IEEE Access
Multidisciplinary ⁝ Rapid Review ⁝ Open Access Journal

Received October 5, 2020, accepted October 29, 2020, date of publication November 6, 2020, date of current version November 18, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3036449

Modularized Predictive Coding-Based Online Motion Synthesis Combining Environmental Constraints and Motion-Capture Data

JAEPYUNG HWANG[1], SHIN ISHII[1,2,3], TAESOO KWON[4], AND SHIGEYUKI OBA[1]

[1]Department of Systems Science, Graduate School of Informatics, Kyoto University, Kyoto 606-8501, Japan
[2]International Research Center for Neurointelligence (WPI-IRCN), University of Tokyo Institutes for Advanced Study, The University of Tokyo, Tokyo 113-0033, Japan
[3]Advanced Telecommunications Research Institute International (ATR), Seika 619-0288, Japan
[4]Department of Computer Software, Hanyang University, Seoul 04763, South Korea

Corresponding author: Jaepyung Hwang (yapyungih@gmail.com)

This work was supported in part by a project, JPNP20006, commissioned by the New Energy and Industrial Technology Development Organization (NEDO), Japan, in part by the Post-K Application Development for Exploratory Challenges project of MEXT, Japan, and in part by JSPS KAKENHI under Grant 17H06308 and Grant 19H04180.

ABSTRACT Motion synthesis benefits from the use of motion capture data and a dynamic model because the motion data can provide a reference to naturalness, and the dynamic model can support environmental constraints such as footskate prevention or perturbation response. However, a combination of a dynamic model and captured motion usually demands professional insights, experience, and additional efforts such as preprocessing or off-line optimization. To address this issue, we propose a modularized predictive coding-based motion synthesis framework that synthesizes natural motion while maintaining the constraints. Modularized predictive coding provides intuitive online mediation of multiple information sources, which can then be applied to motion synthesis. To validate the proposed framework, we applied different types of motion data and character models to synthesize human walking, kickboxing, and backflipping motions, a dog walking motion, and a hand object-grasping motion.

INDEX TERMS Combination of linear models, hybrid-based character animation, neuroscience-inspired, online motion synthesis.

I. INTRODUCTION

Natural character animation in a virtual world is an important component in various applications: moviemaking, building video game environments, and a number of other virtual-reality services that have emerged recently. In these fields, motion-capture (mocap) data have been utilized as sources of natural motions [1], [2]. In many real-time applications, however, character motions need to be adapted to fit various environmental constraints in the virtual world, which vary according to user commands and changes in the environment.

Preserving these constraints reduces motion artifacts when changing the environment or situation [3], [4]. Such kinematic approaches, nevertheless, can occasionally generate unrealistic motions due to the lack of recoverable reference motions in every situation. The hybrid approach uses both mocap data and dynamic models to generate smooth motions that adapt to changing virtual environments. This approach outperforms when a mocap data source is limited because applications with a rich environment often require motions of infinite varieties, and the dynamic model is expected to cover the shortage. Dynamic models, however, require complicated parameter tuning and often generate character motions that are not sufficiently natural to represent human or animal motion [5]. Some researchers have used dynamic models to implicitly apply motion-specific constraints [6], [7]. This type of setup has two advantages. The first is the computational efficiency needed to predict paths and to generate motion along a path [6], [7]. The second is consideration of the environmental constraints on locomotion [8], [9]. However, because the motion pattern of the dynamic model differs from human motion (e.g., mocap), the generated motion may lead to an unnatural character posture that

The associate editor coordinating the review of this manuscript and approving it for publication was Choon Ki Ahn.

This work is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 License.
For more information, see https://creativecommons.org/licenses/by-nc-nd/4.0/

exceeds the physical limits. On the other hand, when only focusing on environmental constraints, the generated motion can be unnatural and jerky. A tracking controller reduces the difference between the motion from the model and the captured motion, while satisfying the constraints of the dynamic model [10]. However, the mapping correspondence between the dynamic model and the mocap data must be specified, and preprocessing such as offline optimization and learning stages [10] are required.

To overcome this difficulty, we propose an online character motion synthesis framework that intuitively utilizes the dynamic model and the mocap data. The predictive coding model (PCM) is well investigated in neuroscience [11]. It assumes that a set of neurons in our brain constructs an internal model that has a hidden state. The internal model receives an observation or perception, updates its state according to the observation, and predicts future observations. The brain then revises its internal state online to minimize the prediction error. We expect this reimbursement to improve the performance of previous optimal control systems, such as a linear–quadratic regulator (LQR), and to filter out the implausible synthesized motion.

As with PCM, we developed a new motion synthesis procedure, ‘‘modularized predictive coding-based online motion synthesis,’’ by applying the concept of PCM to motion synthesis. First, we regard the mocap data as a part of the perception of naturalness, referred to as a ‘‘provider.’’ Second, we use multiple linear dynamic models that correspond to the degrees of freedom (DOFs; joints or sets of joints) of the character model. We regard the state variables of the dynamic model as internal states. We refer to the state and its dynamic model for each DOF of the character as a ‘‘bundle.’’ Third, we construct a predictive mapping from the internal state to the perception. The prediction error is calculated for each perception, and provides feedback to update the state. We call this module a ‘‘matcher,’’ which links a provider and a bundle. Fourth, we generate a virtual control signal to minimize the error between the desired and predicted future state. This is calculated in each bundle. The relationships between the provider, bundle, and matcher are illustrated in Figure 1. Overall, our procedure updates the current posture of the character as in the (extended) Kalman filter, with the simultaneous calculation of the control signal as in the LQR controller, in an online manner. More importantly, our framework is easily extended to include multiple providers, bundles, and matchers to handle complex-structured characters in a complex environment.

The main contributions of this paper can be briefly summarized as follows:

- We develop a modularized predictive coding model (MPCM) as a modular extension of a PCM that handles high-dimensional data by grouping them as subsystems.
- We apply the MPCM to a motion synthesis framework that introduces an intuitive solution combining a linear dynamic model with mocap data, so as to synthesize various motions of three different character models while improving the performance of both the controller and the physically implausible reference motion (e.g., hand penetration of the ground during a backflip motion).

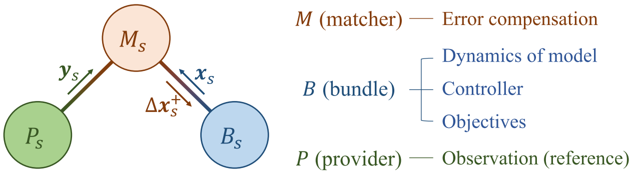


FIGURE 1. Graphic representation of the proposed framework. The basic structure is composed of two bundles and one matcher connecting them.

To validate the proposed framework, we depict various synthesized motions with a linear dynamic model and a small amount of mocap data of walking, kickboxing, backflipping, obstacle clearing, and grasping motions. In Section II, the closest related works proposed by other researchers are introduced for comparison with our work. In Section III, the construction of the modular-structured PCM is described. In Section IV, the online motion synthesis framework used to achieve motion imitation and synthesis tasks is explained. In Section V, the implementation details are explained. In Section VI, the experimental results used to evaluate the applicability of the proposed framework are described. In Section VII, the limitations of this study and scope for future work are discussed.

II. RELATED WORKS

In this section, we introduce the methods of previous works most similar to the method using dynamic models that consider environmental constraints and a composite of controllers that combines multiple existing controllers.

A. ENVIRONMENTAL CONSTRAINTS

To consider environmental constraints in motion synthesis, many online retargeting methods have been proposed. New motion representations have been introduced to synthesize natural interaction motions that preserve the original relationship while varying the environment [3]. A linear dynamic model has been used to guarantee natural motion against the environmental constraints that lead the synthesized motion different from various stylized example motions [12], [13]. Other researchers have used dynamic models based on simplified physics. An inverted pendulum model, in conjunction with mocap, has been widely used to represent human motion, especially for pelvis balancing [14], locomotion [15], running [16], interaction [17], and athletic motions [10]. Similarly, a point mass model was designed to generate a center-of-mass trajectory [8] while guaranteeing collision-free motion [9] in a multi-contact scenario. When utilizing such a dynamic model, our proposed framework can be beneficial in that the correspondence from the dynamic model to the mocap data is intuitively connected to apply various character models and represent various types of motion.

Environmental constraints have also been considered by aligning, blending, and sequencing motions according to the change in user control or the environment. Motion graphs generate a novel sequence of motions according to the constraints related to a user-specified path [1] or sketch [18]. Alignment and blending of example motions generate parameterized motions related to the target parameterization [19], [20]. Statistical models such as principal component analysis [21] or Gaussian process latent variable models [22] have been used to generate motion according to environmental constraints. The trajectory optimization method connects the mocap data to the simulated character to generate agile locomotion [23]. The recurrent neural network performs well in handling sequential data (or time-series data) and satisfies spatiotemporal constraints [2], [24]. We believe that the proposed framework enhances the performance of these previous works by compensating for the error online without any preprocessing or parameter tuning.

B. COMPOSITE OF CONTROLLERS

A composite of controllers was studied to combine existing controllers to improve their performance and achieve various tasks. Some methods used the linear interpolation of parameters to combine multiple controllers [25], [26]. However, a simple linear combination of controllers did not work well on the tracking controller without [27] or with a hand-crafted metric for blending [28], [29]. Subsequently, a low-dimensional dynamic model related to step length allowed the synthesis of locomotion that adapts to terrain variation [30]. Muico *et al.* proposed composite controllers to animate stylized interactive characters [31]. A composite controller generates multiple trajectories that provide more choices of adaptation to various interactive situations, but it requires assembly of the appropriate trajectories by hand and specification of blending weights. Similarly to a composite controller, the sequential composition of different skills was introduced to provide stylistic motions [32]. As in some previous works, our proposed framework also combines multiple tracking controllers. Moreover, our framework can work online while maintaining its environmental constraints.

III. MODULAR PREDICTIVE CODING FRAMEWORK

In contrast to a conventional PCM [11], we develop a modular-structured framework that efficiently models a high-dimensional state space of complex dynamic systems. The proposed framework enables the identification and control of a complex system by hierarchical combination of multiple modules, each of which manages a lower-dimensional dynamical system. It is particularly suitable when the objective dynamic environment consists of multiple modalities; for example, computer graphic characters with high DOFs and constraints (see Section IV). These advantages are derived from the compositions of multiple bundles connected online by matchers.

The proposed framework is generally composed of a provider and a bundle connected by matchers, as depicted in Figure 1. The construction of the proposed framework was inspired by the PCM in neuroscience, in which a brain tends to minimize the free-energy functional of its internal states based on observation [11]. In contrast to the original principle, we designed three components in a modular structure: a provider that detects the current observation and to which the current state should converge, a bundle that updates the internal states to predict the future state, and a matcher that evaluates the prediction error to complement an arbitrary provider-bundle pair.

Overall, the proposed framework updates the internal state of multiple bundles in a distributed manner. First, we consider a locally linear internal dynamic model

$$\dot{\mathbf{x}}_t = \mathbf{A}\mathbf{x}_t. \tag{1}$$

Here, the current state variable $\mathbf{x}_t$ is written as $\mathbf{x}_t = [\mathbf{x}_1, \cdots, \mathbf{x}_S]^T$, where T denotes its transpose. The system matrix $\mathbf{A}$ can be described as

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_{11} & \cdots & \mathbf{A}_{S1} \\ \vdots & \ddots & \vdots \\ \mathbf{A}_{1\,S} & \cdots & \mathbf{A}_{SS} \end{bmatrix},$$

where S denotes the number of subsystems. $\mathbf{x}_{s,t}$ and $\mathbf{A}_{ij}$, such that $i, j \in \{1, \cdots, S\}$, denote the current internal state variable of the s^{th} subsystem and the system matrix that links the states of the i^{th} and j^{th} subsystems, respectively. $\dot{\mathbf{x}}$ denotes the derivative of $\mathbf{x}$. The s^{th} subsystem corresponds to the s^{th} bundle, the state of which is a vector of arbitrary dimensionality (in our experiment, the internal state is a three-dimensional vector: a one-dimensional position and its first and second time derivatives). Note that this linearized formulation is shown for simplicity of explanation; extension to arbitrary non-linear and/or stochastic features is straightforward. Second, we consider the following block-diagonal approximation of the system matrix:

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_{11} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \mathbf{A}_{SS} \end{bmatrix}.$$

This approximation is based on the assumption of independence between the subsystems and helps to achieve numerically efficient solutions of the differential equation and/or control problem. However, this approximation is in many cases too strong an assumption that causes loss of important information. Third, we define an extended internal dynamic model to recover information without losing computational efficiency,

$$\dot{\mathbf{x}}'_t = \mathbf{A}'\mathbf{x}'_t, \tag{2}$$

where the extended state variable vector is represented by $\mathbf{x}' = [\mathbf{x}_1, \cdots, \mathbf{x}_S, \mathbf{x}_1^+, \cdots, \mathbf{x}_S^+]^T$, and the corresponding system matrix is defined as follows:

$$\mathbf{A}' = \begin{bmatrix} \mathbf{A}_{11} & \cdots & \mathbf{0} & \mathbf{A}_1^+ \\ \vdots & \ddots & \vdots & \vdots \\ \mathbf{0} & \cdots & \mathbf{A}_{SS} & \mathbf{A}_S^+ \end{bmatrix}.$$

The model of the s^{th} subsystem includes the additional hidden variable vector $\mathbf{x}_s^+$ with the corresponding linear dynamic system:

$$\dot{\mathbf{x}}_{s,t}^+ = \mathbf{A}_s^+ \mathbf{x}_{s,t}^+. \tag{3}$$

The state-space equation of the s^{th} subsystem is then extracted as

$$\dot{\mathbf{x}}_{s,t} = \mathbf{A}_{ss}\mathbf{x}_{s,t} + \mathbf{A}_s^+ \mathbf{x}_{s,t}^+. \tag{4}$$

Thus, the s^{th} subsystem's hidden variable $\mathbf{x}_{s,t}^+$ is equivalent to the control signal that controls the corresponding subsystem. The hidden variable is updated by integrating an offset parameter:

$$\mathbf{x}_{s,t}^+ = \mathbf{x}_{s,t-1}^+ + \Delta\hat{\mathbf{x}}_{s,t}^+, \tag{5}$$

where $\Delta\hat{\mathbf{x}}_{s,t}^+$ is the offset parameter that determines the current update of the hidden variable.

To compute the offset parameter, we devise a matcher to estimate the parameter with respect to the observation. Here, the observation is acquired by a provider for a more accurate estimation. The matcher searches an optimal offset parameter $\Delta\hat{\mathbf{x}}_{s,t}^+$ by minimizing the error between the state vector and the observation,

$$\Delta\hat{\mathbf{x}}_{s,t}^+ = \text{argmin}_{\Delta\mathbf{x}_{s,t}^+} \|f(\Delta\mathbf{x}_{s,t}^+) - \mathbf{y}_{s,t}\|^2, \tag{6}$$

where $\Delta\hat{\mathbf{x}}_{s,t}^+$ is the current optimal offset parameter, and $\mathbf{y}_{s,t}$ is the s^{th} subsystem's current observation acquired from a provider. The function f outputs a prediction of the current observation based on the bundle, which is derived by substituting $\Delta\mathbf{x}_{s,t}^+$ into $\Delta\hat{\mathbf{x}}_{s,t}^+$ in Equation (5) and then applying this to Equation (4), $f(\Delta\mathbf{x}_{s,t}^+) = (\mathbf{x}_{s,t-1} + \mathbf{A}^+)/(\mathbf{I} - \mathbf{A}\Delta t)$, such that $\dot{\mathbf{x}}_{s,t} = (\mathbf{x}_{s,t} - \mathbf{x}_{s,t-1})/\Delta t$, where Δt is the time step and $\mathbf{x}_{s,t-1}$ is the previous state of the s^{th} subsystem. The function $f(.)$ can be designed arbitrarily for a good prediction. At every time step, the offset parameters of every matcher are first updated based on Equation (6), and then the state vectors of every bundle are updated according to Equation (4). Algorithm 1 describes the procedure of the proposed framework. In Algorithm 1, $Pull(\mathbf{M}_s)$ pulls the current optimal offset parameter from the s^{th} matcher, and $Pull(\mathbf{B}_s, \mathbf{P}_s)$ pulls the current observation and current state from the s^{th} provider and s^{th} bundle, respectively. To show the reproducibility of our framework in cooperation with multiple bundles, we built a framework for motion analysis and synthesis.

IV. PREDICTIVE CODING-BASED ONLINE MOTION SYNTHESIS

We apply the proposed framework to generate character animation for computer graphics. In the application, we regard the state variables as the set of dynamic states of the characters, objects, etc. In contrast, the observation corresponds to the set of reference data (e.g., mocap). Figure 2 describes the conceptual diagram of the proposed framework for character motion synthesis. A provider extracts observations from the mocap data. A bundle includes a dynamic model and its controller to predict the future state by simulating the model according to the controller. A matcher minimizes the error between the observation and the predicted (simulated) state.

Algorithm 1 Modular Predictive Coding Framework

```
Require: the initial state $\mathbf{x}_{s,0}$ and $\mathbf{y}_{s,0}$, and the system matrices $\mathbf{A}_s$ and $\mathbf{A}_s^+$ are given.
function Bundle
    $\Delta\hat{\mathbf{x}}_{s,t}^+ \leftarrow Pull(\mathbf{M}_s)$
    $\mathbf{x}_{s,t}^+ \leftarrow \mathbf{x}_{s,t-1}^+ + \Delta\hat{\mathbf{x}}_{s,t}^+$
    $\dot{\mathbf{x}}_{s,t} \leftarrow \mathbf{A}_s\mathbf{x}_{s,t} + \mathbf{A}_s^+\mathbf{x}_{s,t}^+$
    $\mathbf{x}_{s,t+1} \leftarrow \mathbf{x}_{s,t} + \dot{\mathbf{x}}_{s,t}\Delta t$        ▷ $\Delta t$ indicates a time step
    return $\mathbf{x}_{s,t+1}$
end function
function Matcher
    $\mathbf{x}_{s,t}, \mathbf{y}_{s,t} \leftarrow Pull(\mathbf{B}_s, \mathbf{P}_s)$
    $\Delta\hat{\mathbf{x}}_{s,t}^+ \leftarrow argmin_{\Delta\mathbf{x}_{s,t}^+} \|f(\Delta\mathbf{x}_{s,t}^+) - \mathbf{y}_{s,t}\|^2$
    return $\Delta\hat{\mathbf{x}}_{s,t}^+$
end function
```

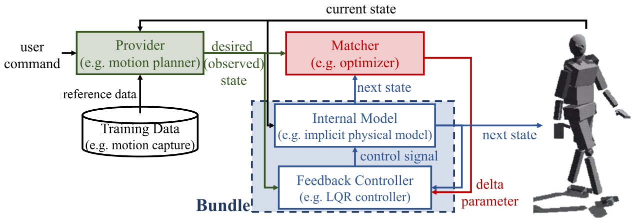


FIGURE 2. System diagram of the proposed framework. The proposed framework mainly consists of the provider (green), bundle (blue), and matcher (red). The user command is specified by the user, who controls the character's facing direction and speed.

A. PROVIDER

The provider utilizes mocap data to extract a virtually observed state, called an observation, as shown in Figure 2. The observation is updated by modifying a time stamp $\xi_{i,t}$.

$$\mathbf{y}_{i,t} = (\mathbf{Y}_{\xi_{i,t}})_{\xi_t = 1:N}, \tag{7}$$

where $\mathbf{y}_{i,t}$ indicates the state of the i^{th} provider at time t, $\mathbf{Y}$ denotes the repository of the mocap data, and N is the length of the mocap data. Because the update rule is controlled by the time stamp ξ_t, the provider emits a reference motion identical to the mocap data by incrementing the time stamp at each frame, thus achieving a motion imitation task. Varying the time stamp changes the speed of the synthesized motion.

To synthesize a new motion, such as an unlimited walking motion based on limited walking motion data, we use a motion graph technique [1] to rearrange the captured motion. Briefly, we segment the mocap data into motion fragments. The fragments are then rearranged and stitched according to the situation (e.g., character posture and user control). Figure 3 describes the sequence of time stamps in the tasks of motion imitation and motion synthesis.

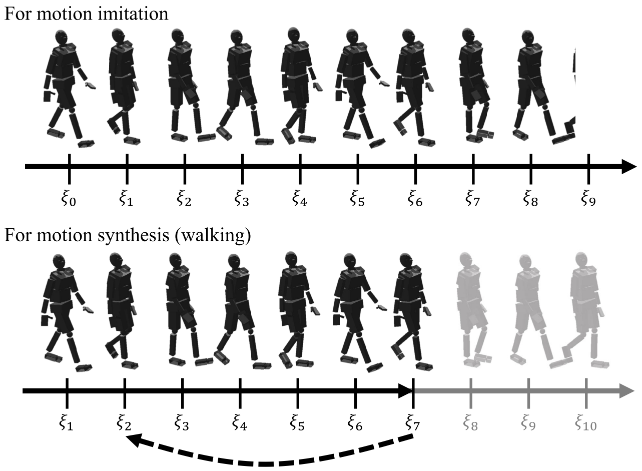

For motion imitation
ξ_0 ξ_1 ξ_2 ξ_3 ξ_4 ξ_5 ξ_6 ξ_7 ξ_8 ξ_9
For motion synthesis (walking)
ξ_1 ξ_2 ξ_3 ξ_4 ξ_5 ξ_6 ξ_7 ξ_8 ξ_9 ξ_{10}

FIGURE 3. Provider: For a motion imitation task, a provider selects the reference motion based on its original sequence. For a motion synthesis task (unlimited walking), the provider generates cyclic motion by rewinding the time stamp.

B. BUNDLE

We devise a bundle that contains an internal dynamic model and the corresponding optimal controller, and provides a synthesized motion as the output of the overall framework. To make a system invariant against the DOF, we defined the s^{th} bundle to represent the s^{th} DOFs of a character. Thus, the character motion can be reconstructed by concatenating the corresponding set of the bundle states, as shown in Figure 4. For an internal dynamic model that updates the state, we use a 1D mass–spring–damper model to represent each bundle that corresponds to a DOF. This model enforces the C1 continuity of the output motion, and the smoothness of the motion can be adjusted by the weight of the bundle. The dynamics of the model for the s^{th} DOF are described in the following state-space equation:

$$\dot{\mathbf{x}}_{s,t} = \mathbf{C}_s \mathbf{x}_{s,t} + \mathbf{D}_s \mathbf{u}_{s,t}, \tag{8}$$

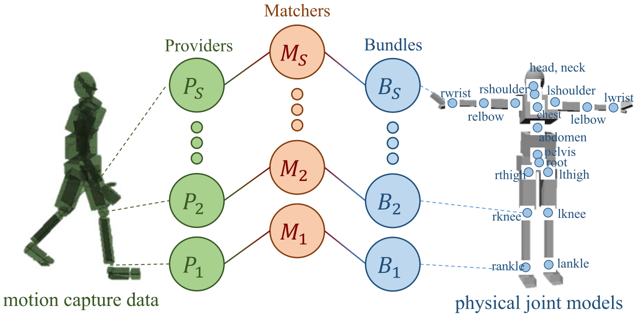


FIGURE 4. Motion representation. The green, blue, and red circles indicate providers, bundles, and matchers, respectively. Each provider and bundle corresponds to a character model's DOF to present their state.

where $\mathbf{x}_{s,t} \in \mathbb{R}^m$ and $\dot{\mathbf{x}}_{s,t} \in \mathbb{R}^m$ are the current model state and its derivative, respectively, that determine the position of the s^{th} DOF, which directly corresponds to the s^{th} subsystem in Section III. Here, we define the state that includes at most the m^{th} order of derivatives. In motion synthesis, $m = 3$. The model state is defined as $\mathbf{x} = [\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}, \ddot{\boldsymbol{\theta}}]$, where $\boldsymbol{\theta}$ is the joint angle, and $\dot{\boldsymbol{\theta}}$ and $\ddot{\boldsymbol{\theta}}$ are the first and second derivatives, respectively. $\mathbf{C}_s \in \mathbb{R}^{m\times m}$ and $\mathbf{D}_s \in \mathbb{R}^{m\times n}$ are the constant system matrices, determined by the spring coefficient, damping coefficient, and weight of each body joint (see details in [33]), and $\mathbf{u}_{s,t} \in \mathbb{R}^n$ is the s^{th} subsystem's current control signal. To allow the dynamic model to track the mocap data, an LQR was used as the optimal controller, modified with the additional parameterization of the desired state [10]. The LQR tracking control law is described as follows:

$$\mathbf{u}_{s,t} = -\mathbf{K}_{s,t}(\mathbf{x}_{s,t} - \mathbf{x}_{s,t}^{\text{desired}}), \tag{9}$$

where $\mathbf{K}_{s,t}$ is the current LQR solution matrix, computed based on the energy minimization of the current state and control, and $\mathbf{x}_{s,t}^{\text{desired}}$ is the current desired state. Based on the LQR tracking control law in Equation (9), the dynamics of the model are rewritten by substituting the control $\mathbf{u}_{s,t}$ into the law

$$\dot{\mathbf{x}}_{s,t} = \mathbf{C}_s \mathbf{x}_{s,t} - \mathbf{D}_s \mathbf{K}_{s,t}(\mathbf{x}_{s,t} - \mathbf{x}_{s,t}^{\text{desired}}). \tag{10}$$

Rearranging the above equation according to each variable clarifies the dynamic equation of the bundle.

$$\dot{\mathbf{x}}_t = \mathbf{C}'_s \mathbf{x}_{s,t} + \mathbf{D}'_s \mathbf{x}_{s,t}^{\text{desired}}, \tag{11}$$

where $\mathbf{C}'_s = (\mathbf{C}_s - \mathbf{D}_s \mathbf{K}_{s,t})$ and $\mathbf{D}'_s = -\mathbf{D}_s \mathbf{K}_{s,t}$ are given by the system matrices and LQR control law that minimizes the quadratic cost (see details in Section V). The form of this state-space equation corresponds to the original bundle dynamics, described in Equation (4). The system matrices $\mathbf{C}'_s$ and $\mathbf{D}'_s$ correspond to the subsystem matrices $\mathbf{A}_{ss}$ and $\mathbf{A}_s^+$, respectively, and $\mathbf{x}_{s,t}^{\text{desired}}$ is related to the subsystem's hidden variable. Here, $\mathbf{x}_{s,t}^{\text{desired}}$ indicates the current desired state, which is updated by integrating the offset parameter as follows:

$$\mathbf{x}_{s,t}^{\text{desired}} = \mathbf{x}_{s,t-1}^{\text{desired}} + \Delta\hat{\mathbf{x}}_{s,t}^{\text{desired}}, \tag{12}$$

where $\mathbf{x}_{s,t-1}^{\text{desired}}$ is the previous desired state. $\Delta\hat{\mathbf{x}}_{s,t}^{\text{desired}}$ is the offset parameter. The update rule in Equation (12) uses the previous parameter value, which is beneficial when the bundle represents time-series or sequential data.

C. MATCHER

In our general framework, a matcher defines a measure of prediction error as a function of state variables of a provider-bundle pair, and then provides the offset parameter to update the hidden variable.

For application to motion synthesis, we specifically devise a matcher for the pair of s^{th} provider and the s^{th} bundle, both corresponding to the s^{th} joint of the character $\mathbf{x}_s$. The prediction error is defined by

$$E_s = ||f(\Delta\mathbf{x}_{s,t}^{\text{desired}}) - \mathbf{y}_{s,t}||^2. \tag{13}$$

where $f(\Delta \mathbf{x}_{s,t}^{\text{desired}})$ is a prediction of the current observation based on the bundle dynamics using the current offset parameter $\Delta \mathbf{x}_{s,t}^{\text{desired}}$. The offset parameter $\Delta \mathbf{x}_{s,t}$ updates the bundle state $\mathbf{x}_{s,t}$ to minimize the prediction error:

$$\Delta \hat{\mathbf{x}}_{s,t}^{\text{desired}} = \arg \min_{\Delta \mathbf{x}_{s,t}^{\text{desired}}} E_s. \tag{14}$$

The optimal offset parameter $\Delta \hat{\mathbf{x}}_{s,t}^{\text{desired}}$ is fed back to the s^{th} bundle. The prediction error E_s is described in Equation 13. Based on the matcher, a higher-level one-frame optimization module, the proposed framework recovers the details of the motion online.

D. ENVIRONMENTAL CONSTRAINTS

Our proposed framework provides a reliable motion controller that synthesizes a natural motion, similar to that in the mocap data. However, the resultant motion can be unnatural when the environment changes because of user control or environmental effects. To maintain environmental constraints under such changes, we add an external force to the dynamics of the model. We first explain how each bundle independently handles its own portion of the external force, and then explain how the external force is distributed to many related bundles. Initially, the environmental constraints redefine the state-space equations of the dynamic model. When force acts on the character, the model reacts to the force while also satisfying the constraints based on its own dynamics. Based on the dynamic model, an external force term can be added to the model dynamics. Hence, we redefined the differential equation of the bundle in Equation (11) to include the environmental effect (external force) as follows:

$$\dot{\mathbf{x}}_{s,t} = \mathbf{C}'_s \mathbf{x}_{s,t} + \mathbf{D}'_s \hat{\mathbf{x}}_{s,t}^{\text{desired}} + \mathbf{f}_{\text{env}}/\varpi_s, \tag{15}$$

where $\mathbf{f}_{\text{env}}$ indicates the environmental force, and ϖ_s denotes the weight of the s^{th} bundle. Here, the additional force is determined by various constraints that occur owing to forces from the environment: external force, collision with an obstacle, and ground friction. The environmental force can be described as follows:

$$\mathbf{f}_{\text{env}} = f_{\text{ext}} + f_{\text{col}} + f_{\text{footskate}}, \tag{16}$$

where f_{ext} is the external force, f_{col} is the force from collision with an obstacle, and $f_{\text{footskate}}$ is the force incurred by changes owing to user control (e.g., a situation in which only a straight walking motion is used when synthesizing a turning motion). Below, we explain how each force is computed to satisfy the environmental constraints.

1) EXTERNAL PERTURBATION

Figure 5 depicts a synthesized reaction to an external force specified by the user. To synthesize a smooth and natural-looking reaction, the external force needs to affect the corresponding body part as well as its neighboring body parts. To distribute the force in Cartesian space to the space of joint angles, we use a weighted Jacobi method. For example, when applying an external force on a character's chest, the force affects the bundles corresponding to the chest, abdomen, and pelvis along with the hierarchy of joints of the character model. Each weight is decided based on the skeleton structure; a joint closer to the root joint is less affected by the external force, which is tuned based on the angle bound of each joint. Figure 5 depicts a recovery motion when applying an external force to the chest. In the figure, the magnitude of the external force (depicted by the red arrow) is specified by the user. Note that this force-distribution scheme is empirical and lacks accurate physical meaning. Nevertheless, the resulting motion appears physically plausible, and the virtual force directly affects the angular acceleration of the joints.

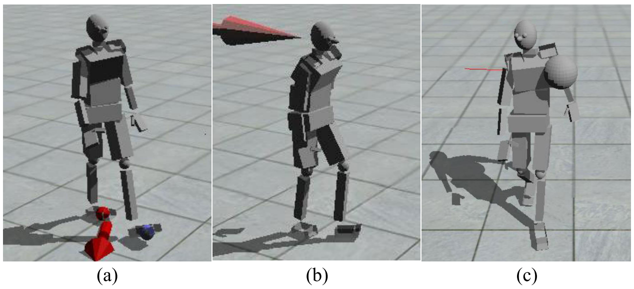


FIGURE 5. Environmental constraints. Three different types of constraint related to the type of motion are considered: ground contact (a), virtual force (b), and object collision (c).

When the motion is constrained by the environment, the time stamp is instructed to wait until the recovery motion ends (when the current posture has returned to the original posture from before the force was applied). This is because when an external force is applied to a specific bundle (one of the character model DOFs), the state of the dynamic model becomes distant from the observation. The speed of the time stamp increment then decreases until the difference between the model state of the bundle and the observation is lower than a threshold.

2) COLLISION RESPONSE

As shown in Figure 5, the collision response can be achieved by considering the reaction force acting on the character. The reaction force is computed by the collision normal and the penetration depth between the obstacle and the character. The reaction direction and penetration depth determine the direction and magnitude of the reaction force, respectively. The reaction force is applied to the dynamic system in the same manner as the external force in Equation (15). Figure 5(c) depicts the result of the collision response.

3) FOOTSKATE CLEARING

To clear footskate, we apply the force derived from foot location changes. Computing the force starts with the vector of the previous foot location and the current foot location when the foot makes contact with the ground. Note that the foot contact information is given with the mocap data.

$$v_{\text{footskate}} = \mathbf{wJ}(\mathbf{x}_{\text{foot},t-1} - \mathbf{x}_{\text{foot},t}), \tag{17}$$

where $v_{\text{footskate}}$ is the footskate vector, $\mathbf{J}$ is a Jacobian matrix computed based on the human model in Figure 6, $\mathbf{w}$ is an equally distributed weight parameter (e.g., $\mathbf{w} = [0.25, 0.25, 0.25, 0.25]$ and $\sum \mathbf{w} = 1$), and $\mathbf{x}_{\text{foot},t}$ and $\mathbf{x}_{\text{foot},t-1}$ are the current and previous foot location, respectively. The external force related to the footskate is computed based on the footskate vector, as with the collision response.

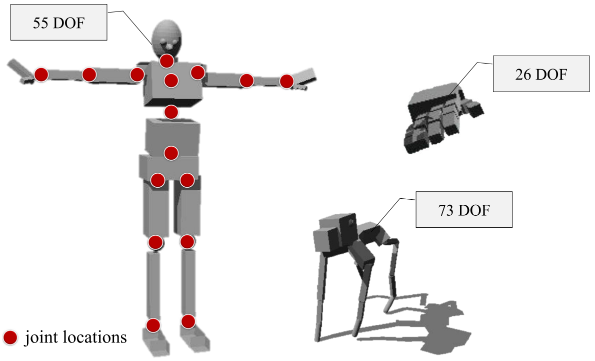


FIGURE 6. Character models. In our experiment, we use three different character models: a human, a hand, and a dog.

V. IMPLEMENTATION

In this section, we explain how the predictive coding-based online motion synthesis was implemented. In our experiments, a human character model has 19 DOFs represented by 19 bundles, consisting of 19 dynamic models and 19 controllers.

Initially, all states in all bundles are initialized; the state of the dynamic model is initially set with values corresponding to an initial character posture in the mocap data. At each time step, every matcher is updated, and then every bundle is updated in parallel so that the framework allows parallel processing. In the case of our mass–spring–damper model, the spring coefficient and damping coefficient are set to 1 and 0.3, respectively. The weight of each bundle is set to 4.0. The overall smoothness and tracking latency can be adjusted simply by adjusting the bundle weight while fixing the coefficients. The weight setting is further discussed in Section VI-C. While updating the bundle, the LQR controller computes the optimal gain matrix $\mathbf{K}_s$ (described in Equation (9)), acquired by minimizing the objective function $L_s(\mathbf{x}_s, \mathbf{u}_s) = \mathbf{x}_s{}^T Q \mathbf{x}_s + \mathbf{u}_s^T R \mathbf{u}_s$, where $\mathbf{u}_s$ is the control signal and $\mathbf{x}_s$ is the current state of the s^{th} bundle. The parametric matrices Q and R are set to $Q = \text{diag}(10000, 0.01)$ and $R = 1$ (see details in [34]). The character's posture is represented as a vector $\boldsymbol{\phi} = [\boldsymbol{\theta}_r, \boldsymbol{\theta}_q, \boldsymbol{\theta}_1, \cdots, \boldsymbol{\theta}_S]$. where $\boldsymbol{\theta}_r$ and $\boldsymbol{\theta}_q$ are the root position and orientation, respectively. $\boldsymbol{\theta}_1, \cdots, \boldsymbol{\theta}_S$ denotes the local joint angles relative to the root.

To test the performance of the matcher, we compared two optimization algorithms: gradient descent (GD) and conjugate gradient (CG). Figure 7 depicts the results of using GD and CG (refer to the supplementary video clip). From comparison of CG and GD, CG clearly outperformed GD in highly dynamic motion, such as human walking motions or dog jumping motions. However, the CG costs twice the computation time. We obtained similar performances in the object-grasping motion (not shown for simplicity).

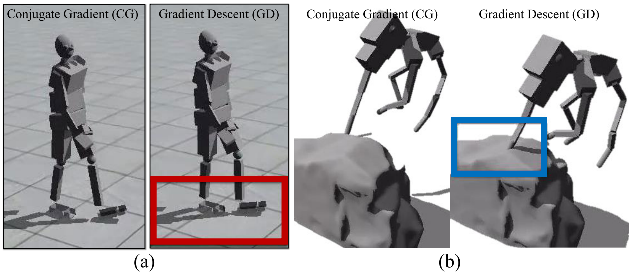


FIGURE 7. Comparison of optimizers: conjugate gradient (CG) and gradient descent (GD). In walking motion, GD occurs in foot-slip (a), shown in the red square, and collision while clearing an obstacle (b), shown in the blue square.

The hardware used for the implementation was an Intel Core i7-8700K CPU and GeForce GTX 1080 Ti. Computation time differed according to the optimization method used. Table 1 describes the computation time of the proposed framework for motion synthesis per frame. Here, we specified the time at which all of the bundles and matchers were computed.

TABLE 1. Time complexity of the optimizers (ms: milliseconds).

Model	DOF	GD	CG
DOG	73 DOF	10.502 ms	55.791 ms
HUMAN	55 DOF	6.516 ms	39.454 ms
HAND	26 DOF	2.040 ms	17.088 ms

VI. EXPERIMENTAL RESULTS

The proposed framework improves system performance for motion synthesis and utilizes a dynamic model and a small amount of motion data. We performed three experiments to validate the proposed framework and compare the performance with previous frameworks.

For comparison, we included similar frameworks that also use a dynamic model and mocap data. The compared frameworks are described in Figure 8. As shown in Figure 8(b), the previous work was devised for a tracking controller that tracks the mocap data [35]. In that case, footskate occurred owing to the tracking delay (shown in Figure 8(e)). Even when we included a constraint module at the end of the framework (shown in Figure 8(c)), the constraint and delay enlarged the unnaturalness of the resulting motion, as shown in Figure 8(f). As described in Figure 8(f), the proposed method reduced the tracking delay to guarantee more natural motion. We further compared the proposed method with the state-of-the-art synthesis method used in [7], [36]. The previous synthesis method learns various locomotion styles to estimate the future trajectory of the footstep by adapting to the current situation. However, the generated motion

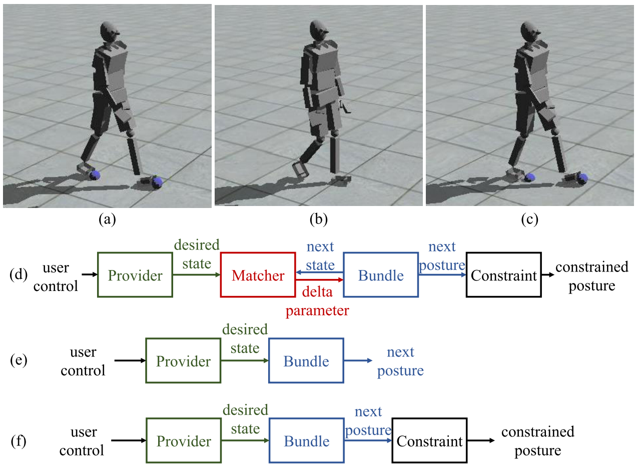


FIGURE 8. Comparison with previous frameworks: the proposed framework leads to reduced motion delay that allows natural constraint adaptation. The proposed framework (d) generates natural motion when applying additional constraints (a). In addition, the previous framework (e) and the constrained one (f) show footskate (b) and unnatural swing foot motion (c), respectively.

may be unnatural, especially when the desired motion is separated from the motion represented by the mocap data. In Figure 9(a), we can see that the footskate occurred when the user control was set to walking slowly. In contrast, the proposed method can filter out the unnatural footskate by considering the physical constraints as shown in Figure 9(b).

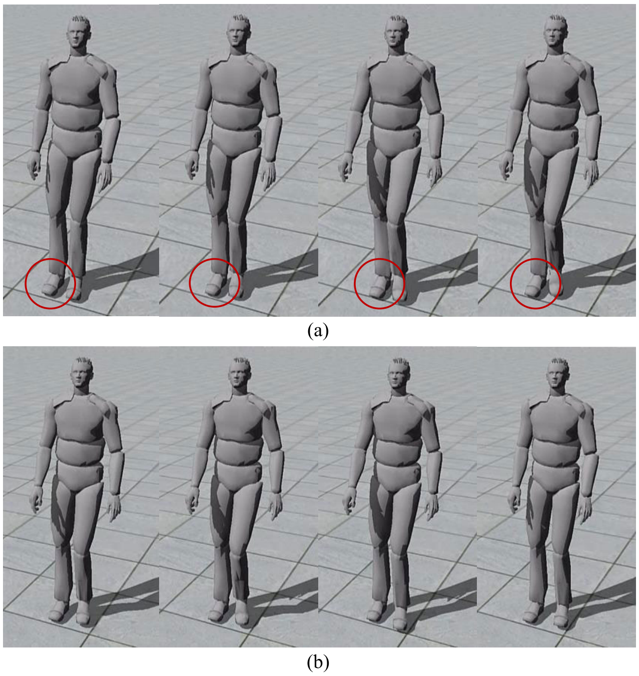


FIGURE 9. Comparison with the previous synthesis method: the red circles in (a) depict the footskate that the previous method generates when synthesizing locomotion at low speed. (b) The proposed method suppresses the anomalies of physical realism.

For detailed verification, we performed two different tasks: motion imitation with three different characters, and motion synthesis for locomotion and force reaction. The motion imitation task was to track the example motion, and the motion synthesis task was to track the motion generated by rearranging the motion fragments based on the motion graph, as proposed by Kovar *et al.* [1]. In these experiments, we used a limited amount of mocap data, as described in Table 2.

TABLE 2. Mocap data (fr.: frame, s: second).

Model	Name	Description	# of fr. (time)
DOG	obstacle clearing	backward stepping, walking to running, jumping	384 fr. (12.8 s)
HUMAN	walking	standing, forward walking	105 fr. (3.5 s)
	kickboxing	kicking and punching	294 fr. (9.8 s)
	backflip	backflipping	102 fr. (3.4 s)
HAND	grasping	approaching and grasping	150 fr. (5.0 s)

A. MOTION IMITATION WITH VARIOUS CHARACTER MODELS

In this section, we report the results of a motion imitation task performed using three different character models: a human, a dog, and a hand (as depicted in Figure 6). In Figure 6, each joint of the human model is composed of a three-dimensional rotational joint, except for one-dimensional joints at the elbows, knees, and toes. The dog model consists of 22 three-dimensional rotational joints, and the hand model has five two-dimensional rotational joints and ten one-dimensional finger joints. Each model has an additional root joint that defines a global coordinate frame using six DOFs for the root position and orientation.

We conducted motion imitation that allows the bundles to track the mocap data. This is achieved by having the provider select the reference motion at each time sequence based on the original mocap sequence. As the state of the provider varies, the bundles start tracking the observed state. Figure 10 shows that the proposed framework successfully mimicked the mocap data relative to the previous work. The systems of the baseline and proposed methods are described in Figure 8. Analyzing the figure, we compared the motion imitation task based on the proposed and the baseline framework, using the LQR tracking controller that was used in several previous works [27], [35].

Figure 11 depicts the tracking trajectory for the walking motion described in Figure 10(a). The figure shows three trajectories acquired from the mocap data, the baseline method, and the proposed method. Comparing the trajectories, the baseline method generated a delayed trajectory compared to that from the proposed method, as shown in Figure 11. However, this result may vary according to parameter settings, such as the weight of the dynamic model. In Section VI-C, further analysis is conducted considering the parameter settings.

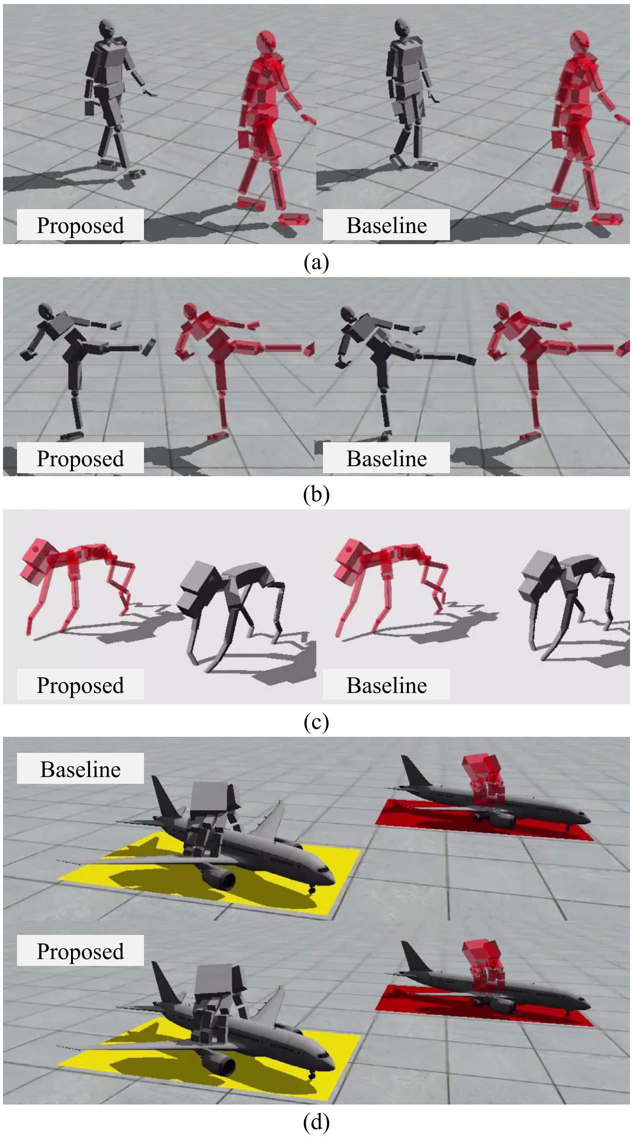


FIGURE 10. Motion imitation task. Compared to the baseline described in Figure 8, the proposed framework outperforms others when imitating the example motion. The red-colored character represents the mocap data. Walking (a) and kickboxing (b) motions by the human model, nozzle-clearing motion (c) by the dog model, and object-grasping motion (d) by the hand model were used.

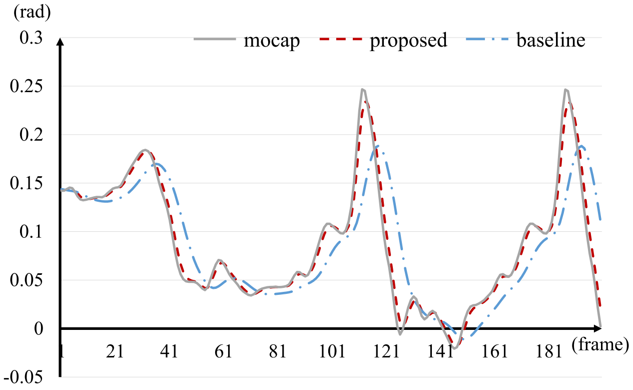


FIGURE 11. Comparison with the baseline method: gray, blue, and red lines denote mocap data, a baseline LQR result, and the proposed result, respectively, subject to the right leg joint angle trajectories.

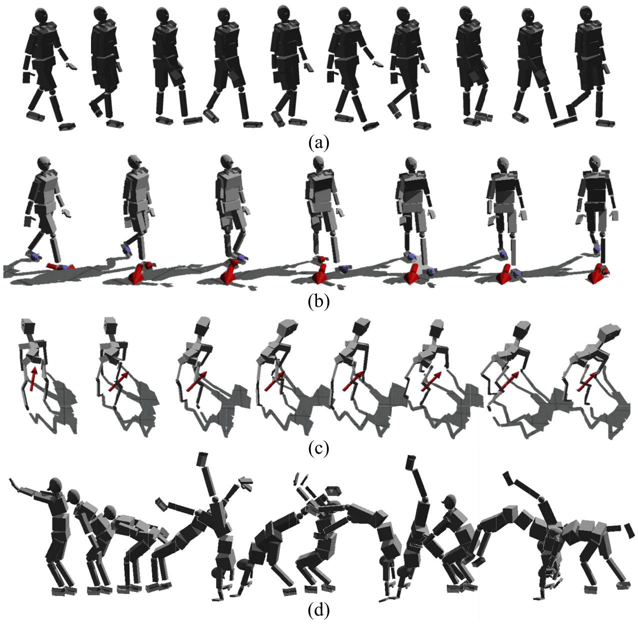


FIGURE 12. Motion synthesis. The proposed framework synthesizes the unlimited walking (a) and backflip motions (d). The character can also be controlled by a user (b and c).

B. MOTION SYNTHESIS

Motion synthesis is achieved by using the motion graph segment to stitch a small amount of straight walking motion and then generate a walking motion for an unlimited time. As shown in Figures 12 (a) and (d), unlimited walking and backflip motions can be generated by sequencing the mocap data of a single cycle of walking motion and one backflip motion.[1] Editing the offset of the root position and facing direction depicted by the red arrow in Figures 12(b) and (c) moves the character in a user-specified direction and at a user-specified speed. Speed control is made possible by controlled warping of the timing of the provider to integrate the time stamp more quickly or slowly. (In our experiment, 0.5-2.0 was suitable.) Figure 13 shows the synthesized results that animate both human and dog walking motions at different speeds. In the same manner as above, the grasping speed can be controlled, which is an important factor in real-time object-manipulation tasks [35].

The motion graph utilizes the mocap data to allow adaptation to changes. However, when there is no close transition between frames, the synthesized motion becomes unnatural. Referring to the mocap data for a dog in Table 2, there is no walking motion. Thus, the transition gap when constructing a walking cycle is large. The proposed framework found the appropriate control signal to generate a motion similar to the mocap data while satisfying the environmental constraints

[1] The generated motions are also available at https://drive.google.com/file/d/1AzYSxovZqxuFsJysukZsLWBUYMjZSZI9/view?usp=sharing.

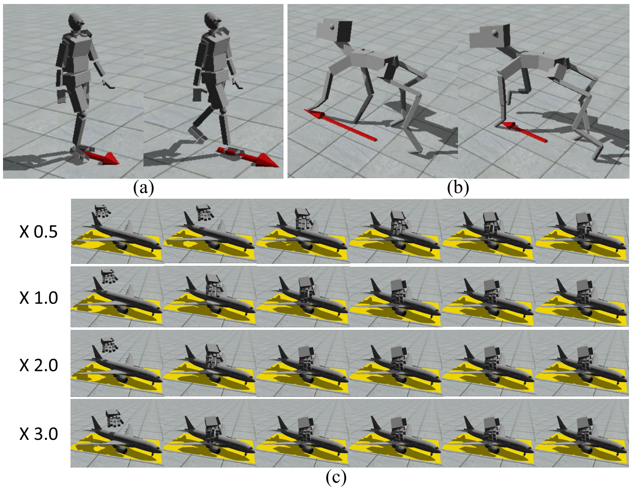


FIGURE 13. Speed variation: changing the time stamp in the provider changes the walking speed of the human (a) and dog (b). The speed of the grasping motion determines the timing when the hand grasps the object (c).

that the character should maintain. Figure 14 describes the comparison between the motion-graph-based method and the proposed motion synthesis method. The motion-graph-based method searches for transitions in mocap data to synthesize a motion. However, the method exhibited visual artifacts while synthesizing a dog's straight walking motion because no straight walking motion was included in the given pool of mocap data. Our proposed framework recovered such small artifacts and could construct more transition paths owing to loosening of the threshold of the transition metric.

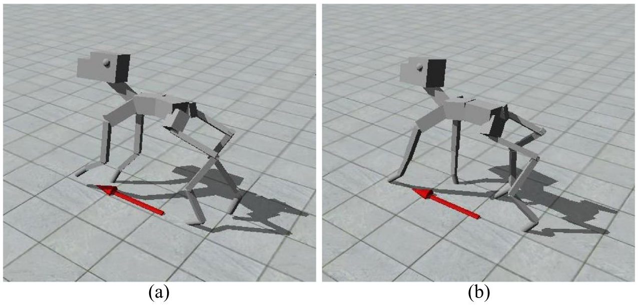


FIGURE 14. Comparison between a motion-graph-based method and the proposed framework. The motion-graph-based method depicts unrealistic motions such as penetration of the ground and jerky motions (a). The proposed framework satisfies the environmental constraints that guarantee a more realistic motion (b).

C. PARAMETERIZATION

In this section, we show the robustness of our framework against small changes in the parameter setting of the proposed framework. Setting the appropriate model parameter is difficult because some parameters may have a tradeoff. For example, if the weight of the bundle is too heavy, the synthesized motion becomes very smooth, but a tracking delay appears. Our proposed framework handles this issue by including a higher-level one-frame optimization mechanism that compensates the error based on the observation.

To analyze the robustness in detail, different weight parameters were tested. Table 3 presents the discontinuity and delay levels according to the total weight. Similarly to the works in [37], we evaluate the discontinuity of the motion as follows:

$$v_{\text{discon}} = (1/T)\Sigma_{t=1}^{T}\|\mathbf{x}_{t-1} - 2\mathbf{x}_t + \mathbf{x}_{t+1}\|^2, \tag{18}$$

TABLE 3. Time complexity of our framework (disc.: discontinuity).

Total Weight	Baseline		Proposed	
	disc.	delay	disc	delay
2.7	0.265	0.020	0.248	0.014
18	0.158	0.112	0.256	0.019
36	0.123	0.283	0.256	0.019
54	0.102	0.426	0.259	0.023
72	0.088	0.542	0.259	0.023

where v_{discon} is the discontinuity level. t and T indicate the frame number and the total number of frames in the motion, respectively, that are used to measure the discontinuity. $\mathbf{x}$ denotes the model state. The delay level is measured by the Euclidean distance from the reference motion:

$$v_{\text{delay}} = (1/T)\Sigma_{t=1}^{T}\|\mathbf{x}_t - \mathbf{y}_t\|^2, \tag{19}$$

where v_{delay} indicates the delay level of the motion that can also be interpreted to be similar to the mocap data.

In Table 3, the baseline with low weight bundles depicts a similar tracking performance in the delay and discontinuity levels for the motion imitation task. In addition, as shown in Figure 15(b), using low-weight bundles (total weight 2.7) results in less recovery time. However, as shown in the upper picture of Figure 15(a), such a low weight causes a jerky recovery motion against the external force. When the total weight is set to 72.0, a smooth recovery appears, as described in the lower picture of Figure 15(a). To clarify the comparison, we draw motion trajectories for the character's chest in Figure 15(b). The proposed framework improves the tracking performance to increase its robustness and generate a smooth reaction against the external force.

D. EXPERIMENTAL FULL-BODY SIMULATION

As a preliminary experiment, we fed the synthesized motion to a forward dynamics simulator to determine whether our framework could be applied to difficult problems such as full-body simulation. We simulated a walking motion by designing the provider with the unlimited motion graph. Here, the motion synthesis step was implemented at 120 Hz and the forward dynamics simulation was performed at 3000 Hz. The torques for all the joints, excluding the unactuated root joint, were computed using a proportional derivative servo as follows:

$$\boldsymbol{\tau} = k_p(\boldsymbol{\phi}_d - \boldsymbol{\phi}) - k_d(\dot{\boldsymbol{\phi}}_d - \dot{\boldsymbol{\phi}}), \tag{20}$$

where $\boldsymbol{\phi}$ and $\dot{\boldsymbol{\phi}}$ denote the current character posture and the derivative of the current posture, respectively. k_p and k_d were set to 3000 and 30. $\boldsymbol{\phi}_d$ and $\dot{\boldsymbol{\phi}}_d$ indicate the desired character

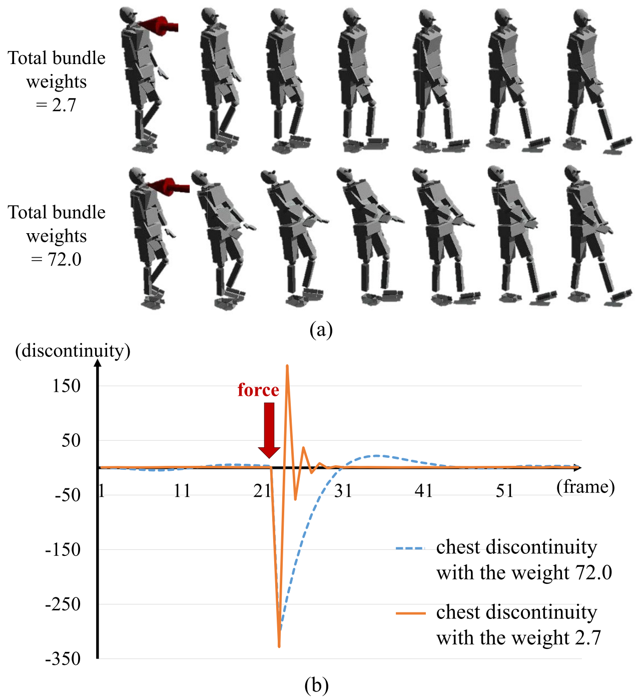


FIGURE 15. External perturbation. The red arrow indicates the external force. A discontinuous reaction occurs when the total weight of the bundles is 2.7, as depicted in the upper picture of (a). When the total weight is set to 72.0, a smooth recovery is generated, as depicted in the lower picture of (a). Motion trajectories for the chest are drawn to distinguish the difference between the recovery motions (b). The orange line is the case of total weight 2.7, and the blue line is the case of total weight 72.

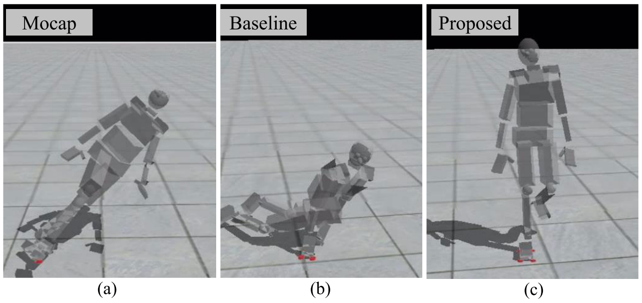


FIGURE 16. Full-body simulation for a motion imitation task. The red spheres represent contact forces from the ground. These figures are captured in 1.5 s while simulating an unlimited walking motion based on mocap (a), the baseline framework (b), and the proposed framework (c).

posture and its derivative, respectively. The desired character posture and its derivatives for the $i+1^{\text{th}}$ motion synthesis step were computed by first updating the bundle states using the simulated state corresponding to the i^{th} motion frame, and then single-stepping all the bundles. $\boldsymbol{\tau}$ indicates the torque acting on the joints of the character model, which becomes the input of the forward dynamics. Interestingly, this simple scheme outperformed two other methods: mocap tracking and baseline framework. In the mocap tracking experiment, $\boldsymbol{\phi}_d$ and $\dot{\boldsymbol{\phi}}_d$ are directly from the captured motions. In the baseline framework, $\boldsymbol{\phi}_d$ and $\dot{\boldsymbol{\phi}}_d$ are from the baseline framework. Without any balancing mechanism, the character eventually falls down after 92 frames (3.07 s) using our framework, compared to 61 frames (mocap) and 35 frames (baseline), as shown in Figure 16. Regarding the result of the baseline, a bad prediction from the baseline system can cause even worse tracking performance than that of the system without a prediction module. However, this type of tracking performance is unexpectedly good for a method that uses such simple dynamics of the bundles, and thus we plan to explore new balancing controllers in further work.

VII. CONCLUSION AND DISCUSSION

Herein, we proposed an online motion synthesis framework motivated by a PCM. Specifically, we devised an architecture consisting of multiple triplets of bundles, providers, and matchers, which increases the motion prediction performance of the system. Hierarchical modeling compensated for the difference between the mocap data and the dynamic model so that the system generates the motion, similarly to the motion data, while satisfying the constraints of the model. In addition, the combination of the bundle and the provider utilized a small amount of mocap data to reduce unnatural effects such as discontinuity, ground penetration, and foot-skate, as described in the experimental results. By defining bundles and matchers independently, the motions for multiple character models were represented and synthesized while reducing their tracking delay online.

The proposed framework has some limitations that we expect to address in future work. The motion quality depends on the design of each bundle. Considering a bundle, we used a bundle dynamic model, LQR optimal controller, and conjugate gradient optimizer. The physical model guarantees smooth motion and successfully reflects the joint recovery adaptation to an external force or collision. However, independence between bundles can occasionally produce a physically implausible recovery motion in response to large external forces (refer to the supplementary video clip).

In future work, utilizing more complex physical models, such as the inverted pendulum model [10] and the abstract model [6], would enable us to consider more complicated environmental constraints to provide better physical realism.

To expand the types of synthesized motions, different models such as deep-learning networks, statistical models, and other non-linear interpolation schemes may estimate the proper motion according to the current situation while covering a wider variety of motion styles (e.g., agile motion) [7], [38]. However, in this study, we focused on the generality of the proposed framework and evaluated it based on its naturalness, tracking delay, and smoothness for various examples of motion synthesis.

REFERENCES

[1] L. Kovar, M. Gleicher, and F. Pighin, "Motion graphs," *ACM Trans. Graph.*, vol. 21, no. 3, pp. 1–10, 2002.

[2] K. Lee, S. Lee, and J. Lee, "Interactive character animation by learning multi-objective control," in *Proc. SIGGRAPH Asia*, 2018, p. 180.

[3] T. Jin, M. Kim, and S.-H. Lee, "Aura mesh: Motion retargeting to preserve the spatial relationships between skinned characters," *Comput. Graph. Forum*, vol. 37, no. 2, pp. 311–320, May 2018.

[4] X. Zhao, M. G. Choi, and T. Komura, "Character-object interaction retrieval using the interaction bisector surface," *Comput. Graph. Forum*, vol. 36, no. 2, pp. 119–129, May 2017.

[5] S. Guo, R. Southern, J. Chang, D. Greer, and J. J. Zhang, "Adaptive motion synthesis for virtual characters: A survey," *Vis. Comput.*, vol. 31, no. 5, pp. 497–512, May 2015.

[6] Y. Ye and C. K. Liu, "Optimal feedback control for character animation using an abstract model," *ACM Trans. Graph.*, vol. 29, no. 4, p. 74, 2010.

[7] J. Hwang, J. Kim, I. H. Suh, and T. Kwon, "Real-time locomotion controller using an inverted-pendulum-based abstract model," *Comput. Graph. Forum*, vol. 37, no. 2, pp. 287–296, May 2018.

[8] C. Kang and S.-H. Lee, "Multi-contact locomotion using a contact graph with feasibility predictors," *ACM Trans. Graph.*, vol. 36, no. 2, p. 22, 2017.

[9] S. Tonneau, P. Fernbach, A. D. Prete, J. Pettré, and N. Mansard, "2PAC: Two-point attractors for center of mass trajectories in multi-contact scenarios," *ACM Trans. Graph.*, vol. 37, no. 5, p. 176, 2018.

[10] T. Kwon and J. K. Hodgins, "Momentum-mapped inverted pendulum models for controlling dynamic human motions," *ACM Trans. Graph.*, vol. 36, no. 1, p. 10, 2017.

[11] K. Friston, "The free-energy principle: A unified brain theory?" *Nature Rev. Neurosci.*, vol. 11, no. 2, p. 127, 2010.

[12] E. Hsu, K. Pulli, and J. Popović, "Style translation for human motion," in *Proc. ACM SIGGRAPH Papers*, 2005, pp. 1082–1089.

[13] S. Xia, C. Wang, J. Chai, and J. Hodgins, "Realtime style transfer for unlabeled heterogeneous human motion," *ACM Trans. Graph.*, vol. 34, no. 4, pp. 1–10, Jul. 2015.

[14] J. Hwang, I. H. Suh, G. Park, and T. Kwon, "Human character balancing motion generation based on a double inverted pendulum model," in *Proc. 10th Int. Conf. Motion Games. ACM*, 2017, p. 11.

[15] Y.-Y. Tsai, W.-C. Lin, K. B. Cheng, J. Lee, and T.-Y. Lee, "Real-time physics-based 3D biped character animation using an inverted pendulum model," *IEEE Trans. Vis. Comput. Graphics*, vol. 16, no. 2, pp. 325–337, Mar. 2010.

[16] T. Kwon and J. Hodgins, "Control systems for human running using an inverted pendulum model and a reference motion capture sequence," in *Proc. ACM SIGGRAPH/Eurographics Symp. Comput. Animation Assoc.*, 2010, pp. 129–138.

[17] J. Hwang, I. H. Suh, and T. Kwon, "Editing and synthesizing two-character motions using a coupled inverted pendulum model," *Comput. Graph. Forum*, vol. 33, no. 7, pp. 21–30, Oct. 2014.

[18] J. Lee, J. Chai, P. S. A. Reitsma, J. K. Hodgins, and N. S. Pollard, "Interactive control of avatars animated with human motion data," *ACM Trans. Graph.*, vol. 21, no. 3, pp. 491–500, Jul. 2002.

[19] L. Kovar and M. Gleicher, "Automated extraction and parameterization of motions in large data sets," *ACM Trans. Graph.*, vol. 23, no. 3, pp. 559–568, Aug. 2004.

[20] H. J. Shin and H. S. Oh, "Fat graphs: Constructing an interactive character with continuous controls," in *Proc. ACM SIGGRAPH/Eurographics Symp. Comput. Animation Assoc.*, 2006, pp. 291–298.

[21] J. Min and J. Chai, "Motion graphs++: A compact generative model for semantic motion analysis and synthesis," *ACM Trans. Graph.*, vol. 31, no. 6, p. 153, 2012.

[22] K. Grochow, S. L. Martin, A. Hertzmann, and Z. Popović, "Style-based inverse kinematics," in *Proc. ACM SIGGRAPH Papers*, 2004, pp. 522–531.

[23] U. Muico, Y. Lee, J. Popović, and Z. Popović, "Contact-aware nonlinear control of dynamic characters," *ACM Trans. Graph.*, vol. 28, no. 3, p. 81, 2009.

[24] Y. Lee, S. Kim, and J. Lee, "Data-driven biped control," *ACM Trans. Graph.*, vol. 29, no. 4, p. 129, 2010.

[25] M. van de Panne, R. Kim, and E. Fiume, "Synthesizing parameterized motions," in *Proc. 5th Eurographics Workshop Animation Simulation*, Oslo, Norway, 1994.

[26] K. Yin, K. Loken, and M. Van de Panne, "SIMBICON: Simple biped locomotion control," *ACM Trans. Graph.*, vol. 26, no. 3, p. 105, 2007.

[27] M. Da Silva, F. Durand, and J. Popović, "Linear Bellman combination for control of character animation," *ACM Trans. Graph.*, vol. 28, no. 3, p. 82, 2009.

[28] K. W. Sok, M. Kim, and J. Lee, "Simulating biped behaviors from human motion data," *ACM Trans. Graph.*, vol. 26, no. 3, p. 107, Jul. 2007.

[29] T. Erez and W. D. Smart, "Bipedal walking on rough terrain using manifold control," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2007, pp. 1539–1544.

[30] D. Holden, T. Komura, and J. Saito, "Phase-functioned neural networks for character control," *ACM Trans. Graph.*, vol. 36, no. 4, p. 42, 2017.

[31] U. Muico, J. Popović, and Z. Popović, "Composite control of physically simulated characters," *ACM Trans. Graph.*, vol. 30, no. 3, p. 16, 2011.

[32] A. Aristidou, E. Stavrakis, M. Papaefthimiou, G. Papagiannakis, and Y. Chrysanthou, "Style-based motion analysis for dance composition," *Vis. Comput.*, vol. 34, no. 12, pp. 1725–1737, Dec. 2018.

[33] F. Haugen, *Exercises to Basic Dynamics and Control*. TechTeach, 2010.

[34] A. Bogdanov, "Optimal control of a double inverted pendulum on a cart," Oregon Health Sci. Univ., OGI School Sci. Eng., Beaverton, OR, USA, Tech. Rep. CSE-04-006, 2004.

[35] J. Hwang, K. Kim, I. H. Suh, and T. Kwon, "Performance-based animation using constraints for virtual object manipulation," *IEEE Comput. Graph. Appl.*, vol. 37, no. 4, pp. 95–102, 2017.

[36] T. Kwon, Y. Lee, and M. Van De Panne, "Fast and flexible multilegged locomotion using learned centroidal dynamics," *ACM Trans. Graph.*, vol. 39, no. 4, pp. 46–51, Jul. 2020.

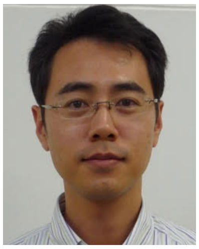
[37] J. Chai and J. K. Hodgins, "Performance animation from low-dimensional control signals," *ACM Trans. Graph.*, vol. 24, no. 3, pp. 686–696, Jul. 2005.

[38] Y. Li, T. Wang, and H. Y. Shum, "Motion texture: A two-level statistical model for character motion synthesis," in *Proc. 29th Annu. Conf. Comput. Graph. Interact. Techn.*, 2002, pp. 465–472.

JAEPYUNG HWANG received the Ph.D. degree from the Department of Computer Software, Hanyang University. He is currently a specially appointed as an Assistant Professor with the Graduate School of Informatics, Kyoto University, Kyoto, Japan. His research interests include artificial intelligence, machine learning, human–robot interaction, computer graphics, and interactive character animation.

SHIN ISHII received the B.E., M.E., and Ph.D. degrees from The University of Tokyo, in 1986, 1988, and 1997, respectively. He was an Associate Professor with the Nara Institute of Science and Technology, Nara, Japan, in 1997, and a Professor, in 2001. He has been a Full Professor with the Graduate School of Informatics, Kyoto University, Kyoto, Japan, since 2007, and the Director of ATR Neural Information Analysis Laboratories, Kyoto, since 2018. His research interests include machine learning, computational neuroscience, and intelligent robotics.

TAESOO KWON received the Ph.D. degree from the Department of Computer Science, Korea Institute of Science and Technology, in 2007. He is currently with the Department of Computer Software Engineering, Hanyang University, Seoul. His research interests include computer animation, physics-based character animation, and machine learning.



SHIGEYUKI OBA received the Ph.D. degree from the Department of Information Science, Nara Institute of Science and Technology, in 2002. He was an Assistant Professor with the Nara Institute of Science and Technology, in 2002, and a Senior Lecturer with Kyoto University, in 2008. Since 2020, he has been a Senior Researcher with the HR Science Institute, Miidas Corporation. His research interests include bio-statistics and machine learning.