ADHERENT: Learning Human-like Trajectory Generators for Whole-body Control of Humanoid Robots

Paolo Maria Viceconte, Raffaello Camoriano, Giulio Romualdi, Diego Ferigo, Stefano Dafarra, Silvio Traversaro, Giuseppe Oriolo, Lorenzo Rosasco, and Daniele Pucci

***Abstract*—*Human-like* trajectory generation and footstep planning represent challenging problems in humanoid robotics. Recently, research in computer graphics investigated machine-learning methods for character animation based on training human-like models directly on motion capture data. Such methods proved effective in virtual environments, mainly focusing on trajectory visualization. This letter presents ADHERENT, a system architecture integrating machine-learning methods used in computer graphics with whole-body control methods employed in robotics to generate and stabilize *human-like* trajectories for humanoid robots. Leveraging human motion capture locomotion data, ADHERENT yields a general footstep planner, including forward, sideways, and backward walking trajectories that blend smoothly from one to another. Furthermore, at the joint configuration level, ADHERENT computes data-driven whole-body postural reference trajectories coherent with the generated footsteps, thus increasing the *human likeness* of the resulting robot motion. Extensive validations of the proposed architecture are presented with both simulations and real experiments on the iCub humanoid robot, thus demonstrating ADHERENT to be robust to varying step sizes and walking speeds.**

***Index Terms*—Humanoid robot systems, machine learning for robot control, whole-body motion planning and control.**

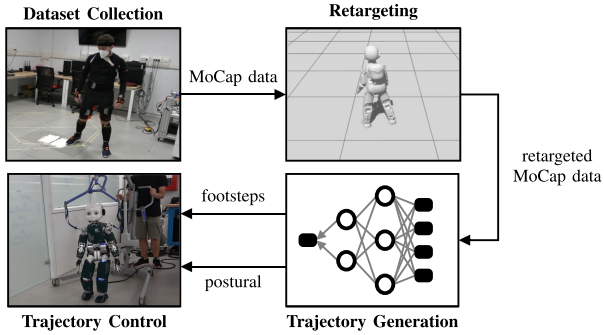

Fig. 1. Sketch of the end-to-end system architecture proposed in this work, from motion capture data collection to robot locomotion control.

I. Introduction

THE complexity of the problem of generating trajectories for humanoid robots increases considerably when targeting real-time trajectory generation for different environmental conditions and robot locomotion modes. For instance, whole-body trajectory generation methods for robot walking soon become numerically intractable due to the high dimensionality of the problem, especially when the overall generated motion is required to fulfill a certain degree of *human likeness* to prove more predictable and interpretable for humans. This paper proposes a system architecture for efficiently addressing the whole-body *human-like* trajectory generation problem for humanoid robots. The architecture builds upon recent machine-learning methods developed for character animation in computer graphics (CG),

Manuscript received September 9, 2021; accepted December 20, 2021. Date of publication January 10, 2022; date of current version February 2, 2022. This letter was recommended for publication by Associate Editor L. Peternel and Editor J. Kober upon evaluation of the reviewers' comments. The work of Daniele Pucci was supported in part by the SoftManBot Project Under Grant Agreement Number 869855, in part by the European Union's Horizon 2020 Research and Innovation Programme through An.Dy Project Under Grant Agreement Number 731540, and in part by the Italian National Institute for Insurance against Accidents (INAIL) ergoCub Project. The work of Lorenzo Rosasco was supported in part by the European Research Council under Grant SLING 819789, in part by the Center for Brains, Minds, and Machines, in part by NSF under STC Award CCF-1231216, in part by the AFOSR under Projects FA9550-18-1-7009, FA9550-17-1-0390, and BAA-AFRL-AFOSR-2016-0007 (European Office of Aerospace Research and Development), in part by the EU H2020-MSCA-RISE under Project NoMADS - DLV-777826, and in part by the NVIDIA Corporation for GPUs donations. *(Corresponding author: Paolo Maria Viceconte.)*

Paolo Maria Viceconte is with the Artificial and Mechanical Intelligence, Istituto Italiano di Tecnologia, 16163 Genoa, Italy, and also with the DIAG, Sapienza Università di Roma, 00185 Roma, Italy (e-mail: paolo.viceconte@iit.it, viceconte@diag.uniroma1.it).

Raffaello Camoriano is with the Laboratory for Computational and Statistical Learning - IIT@MIT, Istituto Italiano di Tecnologia, 16163 Genoa, Italy, and also with the Massachusetts Institute of Technology, Cambridge, MA 02139 USA (e-mail: raffaello.camoriano@iit.it).

Giulio Romualdi is with the Artificial and Mechanical Intelligence, Istituto Italiano di Tecnologia, 16163 Genoa, Italy, and also with the DIBRIS, Università degli Studi di Genova, 16132 Genoa, Italy (e-mail: giulio.romualdi@iit.it).

Diego Ferigo and Daniele Pucci are with the Artificial and Mechanical Intelligence, Istituto Italiano di Tecnologia, 16163 Genoa, Italy, and also with the Machine Learning and Optimisation, University of Manchester, Manchester M13 9PL, UK (e-mail: diego.ferigo@iit.it; daniele.pucci@iit.it).

Stefano Dafarra and Silvio Traversaro are with the Artificial and Mechanical Intelligence, Istituto Italiano di Tecnologia, 16163 Genoa, Italy (e-mail: stefano.dafarra@iit.it; silvio.traversaro@iit.it).

Giuseppe Oriolo is with the DIAG, Sapienza Università di Roma, 00185 Roma, Italy (e-mail: oriolo@diag.uniroma1.it).

Lorenzo Rosasco is with the Laboratory for Computational and Statistical Learning - IIT@MIT, Istituto Italiano di Tecnologia, 16163 Genoa, Italy, and with the Massachusetts Institute of Technology, Cambridge, MA 02139 USA, and with the DIBRIS, Università degli Studi di Genova, 16132 Genoa, Italy, and also with the Center for Brains, Minds and Machines, MIT, Cambridge, MA 02139 USA (e-mail: lrosasco@mit.edu).

This letter has supplementary downloadable material available at https://doi.org/10.1109/LRA.2022.3141658, provided by the authors.

Digital Object Identifier 10.1109/LRA.2022.3141658

integrated with state-of-the-art techniques for whole-body control in humanoid robotics.

State-of-the-art architectures for humanoid locomotion simplify the whole-body trajectory generation problem by hierarchically decomposing it into several layers [1]. Layers' functionalities can be categorized [2] in: 1) *Trajectory optimization*, providing a high-level footstep plan given user input; 2) *Simplified model control*, computing feasible Center of Mass (CoM) trajectories given the footsteps; and 3) *Whole-body QP control*, producing dynamically-feasible joint trajectories. Instead of directly optimizing over large configuration spaces, the first two layers tend to use simplified models to compute solutions. For instance, the *unicycle planner* [3] employs a unicycle model to produce footstep plans at the trajectory optimization layer, constraining the plan to simple directed walking on a plane [4], [5]. In recent years, hierarchical architectures have been successfully applied to produce robust walking on a diverse range of complex humanoids [1] [1]–[3], [6]–[10], also allowing for the integration of reactive strategies [11], [12]. However, simplified models do not fully represent the complex humanoid mechanical structure in order to reduce computational cost and allow for on-line operation. As a result, they restrict the attainable solutions set and the resulting behaviors with respect to those achieved by humans. In particular, they cannot efficiently compute walking patterns with unconstrained footstep placement. Moreover, whole-body human likeness is hard to explicitly encode and optimize for with respect to other attributes such as feasibility, stability, and robustness, and is therefore usually neglected in such schemes. Data-driven models of human trajectories have recently been explored to enable human-like behavior in robotics [13], [14]. Applications include anticipatory trajectory generation for human-robot collaboration [15]. Still, such methods focus on overall path planning (i.e., CoM trajectory), and do not target human likeness at the joint or footstep level.

Another recent research stream focuses on reinforcement learning (RL) for dynamic character control. DeepMimic [16] exploits motion capture (MoCap) data to guide policy training via imitation, demonstrating remarkable capabilities on real-world quadrupeds [17]. *Motion Matching* can also be employed alongside RL for retrieving motions from a MoCap dataset to train a policy in simulation [18]. In [19], mixtures of learning models bias RL policy learning with human data to control the lower body of a simulated humanoid. Policies learned in simulation are shown to successfully transfer to real bipeds such as Cassie [20]. Although very promising [21], the application of RL to complex real-world humanoids is still preliminary, while hierarchical control is at the state of the art.

The problem of human-like trajectory generation is not limited to robotics research. It is a prominent topic in CG research too, especially due to applications to realistic character animation, and has witnessed several recent breakthroughs based on the introduction of machine-learning methods. In particular, the core of the problem can be framed as the kinematic prediction of the whole-body joints configuration in the next time step, given the current configuration and the high-level target trajectory to be followed (i.e., obtained from human input). Many works approach this problem by modeling it as a nonlinear autoregressive model with exogenous inputs. They employ powerful learning-based predictive models able to capture the motion's complexity in high dimensions. In Phase-Functioned Neural Networks (PFNN) [22], the predictive model is a phase-weighted mixture of neural networks trained on human MoCap data. At prediction time, the network weights are blended according to a cyclic phase function encoding the periodicity of the walking motion. This resulted in a significant breakthrough for character control, enabling remarkably natural motion and smooth transitions. However, training data need to be annotated with phase function values, which can be costly or unfeasible for complex and non-periodic motions. In Mode-Adaptive Neural Networks (MANN) [23], the latter problem is solved by substituting the phase function with a *gating network*, which learns end-to-end how to effectively blend the network weights. Note that both PFNN and MANN are limited to trajectory generation for kinematic rendering only. In fact, their target applications are in settings in which natural visual appearance, rather than dynamic control of a real-world system, is the primary requirement (i.e., videogames).

In this work, we exploit efficient learning methods, proved effective in CG character animation, as a solution to the complex problem of generating whole-body trajectories for humanoid robots in real time. Our solution achieves an efficiency comparable to simplified-model trajectory generators. Still, it produces more general whole-body trajectories that state-of-the-art whole-body trajectory generators can only compute offline due to high computational costs. Moreover, such trajectories exhibit a certain degree of human-likeness both at the joint and footstep levels. To our knowledge, this work represents the first successful application of human-driven whole-body trajectory generators to real-world humanoid locomotion control. The main contributions are:

- We present ADHERENT (humAn-Driven wHolE-body REference geNerator and conTroller), a comprehensive learning-based architecture for efficient *human-like* whole-body trajectory generation and control of humanoid robots, and validate its robustness with extensive simulations and real-world experiments on the iCub humanoid robot.
- We demonstrate the generality of the learning-based footstep planner incorporated into ADHERENT by showing its adaptability to diverse walking patterns, facing directions, start and stops, and smooth transitions among these.
- We verify the improved human likeness of the whole-body motions achieved by exploiting the data-driven postural references provided by ADHERENT in conjunction with the planned footsteps.

II. Background

A. Notation

- $\mathcal{I}$ and $\mathcal{B}$ denote the inertial frame and the base frame of the robot. In the specific case of iCub [24], $\mathcal{B}$ is positioned at the level of the waist, in between the two legs, with the X axis pointing backward and the Z axis upwards.
- Given two frames $\mathcal{A}$ and $\mathcal{C}$, ${}^{\mathcal{A}}R_{\mathcal{C}} \in SO(3)$ represents the rotation matrix between the frames, i.e., given two vectors ${}^{\mathcal{A}}\boldsymbol{p}, {}^{\mathcal{C}}\boldsymbol{p} \in \mathbb{R}^3$ respectively expressed in $\mathcal{A}$ and $\mathcal{C}$, the rotation matrix ${}^{\mathcal{A}}R_{\mathcal{C}}$ is such that ${}^{\mathcal{A}}\boldsymbol{p} = {}^{\mathcal{A}}R_{\mathcal{C}}\,{}^{\mathcal{C}}\boldsymbol{p}$.
- Superscripts $\cdot^{\mathcal{H}}$ and $\cdot^{\mathcal{R}}$ indicate quantities referring to the human and the robot, respectively.
- $I_m, 0_m \in \mathbb{R}^{m\times m}$ denote the identity and zero matrices.
- When referring to network inputs $\mathbf{x}$, outputs $\mathbf{y}$, weights $\hat{\boldsymbol{\alpha}}$, and blending coefficients $\boldsymbol{\theta}$ or to their elements, subscript $\cdot_i$ indicates quantities of the i-th time step t_i.

[1] e.g., on Atlas: https://slideslive.com/38946802/boston-dynamics

- The vec$(\cdot)$ operator vectorizes matrices by rows.
- Given $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{R}^3$, we define $\boldsymbol{a}^\wedge = A \in \mathbb{R}^{3\times 3}$ as the skew-symmetric matrix such that $\boldsymbol{a}^\wedge \boldsymbol{b} = \boldsymbol{a} \times \boldsymbol{b}$.
- n denotes the robot's Degrees of Freedom (DoFs).
- $\boldsymbol{\nu} = ({}^{\mathcal{I}}\dot{\boldsymbol{p}}_{\mathcal{B}}, {}^{\mathcal{I}}\boldsymbol{\omega}_{\mathcal{B}}, \dot{\boldsymbol{s}}) \in \mathbb{R}^{6+n}$ is the generalized velocity of the complete floating-base system, where $\dot{\boldsymbol{s}}$ denotes the joint velocities. ${}^{\mathcal{I}}\dot{\boldsymbol{p}}_{\mathcal{B}}$ and ${}^{\mathcal{I}}\boldsymbol{\omega}_{\mathcal{B}}$, respectively, are the linear and angular velocities of the base frame w.r.t. the inertial frame, whose coordinates are expressed in the inertial frame, i.e., ${}^{\mathcal{I}}\dot{R}_{\mathcal{B}} = {}^{\mathcal{I}}\boldsymbol{\omega}_{\mathcal{B}}^\wedge \, {}^{\mathcal{I}}R_{\mathcal{B}}$.

B. Whole-Body Geometric Retargeting

Among the various approaches to human motion retargeting (see, e.g., [25]–[28]), Whole-Body Geometric Retargeting (WBGR) is a recent method easily adaptable to different robot models and human subjects [29]. Assuming a degree of topological similarity between the human's and robot's mechanical structures, WBGR uses m correspondences between frames associated with m human and robot links at a reference configuration. Then, given the human link orientations ${}^{\mathcal{I}}R^i_{\mathcal{H}}$, $i \in 1, \ldots, m$ to be retargeted onto the robot, WBGR allows to retrieve the robot joint angles by solving the inverse kinematics problem with the robot orientations ${}^{\mathcal{I}}R^i_{\mathcal{R}} = {}^{\mathcal{I}}R^i_{\mathcal{H}} \, {}^{\mathcal{H}}R^i_{\mathcal{R}}$ as targets: each ${}^{\mathcal{H}}R^i_{\mathcal{R}}$ is a proper constant rotation accounting for possible human-robot frame misalignment.

C. Mode-Adaptive Neural Networks

MANN is a recently-proposed neural network architecture for responsive character motion generation specifically designed for multi-modal and unlabeled data [23]. In particular, assume that $\mathbf{x}_i$ encodes the previous configuration of the controlled character as well as the desired future motion specified by the user. Then, MANN predicts a new configuration $\mathbf{y}_i$ for the character that achieves the user-specified motion. The next user input is combined with $\mathbf{y}_i$, forming the next autoregressive network input $\mathbf{x}_{i+1}$. This enables MANN to iteratively generate trajectories following the MoCap data distribution while being responsive to the user. The main characteristic of this architecture, which builds upon the *Mixture of Experts* paradigm [30], is that of being composed of two subnetworks:

- The *Motion Prediction Network:* given $\mathbf{x}_i$, it predicts $\mathbf{y}_i$;
- The *Gating Network:* given $\mathbf{x}_i$ or a subsampled input $\hat{\mathbf{x}}_i$, it predicts the blending coefficients vector $\boldsymbol{\theta}_i = [\theta_{i1}, \ldots, \theta_{iK}]^\top$ used to dynamically compute the weights vector $\hat{\boldsymbol{\alpha}}_i$ of the Motion Prediction Network from the K expert network weights vectors $\{\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_K\}$.

In an end-to-end training procedure from unstructured MoCap data, both the weights $\boldsymbol{\mu}$ of the Gating Network and the K expert weights $\{\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_K\}$ are learned. At runtime, the weights $\hat{\boldsymbol{\alpha}}_i$ of the Motion Prediction Network at time step i are dynamically computed by linearly combining the K experts $\{\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_K\}$ with the blending coefficients $\boldsymbol{\theta}_i$ predicted by the Gating Network, that is, $\hat{\boldsymbol{\alpha}}_i = \sum_{j=1}^{K} \theta_{ij} \boldsymbol{\alpha}_j$.

D. Control Architectures for Humanoid Robot Locomotion

A state-of-the-art control architecture for humanoid robot locomotion is composed of three nested layers that exploit both simplified and complete robot models [2].

TABLE I
BREAKDOWN OF THE MOCAP DATA

Walking motion	Duration [min]	Frames	Stops
Forward	8.2	29.500	25
Backward	9	32.450	25
Side	9.45	34.000	27
Diagonal	4.2	15.125	16
Mixed	30.48	109.710	75

Given the footsteps, in the outer *trajectory optimization* loop, an exponential interpolation technique is used to plan a desired Divergent Component of Motion (DCM) trajectory.

Then, the central *simplified model control* loop is in charge of stabilizing the DCM dynamics by using the Zero Moment Point (ZMP) position $\boldsymbol{r}^{zmp} \in \mathbb{R}^2$ as control input. The tracking of the desired DCM position and velocity $\boldsymbol{\xi}_{ref}, \dot{\boldsymbol{\xi}}_{ref} \in \mathbb{R}^2$ is guaranteed by the instantaneous control law given by:

$$\boldsymbol{r}^{zmp}_{ref} = \boldsymbol{\xi}_{ref} - \frac{\dot{\boldsymbol{\xi}}_{ref}}{\bar{\omega}} + K^{\xi}_p(\boldsymbol{\xi} - \boldsymbol{\xi}_{ref}) + K^{\xi}_i \int (\boldsymbol{\xi} - \boldsymbol{\xi}_{ref}) dt, \quad (1)$$

where $K^{\xi}_p > I_2$ and $K^{\xi}_i > 0_2$ [2], $\bar{\omega} = \sqrt{g/z_0}$, g is the gravitational constant and z_0 denotes the constant CoM height assumed for the Linear Inverted Pendulum (LIP) model. The desired ZMP position $\boldsymbol{r}^{zmp}_{ref}$ is then stabilized along with the reference ground CoM position and velocity $\boldsymbol{x}_{ref}, \dot{\boldsymbol{x}}_{ref} \in \mathbb{R}^2$ by means of the control law given by:

$$\dot{\boldsymbol{x}}^* = \dot{\boldsymbol{x}}_{ref} - K_{zmp}(\boldsymbol{r}^{zmp}_{ref} - \boldsymbol{r}^{zmp}) + K_{com}(\boldsymbol{x}_{ref} - \boldsymbol{x}), \quad (2)$$

where $K_{com} > \bar{\omega} I_2$ and $0_2 < K_{zmp} < \bar{\omega} I_2$ [2].

Finally, the inner *whole-body Quadratic Programming (QP) control* loop computes the robot velocity $\boldsymbol{\nu}$ as the solution to a stack of tasks formulation with hard and soft constraints, cast as a QP problem. The joint velocity $\dot{\boldsymbol{s}}$ included in the solution of the QP problem, obtained via standard QP solvers, is then integrated to get joint positions for position control.

III. ADHERENT

The proposed ADHERENT system architecture consists of four main components: *Dataset Collection*, *Retargeting*, *Trajectory Generation*, and *Trajectory Control* – see Fig. 1 for an intuitive overview and Fig. 2 for a more detailed scheme. In the following, we present the methods implementing each component. Note that, in light of ADHERENT's modularity, specific methods can be easily replaced by more efficient and effective ones in future instances of the architecture.

A. Dataset Collection

The *Dataset Collection* component is in charge of acquiring human locomotion trajectory data. For this, we use our human wearable data processing framework [31], [32] that fuses data from a sensorized suit by XSens technologies, carrying 17 wireless inertial sensors scattering the entire body. Data span a wide range of walking motions (forward, backward, lateral, and diagonal) performed on a flat terrain with continuously-changing steering direction. Stops and restarts are included in the collected sequences, characterized by footsteps of variable length. As detailed in Table I, each motion is performed for several minutes in a row. Then, plenty of transitions between different motions are collected in a long mixed sequence. Our

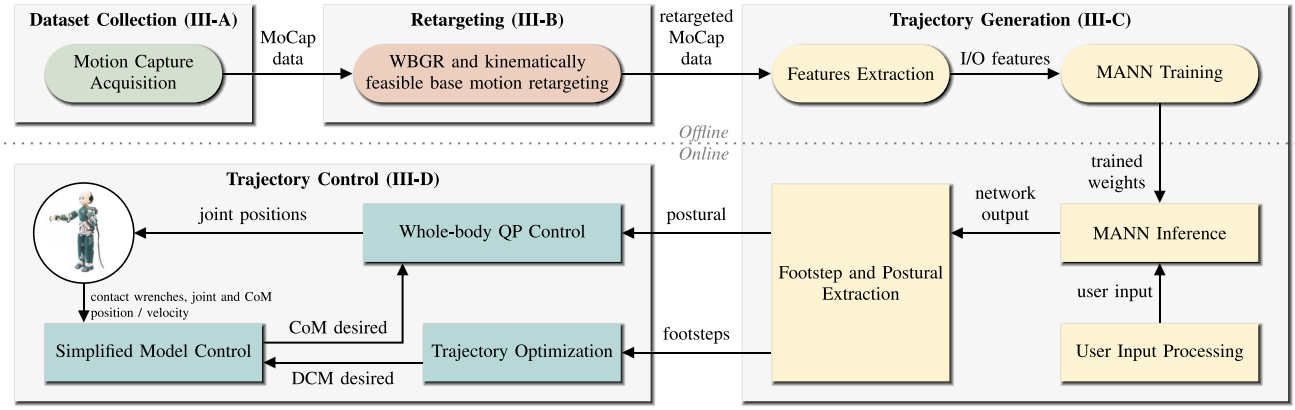

Fig. 2. Block diagram of the overall learning-based ADHERENT architecture proposed in this work.

final dataset comprises around 1 h of unlabeled MoCap data at 60 fps. We then double it by mirroring, i.e., for each data point the base orientation is mirrored with respect to the world X-Z plane, while the left and right link orientations for the limbs are switched and mirrored with respect to the model's mid-sagittal plane, resulting in a total of 441570 data points.

B. Retargeting

The *Retargeting* component adjusts the human trajectories so as when the modified trajectories are applied to the robot, its motion turns out to be *similar* to the human one. We retarget the MoCap dataset by complementing the WBGR introduced in Section II-B with a kinematically-feasible base motion retargeting procedure which renders the robot base velocity compatible with the retargeted robot joint trajectories.

1) Kinematically-Feasible Base Motion Retargeting: WBGR in [29] does not address the base motion retargeting, i.e., the retargeted base position and orientation may not be compatible with the robot kinematics, thus possibly leading to a robot moving forward faster than what its walking pace entails. In other words, a swaying effect arises when dynamic motions are retargeted to robot models structurally different from the human subject [29]. While in CG generating models which fit the collected data is a viable workaround [23] [18], base motion retargeting for actual robots requires special attention.

First, assume that: i) the robot makes at least one known contact with the environment; and ii) each foot is modeled as a rectangular patch. Then, we propose the following approach for kinematically-feasible base motion retargeting:

1) The contact point ${}^{\mathcal{I}}\boldsymbol{p}_c$ is identified as the lowest among the 8 vertices of the feet's rectangular approximations;
2) The retargeted base orientation ${}^{\mathcal{I}}R_{\mathcal{B}}$ is directly retrieved from the MoCap data;
3) The retargeted base position ${}^{\mathcal{I}}\boldsymbol{p}_b$ is computed by forward kinematics from ${}^{\mathcal{I}}\boldsymbol{p}_c$, constrained to remain fixed between two consecutive retargeting steps, via ${}^{\mathcal{I}}\boldsymbol{p}_b = {}^{\mathcal{I}}\boldsymbol{p}_c + {}^{\mathcal{I}}R_{\mathcal{C}}{}^{\mathcal{C}}\boldsymbol{p}_b$, where $\mathcal{C}$ is the frame attached to the contact point (i.e., the frame placed in the lowest vertex and oriented as the support foot) and ${}^{\mathcal{C}}\boldsymbol{p}_b$ is the base position, expressed in $\mathcal{C}$, computed by forward kinematics in the updated joint configuration returned by the latest WBGR iteration.

As a result, we obtain retargeted motions that resemble human ones at the links level and are kinematically feasible at the base level, as shown in the supplementary video.

C. Trajectory Generation

We interactively generate trajectories for the humanoid robot by exploiting the MANN architecture outlined in Section II-C. Details follow on the features extraction and the network structure, inspired by those illustrated in [23]. Moreover, the processing of the input and output of the network at inference time is illustrated, including the footstep and postural extraction.

1) Features Extraction: The input and output vectors for MANN are extracted by processing the retargeted MoCap data. The input vector $\mathbf{x}_i$ includes the state of the robot at t_{i-1} and the ground base trajectory data at t_i, that is, past and future base trajectory data projected on the ground, subsampled to obtain 12 data points equally spaced on a 2 s window centered at t_i. More in detail:

$$\mathbf{x}_i = \{\underbrace{\mathrm{vec}(P_i),\ \mathrm{vec}(D_i),\ \mathrm{vec}(V_i),\ l_i}_{\text{Base trajectory}},\ \underbrace{\boldsymbol{s}_{i-1},\ \dot{\boldsymbol{s}}_{i-1}}_{\text{Robot state}}\} \in \mathbb{R}^{137}, \tag{3}$$

where $P_i = \{\boldsymbol{p}_i^1, \ldots, \boldsymbol{p}_i^{12}\} \in \mathbb{R}^{2\times 12}$ are ground base positions, $D_i = \{\boldsymbol{d}_i^1, \ldots, \boldsymbol{d}_i^{12}\} \in \mathbb{R}^{2\times 12}$ are ground facing directions (i.e., mean of base and chest pointing directions, projected on the ground), $V_i = \{\boldsymbol{v}_i^1, \ldots, \boldsymbol{v}_i^{12}\} \in \mathbb{R}^{2\times 12}$ are ground base velocities and $l_i = \sum_{j=7}^{12} \|\boldsymbol{p}_i^j - \boldsymbol{p}_i^{j-1}\| \in \mathbb{R}$ is the length of the future ground trajectory, while $\boldsymbol{s}_{i-1} \in \mathbb{R}^{32}$ and $\dot{\boldsymbol{s}}_{i-1} \in \mathbb{R}^{32}$ are joint positions and velocities at t_{i-1}.

The output vector $\mathbf{y}_i$ includes the state of the robot at t_i, the ground angular base transformation from t_{i-1} to t_i, and the future ground base trajectory data at t_{i+1} (i.e. 6 data points equally spaced on a 1 s window starting from t_i). Specifically:

$$\mathbf{y}_i = \{\underbrace{\mathrm{vec}(P_{i+1}), \mathrm{vec}(D_{i+1}), \mathrm{vec}(V_{i+1})}_{\text{Future base trajectory}}, \underbrace{\boldsymbol{s}_i, \dot{\boldsymbol{s}}_i}_{\substack{\text{Robot}\\ \text{state}}}, \underbrace{\dot{b}_i^a}_{\substack{\text{Base}\\ \text{transf.}}}\} \in \mathbb{R}^{101}, \tag{4}$$

where $P_{i+1} = \{\boldsymbol{p}_{i+1}^1, \ldots, \boldsymbol{p}_{i+1}^6\} \in \mathbb{R}^{2\times 6}$ are future ground base positions, $D_{i+1} = \{\boldsymbol{d}_{i+1}^1, \ldots, \boldsymbol{d}_{i+1}^6\} \in \mathbb{R}^{2\times 6}$ are future ground facing directions, $V_{i+1} = \{\boldsymbol{v}_{i+1}^1, \ldots, \boldsymbol{v}_{i+1}^6\} \in \mathbb{R}^{2\times 6}$ are future ground base velocities, $\boldsymbol{s}_i, \dot{\boldsymbol{s}}_i \in \mathbb{R}^{32}$ are joint positions and velocities at t_i, while $\dot{b}_i^a = \beta_i / \Delta t_i \in \mathbb{R}$, with $\Delta t_i = t_i - t_{i-1}$, and β_i denoting the angle between the ground facing directions at t_i and t_{i-1} (i.e., $\boldsymbol{d}_i^6$ and $\boldsymbol{d}_{i-1}^6$, respectively).

All the ground base trajectory data in $\mathbf{x}_i$ and $\mathbf{y}_i$ are expressed in the bidimensional local reference frame defined by the ground base position at t_i (i.e., $\boldsymbol{p}_i^6$) and the ground facing direction at

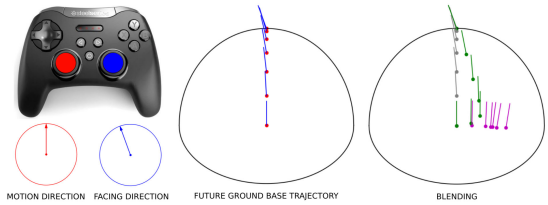

Fig. 3. Desired motion and facing directions from the joypad (left) define the desired future ground base trajectory (center). On the right, the user-specified future trajectory (grey) is blended with the future trajectory from the previous network prediction (magenta), leading to the desired future ground base trajectory (green) actually included in the next input to the network.

t_i (i.e., $\boldsymbol{d}_i^6$) along with its orthogonal vector. By stacking all the input and output vectors resulting from the processing above, we obtain the input and output matrices $X \in \mathbb{R}^{441570\times 137}$ and $Y \in \mathbb{R}^{441570\times 101}$ which, normalized to have zero mean and unit variance, constitute our training set.

2) Network Structure: The adopted MANN is composed of a Motion Prediction Network and a Gating Network with 3 hidden layers of 512 and 32 units each, respectively. As in [23], the ELU activation function is employed. The Gating Network receives the full input $\mathbf{x}_i$. We use $K = 4$ experts.

3) User Input Processing: At inference time, the user provides via joypad two continuous signals to interactively generate trajectories for the robot: i) the *motion direction:* the direction in which the user wants the robot to move; and ii) the *facing direction:* the direction towards which the user wants the robot to align the mean of its base and torso horizontal pointing directions. At fixed facing direction, varying the motion direction allows to switch between frontal, sideways, and backward walking. At fixed motion direction, varying the facing direction allows steering. Moreover, releasing the analog for the motion direction leads the robot to a stop. The user inputs are visualized in Fig. 3 (left), from the local viewpoint of a robot proceeding forward while steering left.

An accurate processing of user inputs is critical for the predictive performances of MANN. We smoothly interpolate the inputs to generate a desired future ground base trajectory $\{P_{i+1}^*, D_{i+1}^*, V_{i+1}^*\}$ whose components are defined as in (4). In particular, the user-specified motion direction is used to define the last point of a quadratic Bézier curve starting from $\boldsymbol{p}_i^6$ and constrained to end on the asymmetric shape shown in black in Fig. 3, composed of two experimentally-found semi-ellipses with horizontal axis of 0.35 m and vertical axes of 0.4 m and 0.25 m for the upper and lower semi-ellipse, respectively. We obtain P_{i+1}^* by subsampling 6 data points from this Bézier curve. As a result of the asymmetric constraint, P_{i+1}^* is longer for forward rather than sideways or backward walking. The user-specified facing direction is instead mapped into a series of facing directions D_{i+1}^* progressively driving the current value to the desired one. V_{i+1}^* is obtained by differentiating P_{i+1}^*. Fig. 3 (center) provides a visualization, from the local robot's viewpoint, of P_{i+1}^* (red dots) and D_{i+1}^* (blue vectors) generated from the user-specified inputs (left).

As a last step, $\{P_{i+1}^*, D_{i+1}^*, V_{i+1}^*\}$ is blended with the $\{P_{i+1}, D_{i+1}, V_{i+1}\}$ retrieved from the previous output $\mathbf{y}_i$ in order to obtain the ground base trajectory data for the next input $\mathbf{x}_{i+1}$. For the blending, an example of which is shown in Fig. 3 (right), we follow the method proposed in [22], Eq. (9), with the responsiveness parameters for positions, facing directions and velocities respectively set to $\tau_p = 1.5$, $\tau_d = \tau_v = 1.3$.

Note that user-specified motion and facing directions may result in a desired motion which is absent or rare in the training dataset. In such case (e.g., when the user requests to steer too abruptly) the network may generate unexpected motions. We solve this issue by limiting the local variation of facing direction that the user can require to 45° and 20° for forward and backward/sideways walking, respectively.

4) Network Output Postprocessing: When the user tries to stop the robot, a small in-place rotation persists. Indeed, given an $\mathbf{x}_i$ corresponding to a desired stop, the network predicts a $\mathbf{y}_i$ whose $\dot{b}_i^a$ component is slightly different from zero. We solve this issue by imposing $\dot{b}_i^a = 0$ once a stop by the robot is detected. Here, we are referring to stops at the network level, which can occur several time instants after the user releases the joypad. We detect such stops by searching for almost-identical consecutive network outputs. In our case, a stop is detected if $\|\mathbf{y}_i - \mathbf{y}_{i-1}\| < \tau_{stop}$, with $\tau_{stop} = 0.05$.

After the processing above, $\mathbf{y}_i$ is used to update the robot configuration. In particular, $\boldsymbol{s}_i$ becomes the new joint configuration and $\dot{b}_i^a$ is exploited to update the base orientation ${}^{\mathcal{I}}R_{\mathcal{B}}$. On the contrary, $\mathbf{y}_i$ contains no information on the linear motion of the robot base. The updated base position ${}^{\mathcal{I}}\boldsymbol{p}_b$ is indeed computed by applying the very same kinematic feasibility procedure used at the retargeting stage (Section III-B).

5) Footstep and Postural Extractor: The desired feet positions and orientations composing the footstep plan are retrieved from the generated trajectory by the *Footstep Extractor*. A new foot position is added to the plan once the support foot changes. Concerning orientations, in the case of flat terrain the plan only includes the predicted yaw angle of the support foot.

The joint positions $\boldsymbol{s}_i \in \mathbf{y}_i$ constitute a whole-body human-like postural. However, having been trained on data collected at 60 fps, the network generates references compatible with such frequency. Still, the *whole-body QP control* layer may require posturals at a different frequency (e.g., in this work, 100 Hz). The *Postural Extractor* interpolates the network's predictions to obtain postural references at the required frequency.

D. Trajectory Control

We execute on the robot the genereted walking trajectories by leveraging the control architecture introduced in Section II-D.

Given the footstep plan provided by the *Footstep Extractor*, a reference DCM trajectory is obtained by the planner described within the *Trajectory Optimization* layer in Section II-D. In particular, we adopt the implementation from [33], providing a feasible DCM trajectory even for variable CoM height.

Given a desired DCM trajectory, at each control cycle the *Simplified Model Control* layer computes the desired CoM velocity $\dot{\boldsymbol{x}}^*$ by concatenating (1) and (2). Then, the desired CoM position $\boldsymbol{x}^*$ is retrieved by Euler integration.

A QP problem is formulated starting from the postural returned by the *Postural Extractor*. The desired feet poses and CoM trajectory $\{\dot{\boldsymbol{x}}^*, \boldsymbol{x}^*\}$ are set as hard constraints. A soft constraint aims to zero chest roll and pitch angles. The desired joint velocity $\dot{\boldsymbol{s}}^*$ included in the QP solution is integrated, and the resulting desired joint position $\boldsymbol{s}^*$ is sent to the robot.

IV. Results

We now illustrate the results obtained after training the proposed architecture on the dataset described in Section III-C1. 150 training epochs on the whole dataset, using mini-batches of

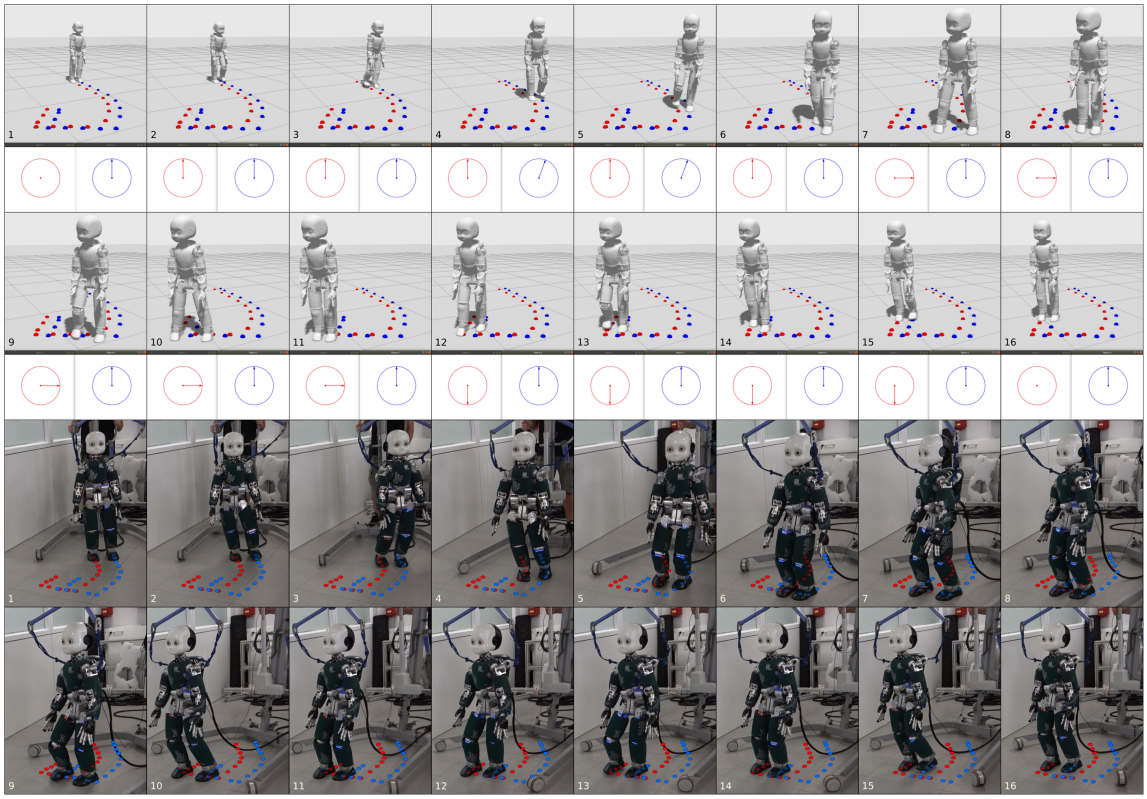

Fig. 4. *Top:* Visualization of a mixed trajectory including forward (1-3), oriented-forward (4-6), side (7-11), and backward (12-15) walking, with smooth transitions between them and a final stop (16). Below each frame, the user inputs interactively shaping the trajectory are plotted from the robot's local viewpoint (red: Desired motion direction; blue: Desired facing direction). *Bottom:* The same trajectory performed on iCub. Numbering highlights frame correspondences.

32 randomly selected samples, require around 25 hours on an NVIDIA GeForce GTX 1650 GPU. The MANN architecture is trained in a classical regression setting, minimizing the mean squared error (MSE) between the ground truth and the network prediction. The Adam optimizer with warm restart (AdamWR) is configured as in [23] (i.e., learning rate $\eta = 1.0 \cdot 10^{-4}$, weight decay rate $\lambda = 2.5 \cdot 10^{-3}$, η-restart control parameters $T_i = 10$, and $T_{mul} = 2$). The Dropout keep probability is set to 0.7. Data and code are available at https://github.com/ami-iit/paper_viceconte_2021_ral_adherent.

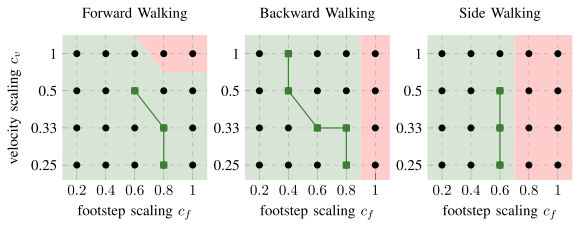

Fig. 5. Simulated and experimental results for different combinations of c_v and c_f. Red and green areas denote failures and successes in simulation. The green line connects the most challenging successes on the real robot.

A. Trajectory Generation

Once trained, ADHERENT's *Trajectory Generation* component interactively generates walking trajectories. Each network prediction requires around 3 ms on a 9-th generation Intel Core i7 CPU @ 2.60 GHz. By simply varying motion and facing directions, the user is able to move the robot forward, backward, and sideways in a human-like fashion. Changes in the input signals promptly translate into smooth transitions between different walking patterns. By releasing the motion direction analog stick, the user can stop the robot and then restart the motion at will. Fig. 4 (top) shows the kinematic visualization of a complex trajectory, including several walking patterns and smooth transitions between them, interactively generated from the user inputs shown below each frame. The footstep positions extracted from the entire trajectory are visualized in red and blue for the right and left foot, respectively. A larger variety of trajectory generations is reported in the supplementary video.

B. Trajectory Control

We execute the generated trajectories on the real-world 32-Degree of Freedom (DoF) iCub humanoid [24], which is 104 cm tall and weighs approximately 33 Kg. The control architecture composed by the *Simplified Model Control* and *Whole Body QP Control* layers runs at 100 Hz on a 4-th generation Intel Core i7 @ 1.7 GHz. Fig. 4 (bottom) illustrates the successful execution of the complex trajectory visualized in the upper part of the same figure. The footstep sequence performed by the robot is added for the sake of clarity. The execution of this trajectory, along with others, is shown in the supplementary video.

C. Robustness Analysis

We evaluate the robustness of ADHERENT to a challenging range of step sizes and walking speeds, both in simulation (using the Gazebo simulator [34]) and on the real robot. The generated forward, backward, and side walking are characterized by a

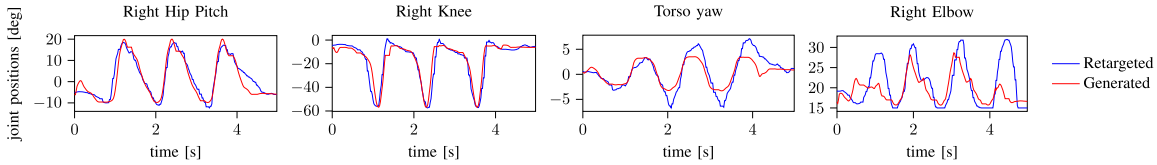

Fig. 6. ADHERENT-generated vs. human-retargeted joint trajectories for a set of representative joints during forward walking.

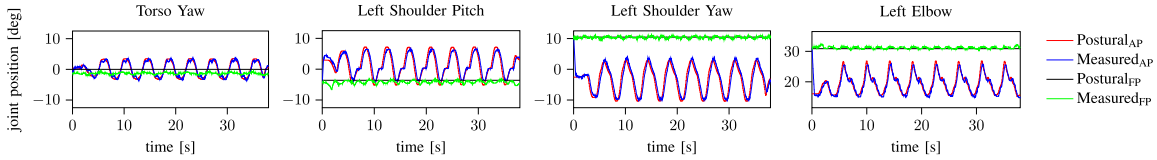

Fig. 7. Output of the *Postural Extractor* and measured joint positions with ADHERENT Postural (AP) vs. Fixed Postural (FP) for four upper-body joints.

nominal walking speed v (defined as average swing velocity [2]) of 0.525, 0.175, and 0.24 ms^{-1}, and a nominal average step size f of 28.0, 19.5, and 17.5 cm. We scale the generated trajectories by a velocity scaling $c_v \in \{0.25, 0.33, 0.5, 1\}$ and a footstep scaling $c_f \in \{0.2, 0.4, 0.6, 0.8, 1\}$, resulting in a scaled footstep size $\bar{f} = c_f \cdot f$ and walking speed $\bar{v} = c_v \cdot c_f \cdot v$. Concerning simulations, Fig. 5 illustrates the results for all the combinations of c_v and c_f, with the successful trials and failures represented by the green and the red areas, respectively. For each motion, most combinations are successful. Regardless of c_v, the maximum admissible c_f for successful backward and side walking is 0.8 and 0.6, respectively. As regards real experiments, the solid green line in Fig. 5 connects the most challenging parameter combinations resulting in successful outcomes. In particular, a $\bar{v}$ of 0.158, 0.068, and 0.071 ms^{-1} and an $\bar{f}$ of 16.8, 7.9, and 10.5 cm are achieved in the most challenging forward, backward, and side walking, respectively.

D. Human Likeness

We evaluate the *human likeness* of the trajectories generated with ADHERENT by comparing them with ground-truth trajectories directly retargeted from the human. A comparison of four representative joints during forward walking is displayed in Fig. 6. The generated trajectories, learned from a large dataset, do not follow in every detail the specific individual retargeted motion they are compared with (e.g. elbow in Fig. 6). Still, the similarity of the patterns demonstrates that the trained model is indeed able to generate trajectories which resemble the human motions retargeted onto the robot, being therefore *human-like*. The accuracy of the generated trajectories is confirmed by an average value of 3.09 degrees for the test-set RMSE computed on the network-predicted joint positions $\boldsymbol{s}_i \in \mathbf{y}_i$.

Moreover, we evaluate the *human likeness* of the actual robot motion by showing that the measured joint positions on the real-world iCub closely track the ADHERENT-generated trajectories (i.e., the output of the *Postural Extractor*), despite the lower-level action of the *Whole Body QP Controller*. This can be seen, for a representative set of joints during forward walking, from the trajectories using the ADHERENT postural in Fig. 7. The good tracking demonstrates how reference motions produced by generators trained on human data can actually be realized on the real robot, resulting in *human-like* motions.

Finally, we compare forward walking using ADHERENT postural with one adopting a fixed postural for the upper body,

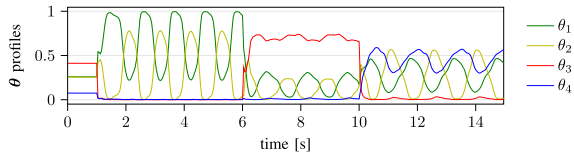

Fig. 8. Blending coefficients $\boldsymbol{\theta}$ profiles with $K = 4$ expert weights for an articulated trajectory including standing (0-1 s), straight (1-4 s) and steered (4-6 s) forward walking, right-side (6-10 s), and left-side (10-15 s) walking.

as is often the case in classical humanoid robot locomotion. When using a fixed postural, measured joint positions oscillate in proximity of the constant reference (see Fig. 7). A side-by-side comparison of the motions is shown in the supplementary video. As it can be observed, the overall motion with ADHERENT postural shows an improved *human likeness*.

E. Blending Coefficients Activation

We analyze how the experts specialize in different motions by plotting the profiles of the corresponding blending coefficients $\boldsymbol{\theta}$ in Fig. 8. Note that $\boldsymbol{\theta}$ activations show distinctive periodic patterns characterizing each motion type. For instance, in both the straight and steered forward walking phases, only θ_1 and θ_2 are active, and specialize in left and right swing motions, respectively. Moreover, θ_3 and θ_4 become active during right- and left-side walking, respectively. The real-time evolution of expert activations is shown in the supplementary video.

V. Discussion

In terms of trajectory generation, ADHERENT is able to compute whole-body reference trajectories in real time (3 ms per prediction step), thanks to efficient neural-network-based feedforward prediction. Using ADHERENT, a wide variety of walking motions are successfully executed, with smooth transitions from one another, on an advanced humanoid robot whose physical properties significantly differ from those of the human body. Improving *human likeness* of the overall robot motion is an additional feature enabled by ADHERENT.

As long as meaningful correspondences between the human and the robot links can be defined to retarget the MoCap data, the proposed approach can be easily transferred to other humanoid robots. To this end, training from scratch on the customly-retargeted dataset for the new platform is required.

Furthermore, the current implementation of the *Retargeting* component assumes that the robot has at least one known contact with the environment at any time step. This prevents applicability to contact-free motions, e.g., running or jumping.

Moreover, we observe that different motion types have specific optimal velocity and footstep scaling factors (see Fig. 5). Adaptively adjusting these factors when transitioning between motions could improve the overall robot mobility.

Finally, the constraints imposed on the user-specified input (see Section III-C3) do not allow the robot to turn abruptly.

VI. Conclusion

We presented ADHERENT, a novel architecture for whole-body trajectory generation and control of humanoid robots. ADHERENT joins together the advantages of learning-based trajectory generation and state-of-the-art hierarchical locomotion control. Our approach allows for human-like whole-body motions and general footstep plans to be efficiently computed and executed on a complex humanoid. We demonstrate the applicability and robustness of the architecture through an extensive validation in simulation and experimental campaign on iCub. ADHERENT represents a promising first step towards the development of general human-like trajectory generation, sidestepping the computational complexity of classical trajectory optimization methods by learning from human data.

In this work, we employed instantaneous controllers for trajectory control. Possible future work includes investigating, comparing, and integrating ADHERENT with more advanced control architectures (i.e., MPC- or RL-based). From a machine-learning perspective, ADHERENT must be retrained from scratch whenever new motion skills need to be added. This could be tackled by integrating continual/lifelong learning methods in the architecture. Further architectural changes could be introduced to enable trajectory generation in accordance with different walking styles. Finally, an extension of our work for the navigation of uneven ground could be pursued by including perceptual terrain features in the network input.

References

[1] S. Feng, E. Whitman, X. Xinjilefu, and C. G. Atkeson, "Optimization based full body control for the atlas robot," in *Proc. IEEE-RAS Int. Conf. Humanoid Robots*, 2014, pp. 120–127.

[2] G. Romualdi *et al.*, "A benchmarking of DCM based architectures for position, velocity and torque controlled humanoid robots," *Int. J. Humanoid Robot.*, vol. 17, no. 1, 2020.

[3] S. Dafarra *et al.*, "A control architecture with online predictive planning for position and torque controlled walking of humanoid robots," in *Proc. Int. Conf. Intell. Robots Syst.*, 2018, pp. 1–9.

[4] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, *Robotics: Modelling, Planning and Control*. 3rd ed., Berlin, Germany: Springer, 2008.

[5] T.-V.-A. Truong, D. Flavigne, J. Pettrèe, K. Mombaur, and J.-P. Laumond, "Reactive synthesizing of human locomotion combining nonholonomic and holonomic behaviors," in *Proc. 3rd IEEE RAS and EMBS Int. Conf. Biomed. Robot. Biomechatronics*, 2010, pp. 632–637.

[6] H. Dai, A. Valenzuela, and R. Tedrake, "Whole-body motion planning with centroidal dynamics and full kinematics," in *Proc. IEEE-RAS Int. Conf. Humanoid Robots*, 2014, pp. 295–302.

[7] A. Herzog, N. Rotella, S. Schaal, and L. Righetti, "Trajectory generation for multi-contact momentum control," in *Proc. IEEE-RAS 15th Int. Conf. Humanoid Robots*, 2015, pp. 874–880.

[8] J. Carpentier, S. Tonneau, M. Naveau, O. Stasse, and N. Mansard, "A versatile and efficient pattern generator for generalized legged locomotion," in *Proc. Int. Conf. Robot. Autom.*, 2016, pp. 3555–3561.

[9] 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 *Proc. IEEE-RAS Int. Conf. Humanoid Robots*, 2019, pp. 25–32.

[10] N. Ramuzat, G. Buondonno, S. Boria, and O. Stasse, "Comparison of position and torque whole body control schemes on the TALOS humanoid robot," *Conf. Adv. Robot.*, pp. 785–792, 2021.

[11] M. Bombile and A. Billard, "Capture-point based balance and reactive omnidirectional walking controller," in *Proc. IEEE-RAS Int. Conf. Humanoid Robots*, 2017, pp. 17–24.

[12] M. Shafiee, G. Romualdi, S. Dafarra, F. J. A. Chavez, and D. Pucci, "Online DCM trajectory generation for push recovery of torque-controlled humanoid robots," in *Proc. IEEE-RAS Int. Conf. Humanoid Robots*, 2019, pp. 671–678.

[13] K. Mombaur, A. Truong, and J.-P. Laumond, "From human to humanoid locomotion - an inverse optimal control approach," *Auton. Robots*, vol. 28, pp. 369–383, 2010.

[14] A. V. Papadopoulos, L. Bascetta, and G. Ferretti, "Generation of human walking paths," in *Proc. Int. Conf. Intell. Robots Syst.*, 2013, pp. 1676–1681.

[15] I. Maroger, N. Ramuzat, O. Stasse, and B. Watier, "Human trajectory prediction model and its coupling with a walking pattern generator of a humanoid robot," *IEEE Robot. Automat. Lett.*, vol. 6, no. 4, pp. 6361–6369, Oct. 2021.

[16] X. B. Peng, P. Abbeel, S. Levine, and M. van de Panne, "DeepMimic: Example-guided deep reinforcement learning of physics-based character skills," *ACM Trans. Graph.*, vol. 37, no. 4, pp. 1–14, 2018.

[17] X. B. Peng, E. Coumans, T. Zhang, T.-W. Lee, J. Tan, and S. Levine, "Learning agile robotic locomotion skills by imitating animals," *Robot.: Sci. Syst.*, Jul. 2020, doi: 10.15607/RSS.2020.XVI.064.

[18] K. Bergamin, S. Clavet, D. Holden, and J. R. Forbes, "DReCon: Data-driven responsive control of physics-based characters," *ACM Trans. Graph.*, vol. 38, no. 6, pp. 1–11, 2019.

[19] C. Yang, K. Yuan, S. Heng, T. Komura, and Z. Li, "Learning natural locomotion behaviors for humanoid robots using human bias," *IEEE Robot. Automat. Lett.*, vol. 5, no. 2, pp. 2610–2617, Apr. 2020.

[20] Z. Xie, P. Clary, J. Dao, P. Morais, J. Hurst, and M. van de Panne, "Learning locomotion skills for cassie: Iterative design and sim-to-real," in *Proc. Conf. Robot Learn.*, 2020, vol. 100, pp. 317–329.

[21] D. Rodriguez and S. Behnke, "DeepWalk: Omnidirectional bipedal gait by deep reinforcement learning," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2021, pp. 3033–3039.

[22] D. Holden, T. Komura, and J. Saito, "Phase-functioned neural networks for character control," *ACM Trans. Graph.*, vol. 36, no. 4, pp. 1–13, 2017.

[23] H. Zhang, S. Starke, T. Komura, and J. Saito, "Mode-adaptive neural networks for quadruped motion control," *ACM Trans. Graph.*, vol. 37, no. 4, pp. 1–11, 2018.

[24] G. Metta *et al.*, "The iCub humanoid robot: An open-systems platform for research in cognitive development," *Neural Netw.*, vol. 23, no. 8–9, pp. 1125–1134, 2010.

[25] N. Pollard, J. Hodgins, M. Riley, and C. Atkeson, "Adapting Human motion for the control of a humanoid robot," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2002, pp. 1390–1397.

[26] C. Ott, D. Lee, and Y. Nakamura, "Motion capture based human motion recognition and imitation by direct marker control," in *Proc. 8th IEEE-RAS Int. Conf. Humanoid Robots*, 2008, pp. 399–405.

[27] K. Miura *et al.*, "Robot motion remix based on motion capture data towards human-like locomotion of humanoid robots," in *Proc. 9th IEEE-RAS Int. Conf. Humanoid Robots*, 2009, pp. 596–603.

[28] L. Penco *et al.*, "Robust real-time whole-body motion retargeting from human to humanoid," in *Proc. IEEE-RAS Int. Conf. Humanoid Robots*, 2018, pp. 425–432.

[29] K. Darvish *et al.*, "Whole-body geometric retargeting for humanoid robots," in *Proc. IEEE-RAS Int. Conf. Humanoid Robots*, 2019, pp. 679–686.

[30] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton, "Adaptive mixtures of local experts," *Neural Computation*, vol. 3, no. 1, pp. 79–87, 1991.

[31] C. Latella *et al.*, "A human wearable framework for physical human-robot interaction," in *Proc. I-RIM*, 2019, pp. 70–71.

[32] L. Rapetti, Y. Tirupachuri, K. Darvish, C. Latella, and D. Pucci, "Model-based real-time motion tracking using dynamical inverse kinematics on SO(3)," *Algorithms*, vol. 13, no. 10, 2020, Art. no. 266.

[33] G. Romualdi, S. Dafarra, G. L'Erario, and D. Pucci, "Non-linear DCM trajectory optimization with variable center of mass height," in *Proc. I-RIM*, 2020, pp. 129–130.

[34] N. Koenig and A. Howard, "Design and use paradigms for gazebo, an open-source multi-robot simulator," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2004, pp. 2149–2154.