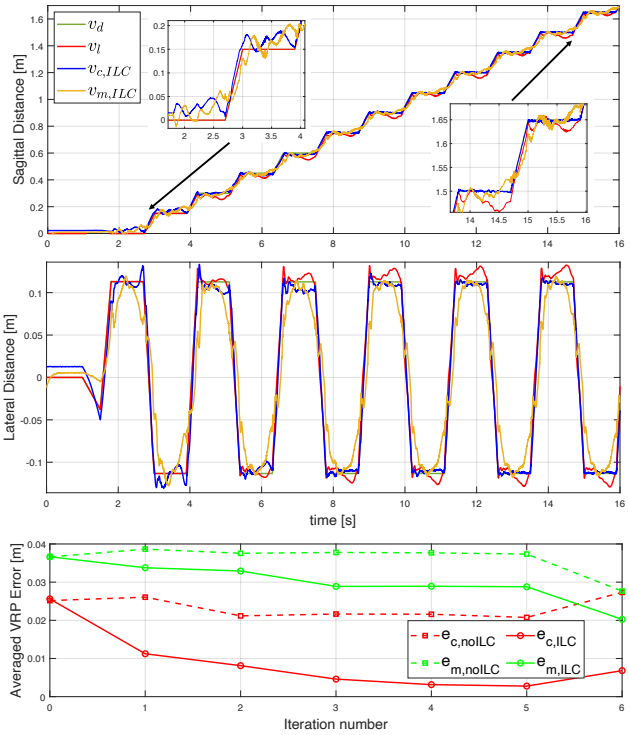

Fig. 4: Experimental results of a straightforward walking with T_{SS}/T_{DS}: 0.9 s/0.3 s, sagittal and lateral learning gain $k_{l,sa}/k_{l,la}$: 1/0.5. The upper and middle graphs are the VRP plots of applying VRP-OILC, and the lower graph shows the measured and commanded VRP average error with and without using our framework.

B. Experiments

Fig. 4 shows the experimental results of applying commanded-error-based VRP-OILC for straightforward walking. The commanded VRP trajectory $\boldsymbol{v}_{c,ILC}$ converged to the desired trajectory successfully by tracking the learned VRP reference trajectory $\boldsymbol{v}_l$, as shown in the upper and middle graphs. Also, the measured VRP trajectory $\boldsymbol{v}_{m,ILC}$, a filtered measurement based on the DCM model, was closer to the desired trajectory. The lower graph of Fig. 4 illustrates the convergence of the commanded and measured VRP: the average commanded error $\mathrm{e_{c,ILC}}$ decreased from 25 mm to less than 5 mm and the average measured error $\mathrm{e_{m,ILC}}$ was reduced from around 40 mm to 30 mm.

Moreover, our VRP-OILC achieved fast walking with $T_{SS} = 0.7$ s and $T_{DS} = 0.2$ s, which was impossible without the proposed framework. The upper graph in Fig. 5 shows that the robot fell at $t = 4.6$ s because the effect of the model inaccuracies increased the VRP tracking error eventually leading to the robot falling. The second and third graphs in the middle in Fig. 5 illustrates a successful fast walking by applying VRP-OILC, where the zoomed-in areas indicate that the measured VRP $\boldsymbol{v}_{m,ILC}$ and the commanded VRP $\boldsymbol{v}_{c,ILC}$ were closer to the desired trajectory. The convergence of $\boldsymbol{v}_{m,ILC}$ and $\boldsymbol{v}_{c,ILC}$ can also be demonstrated in the lower graph of Fig. 5, where the average commanded error $\mathrm{e_{c,ILC}}$ was reduced from 30 mm to less than 10 mm and the average measured error $\mathrm{e_{m,ILC}}$ decreased from 50 mm to around 40 mm.

V. Conclusions

This paper proposes an online VRP adaptation framework for improving the robustness of the DCM-based biped walking. Our framework can be divided into two different implementations: measurement-error-based and commanded-error-based framework. The former learns an adjusted VRP reference trajectory from the measured VRP error, while the latter uses the commanded VRP error when FT sensors are not available. Both implementations use the ILC for the learning process. The enhanced walking robustness was demonstrated through the reduced average VRP error and increased achievable walking speed.

For future work, we plan to test the measured-error-based VRP-OILC on TORO. Additionally, we will study different methods to improve the robustness of walking in complex environments, where the external disturbances are non-repetitive.

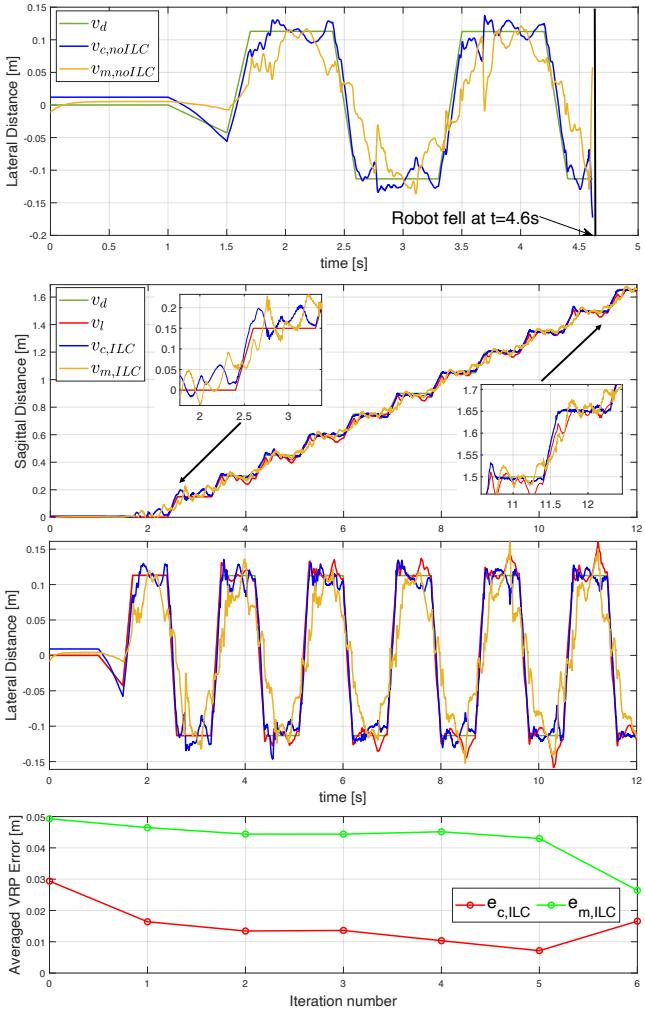

Fig. 5: Experimental results of a straightforward fast walking with T_{SS}/T_{DS}: 0.7 s/0.2 s, sagittal and lateral learning gain $k_{l,sa}/k_{l,la}$: 1/0.5. The upper graph is the lateral VRP plot without using VRP-OILC. The second and third graphs in the middle are the VRP plots of applying VRP-OILC. The lower graph shows the measured and commanded VRP average error of using our framework.

APPENDIX

A. Derivation of Measurement-Error-Based VRP-OILC's State-Space Model

We assume the i^{th} iteration is the current iteration ($i \geq 0$). To distinguish the waypoint duration T (i.e., the duration between two waypoints) from the iteration duration T_{iter}, the relation between them is $T_{iter} = n_w T$, where n_w is the waypoint number in each iteration. In Section III-D, n_w equals 1 because there is only one waypoint per iteration. Moreover, the waypoint is assumed to be at the local time moment of $t = 0$ s.

• *Calculation of* $\boldsymbol{v}_{l,i+1}$: As mentioned in line 26 of Algorithm 1, the learned DCM is calculated as:

$$\boldsymbol{\xi}_{l,i} = \alpha\, \boldsymbol{v}_{l,i} + \beta\, \boldsymbol{v}_{d,i+1} + \gamma\, \boldsymbol{\xi}_{d,i+1}. \tag{17}$$

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

$$\boldsymbol{v}_{c,i} = \boldsymbol{v}_{l,i} + (1 + k_\xi b)\,(\boldsymbol{\xi}_i - \boldsymbol{\xi}_{l,i})\,. \tag{18}$$

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

$$\boldsymbol{v}_{l,i+1} = \boldsymbol{v}_{d,i+1} + k_f\,(\boldsymbol{v}_{l,i} - \boldsymbol{v}_{d,i}) + k_l\,(\boldsymbol{v}_{d,i} - \boldsymbol{v}_{m,i})\,. \tag{19}$$

When model inaccuracies exist, there is a force error $\Delta\boldsymbol{F}$ between the CoM total force $\boldsymbol{F}$ and the commanded force $\boldsymbol{F}_c$, which can be modeled as $\Delta\boldsymbol{F}(t) = \boldsymbol{F}(t) - \boldsymbol{F}_c(t)$. The $\boldsymbol{F}$ and $\boldsymbol{F}_c$ can be further derived to $\boldsymbol{v}_m$ and $\boldsymbol{v}_c$ based on (2). Therefore, the VRP disturbance error is modeled as:

$$\boldsymbol{d}(0) = \boldsymbol{v}_{m,i} - \boldsymbol{v}_{c,i}. \tag{20}$$

Inserting (17), (18) and (20) into the ILC updating law (19) yields

$$\begin{aligned}\boldsymbol{v}_{l,i+1} &= \boldsymbol{v}_{d,i+1} + k_f\boldsymbol{v}_{l,i} - k_f\boldsymbol{v}_{d,i} + k_l\boldsymbol{v}_{d,i} - k_l\boldsymbol{v}_{l,i} \\ &\quad - k_l\underbrace{(1 + k_\xi b)}_{\zeta}(\boldsymbol{\xi}_i - \boldsymbol{\xi}_{l,i}) - k_l\boldsymbol{d}(0) \\ &= \underbrace{[k_f + k_l(\zeta\alpha - 1)]}_{a_4}\boldsymbol{v}_{l,i} + \underbrace{(-k_l\zeta)}_{a_3}\boldsymbol{\xi}_i \\ &\quad - k_l\boldsymbol{d}(0) + \boldsymbol{F}\boldsymbol{F}\boldsymbol{T}_1,\end{aligned} \tag{21}$$

where $\boldsymbol{F}\boldsymbol{F}\boldsymbol{T}_1$ is a feedforward term denoting all the original desired quantities (e.g., $\boldsymbol{v}_{d,i+1}$ and $\boldsymbol{\xi}_{d,i+1}$), and $\boldsymbol{\xi}_i$ is the measured DCM position considered as a system state.

• *Calculation of* $\boldsymbol{\xi}_{i+1}$: A discontinuity of the $\boldsymbol{\xi}_{l,i}$ is caused by adjusting the next iteration's reference VRP and DCM point from their original desired waypoints to the learned waypoints. The calculation of $\boldsymbol{\xi}_{l,i+1}$ is expressed by substituting $i + 1$ for i in (17):

$$\boldsymbol{\xi}_{l,i+1} = \alpha\,\boldsymbol{v}_{l,i+1} + \boldsymbol{F}\boldsymbol{F}\boldsymbol{T}_2. \tag{22}$$

After the ILC updating, (17) becomes the following equation based on (21) and (22):

$$\begin{aligned}\boldsymbol{\xi}^*_{l,i} &= [\alpha + (\beta + \gamma\alpha)\,a_4]\,\boldsymbol{v}_{l,i} + (\beta + \gamma\alpha)\,a_3\boldsymbol{\xi}_i \\ &\quad - k_l\,(\beta + \gamma\alpha)\,\boldsymbol{d}(0) + \boldsymbol{F}\boldsymbol{F}\boldsymbol{T}_3,\end{aligned} \tag{23}$$

where $\boldsymbol{\xi}^*_{l,i}$ is the learned DCM waypoint at the i^{th} iteration after the ILC learning.

To find the relation between $\boldsymbol{\xi}_{i+1}$ and $\boldsymbol{\xi}_i$, it is necessary to start from the DCM dynamics. The continuous-time DCM dynamics in i^{th} iteration is expanded by inserting the (3), (18) and (20):

$$\begin{aligned}\dot{\boldsymbol{\xi}}_i(t) &= \frac{1}{b}\,(\boldsymbol{\xi}_i(t) - \boldsymbol{v}_{m,i}(t)) \\ &= \frac{1}{b}\,(\boldsymbol{\xi}_i(t) - \boldsymbol{v}_{c,i}(t) - \boldsymbol{d}(t)) \\ &= \dot{\boldsymbol{\xi}}_{l,i}(t) - k_\xi\,(\boldsymbol{\xi}_i(t) - \boldsymbol{\xi}_{l,i}(t)) - \frac{\boldsymbol{d}(t)}{b}.\end{aligned} \tag{24}$$

Moving the $\dot{\boldsymbol{\xi}}_{l,i}(t)$ to the left side of (24), a DCM closed-loop dynamics with the disturbance term can be obtained as:

$$\underbrace{\dot{\boldsymbol{\xi}}_i(t) - \dot{\boldsymbol{\xi}}_{l,i}(t)}_{\dot{\boldsymbol{\epsilon}}_i(t)} = -k_\xi\,\underbrace{(\boldsymbol{\xi}_i(t) - \boldsymbol{\xi}_{l,i}(t))}_{\boldsymbol{\epsilon}_i(t)} - \frac{\boldsymbol{d}(t)}{b}, \tag{25}$$

where the DCM error $\boldsymbol{\epsilon}_i(t)$ can be derived by solving (25):

$$\begin{aligned}\underbrace{\boldsymbol{\xi}_i(t) - \boldsymbol{\xi}_{l,i}(t)}_{\boldsymbol{\epsilon}_i(t)} &= e^{-k_\xi t}\left[\underbrace{\boldsymbol{\xi}_i(0) - \boldsymbol{\xi}^*_{l,i}(0)}_{\boldsymbol{\epsilon}_i(0)} - \frac{1}{b}\int_0^t e^{k_\xi \tau}\boldsymbol{d}(\tau)d\tau\right] \\ &= e^{-k_\xi t}\,\boldsymbol{\xi}_i - e^{-k_\xi t}\,\boldsymbol{\xi}^*_{l,i} \\ &\quad - \frac{1}{b}\underbrace{e^{-k_\xi t}\int_0^t e^{k_\xi \tau}\boldsymbol{d}(\tau)d\tau}_{\boldsymbol{D}(t)}.\end{aligned} \tag{26}$$

Inserting (21), (22) and (23) into (26), the relation between $\boldsymbol{\xi}_{i+1}$, $\boldsymbol{\xi}_i$ and $\boldsymbol{v}_{l,i}$ can be found as:

$$\begin{aligned}\boldsymbol{\xi}_{i+1} = \boldsymbol{\xi}_i(T) &= \boldsymbol{\xi}_{l,i}(T) + e^{-k_\xi T}\,\boldsymbol{\xi}_i - e^{-k_\xi T}\,\boldsymbol{\xi}^*_{l,i} - \frac{\boldsymbol{D}(T)}{b} \\ &= \boldsymbol{\xi}_{l,i+1} + e^{-k_\xi T}\,\boldsymbol{\xi}_i - e^{-k_\xi T}\,\boldsymbol{\xi}^*_{l,i} - \frac{\boldsymbol{D}(T)}{b} \\ &= \underbrace{\left[\alpha a_4 - e^{-k_\xi T}\left[\alpha + a_4\left(\beta + \gamma\alpha\right)\right]\right]}_{a_2}\boldsymbol{v}_{l,i} \\ &\quad + \underbrace{\left[e^{-k_\xi T} + \alpha a_3 + e^{-k_\xi T}k_l\zeta\left(\beta + \gamma\alpha\right)\right]}_{a_1}\boldsymbol{\xi}_i \\ &\quad + \underbrace{\left[e^{-k_\xi T}k_l(\beta + \gamma\alpha) - \alpha k_l\right]}_{a_5}\boldsymbol{d}(0) \\ &\quad - \frac{\boldsymbol{D}(T)}{b} + \boldsymbol{FFT}_4.\end{aligned} \tag{27}$$

Finally, the state-space model is built based on (21) and (27) as:

$$\begin{aligned}\begin{bmatrix}\boldsymbol{\xi}_{i+1}\\ \boldsymbol{v}_{l,i+1}\end{bmatrix} &= \underbrace{\begin{bmatrix}a_1 & a_2\\ a_3 & a_4\end{bmatrix}}_{A}\begin{bmatrix}\boldsymbol{\xi}_i\\ \boldsymbol{v}_{l,i}\end{bmatrix} \\ &\quad + \begin{bmatrix}a_5\boldsymbol{d}(0) - \frac{\boldsymbol{D}(T)}{b}\\ -k_l\boldsymbol{d}(0)\end{bmatrix} + \begin{bmatrix}\boldsymbol{FFT}_4\\ \boldsymbol{FFT}_1\end{bmatrix},\end{aligned} \tag{28}$$

B. Derivation of Commanded-Error-Based VRP-OILC's State-Space Model

In this framework, the ILC updating law becomes:

$$\boldsymbol{v}_{l,i+1} = \boldsymbol{v}_{d,i+1} + k_f\left(\boldsymbol{v}_{l,i} - \boldsymbol{v}_{d,i}\right) + k_l\left(\boldsymbol{v}_{d,i} - \boldsymbol{v}_{c,i}\right). \tag{29}$$

Note that (29) is exactly the same as (21) without the disturbance term $\boldsymbol{d}(0)$. Therefore, (29) can be rewritten as:

$$\boldsymbol{v}_{l,i+1} = a_4\boldsymbol{v}_{l,i} + a_3\boldsymbol{\xi}_i + \boldsymbol{FFT}_1, \tag{30}$$

where the disturbance term $\boldsymbol{d}(0)$ disappears in (30). Therefore, the calculation of $\boldsymbol{\xi}_{i+1}$ can be obtained by following the same derivation as for (22) - (27) as:

$$\boldsymbol{\xi}_{i+1} = a_2\boldsymbol{v}_{l,i} + a_1\boldsymbol{\xi}_i - \frac{\boldsymbol{D}(T)}{b} + \boldsymbol{FFT}_4. \tag{31}$$

From (30) and (31), the state-space model is:

$$\begin{aligned}\begin{bmatrix}\boldsymbol{\xi}_{i+1}\\ \boldsymbol{v}_{l,i+1}\end{bmatrix} &= \underbrace{\begin{bmatrix}a_1 & a_2\\ a_3 & a_4\end{bmatrix}}_{A}\begin{bmatrix}\boldsymbol{\xi}_i\\ \boldsymbol{v}_{l,i}\end{bmatrix} \\ &\quad + \begin{bmatrix}-\frac{\boldsymbol{D}(T)}{b}\\ \mathbf{0}\end{bmatrix} + \begin{bmatrix}\boldsymbol{FFT}_4\\ \boldsymbol{FFT}_1\end{bmatrix}.\end{aligned} \tag{32}$$

ACKNOWLEDGMENT

This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement No. 819358).

REFERENCES

[1] T. Sugihara, Y. Nakamura, and H. Inoue, "Real-time humanoid motion generation through zmp manipulation based on inverted pendulum control," in *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No. 02CH37292)*, vol. 2. IEEE, 2002, pp. 1404–1409.

[2] M. Vukobratović and J. Stepanenko, "On the stability of anthropomorphic systems," *Mathematical biosciences*, vol. 15, no. 1-2, pp. 1–37, 1972.

[3] T. Sugihara, "Standing stabilizability and stepping maneuver in planar bipedalism based on the best com-zmp regulator," in *2009 IEEE International Conference on Robotics and Automation*. IEEE, 2009, pp. 1966–1971.

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

[5] S. Kajita, M. Morisawa, K. Miura, S. Nakaoka, K. Harada, K. Kaneko, F. Kanehiro, and K. Yokoi, "Biped walking stabilization based on linear inverted pendulum tracking," in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2010, pp. 4489–4496.

[6] Y. Choi, D. Kim, Y. Oh, and B.-J. You, "Posture/walking control for humanoid robot based on kinematic resolution of com jacobian with embedded motion," *IEEE Transactions on Robotics*, vol. 23, no. 6, pp. 1285–1293, 2007.

[7] S. Kawamura, T. Kawamura, D. Fujino, F. Miyazaki, and S. Arimoto, "Realization of biped locomotion by motion pattern learning," *Journal of the Robotics Society of Japan*, vol. 3, no. 3, pp. 177–187, 1985.

[8] Q. Li, A. Takanishi, and I. Kato, "Learning control for a biped walking robot with a trunk," *Journal of the Robotics Society of Japan*, vol. 11, no. 7, pp. 1011–1019, 1993.

[9] T. B. Sheridan, "Three models of preview control," *IEEE Transactions on Human Factors in Electronics*, no. 2, pp. 91–102, 1966.

[10] K. Hu, C. Ott, and D. Lee, "Learning and generalization of compensative zero-moment point trajectory for biped walking," *IEEE Transactions on Robotics*, vol. 32, no. 3, pp. 717–725, 2016.

[11] J. Englsberger, C. Ott, and A. Albu-Schäffer, "Three-dimensional bipedal walking control based on divergent component of motion," *Ieee transactions on robotics*, vol. 31, no. 2, pp. 355–368, 2015.

[12] G. Mesesan, J. Englsberger, C. Ott, and A. Albu-Schäffer, "Convex properties of center-of-mass trajectories for locomotion based on divergent component of motion," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 3449–3456, 2018.

[13] G. Mesesan, J. Englsberger, G. Garofalo, C. Ott, and A. 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)*. IEEE, 2019, pp. 25–32.

[14] J. Englsberger, G. Mesesan, and C. 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)*. IEEE, 2017, pp. 4560–4567.

[15] Y. Wang, F. Gao, and F. J. Doyle III, "Survey on iterative learning control, repetitive control, and run-to-run control," *Journal of Process Control*, vol. 19, no. 10, pp. 1589–1600, 2009.

[16] D. A. Bristow, M. Tharayil, and A. G. Alleyne, "A survey of iterative learning control," *IEEE control systems magazine*, vol. 26, no. 3, pp. 96–114, 2006.

[17] J. Englsberger, A. Werner, C. Ott, B. Henze, M. A. Roa, G. Garofalo, R. Burger, A. Beyer, O. Eiberger, K. Schmid, *et al.*, "Overview of the torque-controlled humanoid robot toro," in *2014 IEEE-RAS International Conference on Humanoid Robots*. IEEE, 2014, pp. 916–923.

[18] F. Kanehiro, H. Hirukawa, and S. Kajita, "Openhrp: Open architecture humanoid robotics platform," *The International Journal of Robotics Research*, vol. 23, no. 2, pp. 155–165, 2004.