A Survey on Deep Learning for Skeleton-Based Human Animation

Lucas Mourot, Ludovic Hoyet, François Le Clerc, François Schnitzler, Pierre Hellier

Published in COMPUTER GRAPHICS *Forum* (CGF) https://doi.org/10.1111/cgf.14426

A Survey on Deep Learning for Skeleton-Based Human Animation

L. Mourot[1,2], L. Hoyet[1], F. Le Clerc[2], F. Schnitzler[2] and P. Hellier[2]

[1]Inria, Univ Rennes, CNRS, IRISA
[2]InterDigital, Inc

Abstract

Human character animation is often critical in entertainment content production, including video games, virtual reality or fiction films. To this end, deep neural networks drive most recent advances through deep learning and deep reinforcement learning. In this article, we propose a comprehensive survey on the state-of-the-art approaches based on either deep learning or deep reinforcement learning in skeleton-based human character animation. First, we introduce motion data representations, most common human motion datasets and how basic deep models can be enhanced to foster learning of spatial and temporal patterns in motion data. Second, we cover state-of-the-art approaches divided into three large families of applications in human animation pipelines: motion synthesis, character control and motion editing. Finally, we discuss the limitations of the current state-of-the-art methods based on deep learning and/or deep reinforcement learning in skeletal human character animation and possible directions of future research to alleviate current limitations and meet animators' needs.

CCS Concepts

• ***General and reference*** → ***Surveys and overviews;*** • ***Applied computing*** → ***Media arts;*** • ***Computing methodologies*** → ***Motion processing; Physical simulation; Neural networks;*** *Supervised learning; Unsupervised learning; Reinforcement learning;*

1. Introduction

Humans and their representations are ubiquitous in culture. In the past decades digital technologies brought more and more tools to assist designers, animators and artists, with the goal of increasing their creative capabilities and the realism of their artworks while reducing production costs. For instance, digital doubles have brought to life non-human fictional creatures like the Na'vi in *Avatar* or even visual identity of long time deceased actors like Grand Moff Tarkin in *Rogue One* in 2016 portrayed by the British actor Peter Cushing, even though he passed away in 1994. Within this context, computer-assisted human character animation plays a central role, such as for the synthesis of the motion of digital doubles. Another example is the video game production, where animation and control of the characters directly condition the success of a game.

However, animation of human characters is challenging: the way humans move is very diverse and is influenced by many factors including the mood, the intentions, the activity or even individual characteristics. In addition, Newton's second law of motion inherently makes human motion a dynamic process, while our understanding of biomechanics is far from being comprehensive. In practice, compromises therefore need to be made to balance between production costs, the realism, the amount of manual work, and the level of expertise of the animator. The research area of character animation has been active for decades to mitigate these compromises and make animation more accessible, starting from pioneering works such as editing and deforming motion examples [WP95], retargeting motions to new characters [Gle98], controlling characters using motion graphs [KGP02; MC12], etc.

Over the past few years, *Deep Neural Networks* (DNNs) have emerged as a powerful means to enhance the performance and capabilities of character animation, as evidenced by the growing number of publications on the topic leveraging *Deep Learning* (DL) and *Deep Reinforcement Learning* (DRL) (see Figure 1). These two techniques have shown an unprecedented ability to address complex tasks in a wide variety of domains not restricted to animation, such as computer vision, *Natural Language Processing* (NLP) and many more. Humans are outperformed by artificial intelligence algorithms in a growing number of tasks such as image classification or playing Go. This is notably due to the fact that DNNs are powerful function approximators, able to learn sophisticated patterns in complex real-world phenomena from data. Moreover, once the training is complete, training data is discarded, leaving compact models that are able to meet performance requirements of real-time applications or to scale to embedded systems. In animation, DL & DRL based approaches attempt to handle the human motion complexity and provide promising perspectives for cheaper and faster animation techniques with more and more fidelity and creative capabilities.

In this article, we present an overview of the recent growing trend of DL & DRL in skeletal character animation, focused on humanoid characters. *Skeletal* here means using a representation derived from a skeleton, as commonly used in the movie and game in-

This article has been published in COMPUTER GRAPHICS *Forum* (CGF). https://doi.org/10.1111/cgf.14426

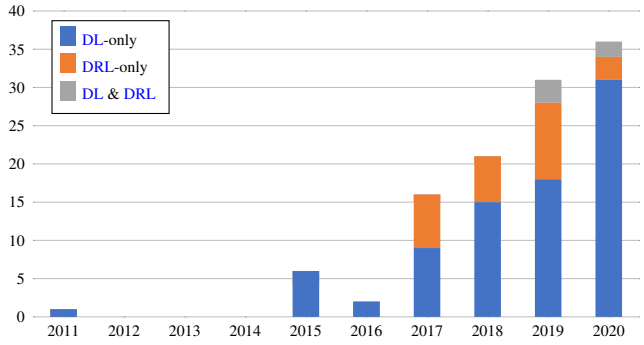

Figure 1: *Histogram of the peer-reviewed publications over the past decade in human skeletal character animation using* Deep Learning *(DL) and/or* Deep Reinforcement Learning *(DRL) addressed in this survey.*

dustries in combination with 3D skinned meshes (see Section 2.1). On the one hand, DL is particularly effective at building compact models from such human motion data, e.g. for motion synthesis and editing. On the other hand, DRL, which is concerned with how agents ought to act in a simulated environment in order to maximise some cumulative reward, is well-suited for character control where characters are agents, and their actuation model parameters are actions. The goal of this review is therefore to provide researchers involved in character animation and related fields with an overview of existing DL & DRL based methods in that domain, best practices to represent and process motion data with DNNs and the most successful approaches for different well-studied families of problems.

In the last decade, various state-of-the-art reports have studied character animation topics: Pejsa and Pandzic [PP10] covered motion planning, motion graphs and parametric models for motion synthesis in interactive applications from examples; Geijtenbeek and Pronost [GP12] reviewed the literature on physical simulation for interactive character animation; then Karg et al. [KSG*13] studied the generation and recognition of motions based on affective expressions; finally, Wang et al. [WCW14] presented an analysis of state-of-the-art techniques in 3D human motion synthesis while focusing on motion capture data-driven methods. However, none of these reviews covered DL or DRL based methods because most recent advances appeared since 2015, as shown in Figure 1. More recently, Alemi and Pasquier [AP19] reviewed the topic of data-driven movement generation with *Machine Learning* (ML). While their paper includes the review of some DL-based methods, numerous other advances have emerged in very recent years. In addition, some of the ML approaches presented by Alemi and Pasquier [AP19], such as Hidden Markov Models or Principal Component Analysis, are clearly outperformed by DL as of today. In contrast, our work addresses state-of-the-art of DL and DRL in skeletal human character animation.

This paper is organised as follows: in Section 2, we begin with an overview of low-level concerns encountered when processing human motion data with DNNs, presenting pose representations (Section 2.1) and human motion datasets (Section 2.2) frequently used in the literature, as well as how to efficiently and successfully learn spatial (Section 2.3) and temporal (Section 2.4) features.

Next, Section 3 covers motion synthesis, that we define as the process of creating perceptually plausible motion sequences with a desired style or expressed emotion for instance. Motion synthesis models are thus capable of generating different motions depending on inputs, including low-level parameters (e.g., latent variables), high-level parameters (e.g. trajectory or specific action to perform) or historical parameters (e.g. past motions to be extrapolated). In this review, we divide motion synthesis approaches into three categories: predictive in the short-term (Section 3.1), predictive in the long-term (Section 3.2) and generative (Section 3.3). The first focuses on deterministically synthesising motion segments from few past frames while the second tries to generate longer plausible motion continuations. The third category covers motion synthesis from other types of parameters, such as the trajectory to follow, and relies on generative models.

Section 4 deals with the task of controlling the motion of characters so that they react naturally to user inputs while accounting for environment and biomechanical constraints. We divide these approaches into kinematic (Section 4.1), physical (Section 4.2) and biomechanical (Section 4.3) control. Kinematic approaches directly produce motions as joint positions or angles. In contrast, both physical and biomechanical approaches strive to obey the laws of physics, while differing in the actuation model: physical models are actuated by forces and torques, while biomechanical models (a.k.a. musculoskeletal models) are driven by muscle activations.

Finally, Section 5 gathers motion editing approaches at large, i.e. methods aiming to process or transform some aspects of existing motion data. Motion cleaning (Section 5.1) improves motion data, e.g. by removing noise or filling in missing information such as marker or joint positions. Note that we distinguish here markers and joints completion from motion prediction and in-betweening that are addressed in Sections 3.1 and 3.2 respectively, although all three might be formulated as completion from partial observations. Then, retargeting (Section 5.2) strives to transfer the motion from a source character to a target character, while motion style transfer (Section 5.3) edits the style of a motion segment while preserving the action performed and the character.

2. Human Motion Representation, Data and Modelling

In both DL and DRL, choices of input and output spaces for DNNs are often impactful on the effectiveness of the learning and on what specific aspects of the data will be retained. When dealing with human motion, the pose representation mainly determines these input and output spaces. Moreover in DNNs, the computational workflow can be structured around spatial and temporal aspects of motion. In this section we explore the different human pose representations commonly encountered in deep animation and their key strengths and weaknesses (Section 2.1), commonly used datasets (Section 2.2), as well as DNN architectures structured with respect to spatial (Section 2.3) and temporal (Section 2.4) domains.

This article has been published in COMPUTER GRAPHICS *Forum* (CGF). https://doi.org/10.1111/cgf.14426

2.1. Pose Representation

Traditional animation approaches typically use 3D rigged skeletons with skinned meshes, which provides a good trade-off between quality and complexity. Rigging offers convenient ways to manipulate 3D models as strings do for a puppet, while skinning is the process of binding actual 3D meshes to animated characters. In that framework, human motions are usually represented as sequences of poses separated by constant time intervals whose rate generally ranges from 30 to 250 hertz. At each time step, the state of the human body is then represented as a set of links (i.e. bones) connected by joints. This skeletal representation is a good compromise for the complexity and the diversity of human movements that can be represented. When bone lengths are kept constant over time, the *Degrees of Freedom* (DOFs) are the orientations of the bones, commonly expressed relative to their parent. In the following, we call such a representation an *angular pose representation*, as opposed to a *positional pose representation* where the skeleton DOFs are the coordinates of the joint positions, which does not explicitly constrain bone lengths to remain constant over time.

2.1.1. Positional Pose Representations

In a positional pose representation, each joint is directly represented by its position, generally expressed at each time step in the body's local coordinate system [HSKJ15; HSK16; WHSZ19; HAB20; DHS*19; HHS*17; ZLX*18; HHKK17; SCNW19; TCHG17; KAS*20], which allows the decomposition of the whole motion into the local movements of limbs with respect to the body itself and the global movement of the body with respect to its environment. Although different coordinate systems could be formulated to embed the set of joint positions, positional pose representations almost always rely on the Cartesian coordinate system. It has neither discontinuities nor singularities and constitutes a convenient space for interpolation, visualisation and optimisation. Moreover, within the framework described here, positional pose representation does not present some of the limitations inherent to angular representations presented in the following section. However, it suffers from some limitations related to the structure of human motions. For instance, joint positions do not encode the information of bone orientations around themselves which is often needed for concrete applications in animation, in order to display more natural mesh deformations. Positional representations also do not explicitly constrain bone lengths to remain constant over time, therefore requiring reliance on additional constraints to ensure that the skeleton does not break apart. For these reasons, communities closer to animation, such as computer graphics, rarely use purely positional pose representations, while on the opposite, communities more commonly involved in deep learning, such as computer vision, are more prone to employing these representations.

2.1.2. Angular Pose Representations

Angular representations have been widely used in animation mainly because their hierarchical nature allows straightforward orientation of any joint together with all of its descendants, while keeping bone lengths constant. Indeed, the position of each joint is described with respect to its parent as a 3D rigid transformation, often decomposed into a variable rotation and a fixed translation, corresponding to the joint orientation and the bone dimensions respectively. Main differences among angular pose representations are determined by the parameterisation of the rotations, however there are also representations working at the level of rigid transformation parameterisations.

Formally, the set of all rotations of $\mathbb{R}^3$ equipped with the composition is the 3D rotation group often denoted $\mathrm{SO}(3)$, standing for special orthogonal group of dimension 3. $\mathrm{SO}(3)$ can be identified with the group of orthogonal 3×3 matrices with determinant 1 under the matrix multiplication. Similarly, the 3D special Euclidean group whose elements are proper 3D rigid transformations (i.e. excluding reflections) is $\mathrm{SE}(3) = \mathrm{SO}(3) \times \mathbb{R}^3$. Both $\mathrm{SO}(3)$ and $\mathrm{SE}(3)$ are Lie groups, i.e. differentiable spaces that locally resemble Euclidean space. Furthermore, a Lie algebra is associated to every Lie group, called $\mathfrak{so}(3)$ and $\mathfrak{se}(3)$ for $\mathrm{SO}(3)$ and $\mathrm{SE}(3)$, respectively. Lie algebras are vector spaces tangent to their Lie group at the identity element completely capturing its local structure, making them compelling as representation spaces.

Euler Angles. The most intuitive parameterisation of $\mathrm{SO}(3)$ is probably *Euler angles*, that represents a 3D orientation as three successive rotations around different axes, e.g. yaw, pitch and roll. However, it suffers from the well-known gimbal lock when two of the three rotation axes align, causing a DOF to be lost. Gimbal lock can be avoided only if at least one rotation axis is limited to a range smaller than 180°, which is not always possible in practice. As a result, Euler angles are unsuitable for *Inverse Kinematics* (IK), dynamics and spacetime optimisation [Gra98]. Moreover, they do not work well for interpolations since the space of orientations is highly nonlinear [Gra98]. Finally, multiple conventions exist for the axes considered, including their order, which requires to define them formally in each application to avoid any ambiguity. For these reasons, this representation is inappropriate for a lot of applications.

Lie Algebras. A popular angular pose representation in skeletal character animation consists in representing each joint rotation as an element of $\mathfrak{so}(3)$, where the direction and magnitude of the vector correspond to the axis and angle of the rotation [Gra98], respectively. Since such a vector is an element of $\mathfrak{so}(3)$, the parameterisation is defined by the exponential map from $\mathfrak{so}(3)$ to $\mathrm{SO}(3)$ which can be efficiently computed with the Rodrigues' formula [Rod40]. This pose representation is often called *exponential map*, and is sometimes confused with the so-called *axis-angle* representation that is equivalent but separates the vector into a unit vector and a scalar describing the axis and the magnitude of the rotation.

Since Lie algebras are locally linearised versions of their Lie group, $\mathfrak{so}(3)$ is a compelling space to work with elements of $\mathrm{SO}(3)$. However, as all parameterisations of $\mathrm{SO}(3)$ in $\mathbb{R}^3$, the exponential map representation has singularities [Gra98] leading to losing a DOF in some parts of the representation space, even though these are located on the spheres of radius $2k\pi$ for $k \in \mathbb{N}^+$, since a rotation of 2π about any axis is equivalent to no rotation. Therefore, this representation is often well-suited in animation since control and simulation deal with small time steps and thus with small rotations that stay inside the sphere of radius 2π, far from the singularities. It has been employed in early works in deep animation [THR06; TH09], and broadly exploited for motion synthe-

This article has been published in COMPUTER GRAPHICS *Forum* (CGF). https://doi.org/10.1111/cgf.14426

sis [CM11; ALP15] and prediction [FLFM15; JZSS16; BBKK17; MBR17; LZLL18; TMLZ18; GWLM18; GMK*19; LWY*19; GC19; KGB19; WAC*19; CSY20; MLS20; LCZ*20; CSK*21; LCC*21], as well as in other topics [HKS17; AP17; MSZ*18; JL20]. However, as quality needs increase, long-term correlations are more and more important and inevitably imply larger rotations, getting close to the singularities of the parameterisation.

Similarly to the exponential map representation which uses $\mathfrak{so}(3)$ to represent joint orientations w.r.t. their parents, Liu et al. [LWJ*19] proposed a pose representation using $\mathfrak{se}(3)$ to represent rigid transformations of each joint w.r.t. its parent. The main motivation for choosing such a representation is to explicitly encode both geometric constraints (i.e. bone lengths) and actual DOFs (i.e. joint orientations) together. Nevertheless, it still has the same singularities as the exponential map representation. As we will see in the following paragraphs, other parameterisations in higher-dimensional spaces than $\mathbb{R}^3$, i.e. over-parameterised representations, are able to prevent such singularities.

Rotation Matrices. In computer graphics, rotation matrices are widely used to represent 3D rotations. The corresponding parameterisation is the identity since elements of $\mathrm{SO}(3)$ are 3×3 matrices. Rotation matrices have no singularity and can be integrated together with joint translations into a 4×4 homogeneous matrix, which is elegant and effective when involved in computations like composition or inverse. However, such a representation is particularly difficult to work with when its parameters must be estimated since the representation is over-parameterised. Indeed, not all 3×3 matrices belong to $\mathrm{SO}(3)$. By definition, a matrix $R \in \mathbb{R}^{3\times3}$ must satisfy $R^\top R = I$ and $det(R) = 1$ to be a valid 3D rotation. For instance, predicting the orientation of a joint in matrix representation would require to solve a constrained optimisation problem to ensure the validity of the rotation, which can be tedious.

Unit Quaternions. A more compact representation than rotation matrices are unit quaternions. Lying in $\mathbb{R}^4$, they are free of singularities, suitable for interpolation [Gra98], numerically stable and computationally efficient [PGA18]. Like $\mathfrak{so}(3)$, the space of unit quaternions has the same local geometry and topology as $\mathrm{SO}(3)$ [Gra98]. However, unit quaternions are also over-parameterised, but have only four parameters (in comparison to nine parameters for rotation matrices). Thus, they must be constrained to remain on the unit 4-sphere. This angular representation has been popularised in DL-based animation with *QuaterNet*, a quaternion-based framework for human motion prediction proposed by Pavllo et al. [PGA18; PFAG20] (see Section 3.1). In that framework, Pavllo et al. introduced a penalty term in the loss function for all quaternions predicted by the network that minimises their divergence from the unit length. It encourages the network to predict valid rotations and leads to better training stability. Moreover, the predicted quaternions are also normalised after computing the penalty to enforce their validity. According to the authors, the distribution of predicted quaternion norms converges to a Gaussian with mean 1 during the training, suggesting that the model actually learns to represent valid rotations. Since Pavllo et al. [PGA18] showed promising results using quaternions, their use is gaining popularity [KPKH20; AWL*20; ASS*20; ALL*20; PFAG20; LCC19; VYCL18; HYNP20; GWE*20].

Gram-Schmidt-like. Zhou et al. [ZBL*19] recently pointed out that all representations in $\mathbb{R}^n$ with $n \leq 4$ have discontinuities which can be unfavorable for DNNs training. They therefore introduced a continuous representation of n-dimensional rotations $\mathrm{SO}(n)$ with $n^2 - n$ dimensions. The mapping from $\mathrm{SO}(n)$ to the representation space simply drops the last column vector of the input $n \times n$ matrix. The inverse mapping back to $\mathrm{SO}(n)$, called Gram-Schmidt-like process, is a Gram-Schmidt process over the $n-1$ column vectors followed by the computation of the last column vector by a generalisation to n dimensions of the cross product. In the case of $\mathrm{SO}(3)$, this *Gram-Schmidt-like* representation gives a 6D representation. Zhou et al. [ZBL*19] also provided a method to further reduce the dimensionality from 6D to 5D while still keeping a continuous representation using a stereographic projection combined with normalisation. However, they empirically found that nonlinearities introduced by the projection can make the learning process more difficult. To the best of our knowledge, very few works have investigated this promising representation of 3D rotations.

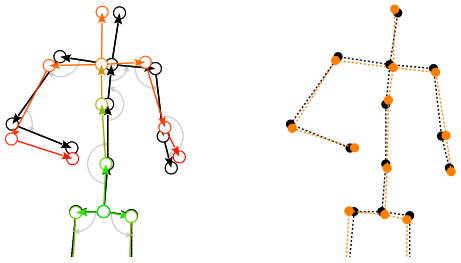

Figure 2: *Illustration of the error accumulation problem with angular representations: even small angular errors along the kinematic chain can lead to large accumulated joint positioning errors (left-hand stick figure, see color gradient). This is problematic in optimization-based methods, e.g. DL/DRL, when penalizing joint orientation deviations. This is not the case with positional representations (right-hand stick figure) where joint positions are directly optimized.*

Hierarchical Representation Limitations. In skeletal character animation, a hierarchical modelling approach is used in conjunction with angular pose representations, i.e. the representation of the joint orientations relative to their parents. In that context, positional errors over proximal joints (e.g. the shoulder) are propagated and accumulated down the kinematic chains. This is problematic in optimisation-based methods such as DL or DRL since equally distributed joint orientation errors will result in growing joint position errors along the kinematic chains as depicted in Figure 2, making it difficult to accurately handle end effectors. This is especially true in motions sequences involving fast or ample movements, e.g. running.

To solve this problem, Pavllo et al. [PGA18; PFAG20] performed *Forward Kinematics* (FK) to convert quaternion-based poses predicted by their DNN into 3D joint positions, and then penalised absolute position errors instead of angular errors. Since FK is a differentiable operation with respect to joint orientations, they can train their network end-to-end using a positional loss.

This article has been published in COMPUTER GRAPHICS *Forum* (CGF). https://doi.org/10.1111/cgf.14426

While recent works have followed the same approach [ASS*20; ALL*20], Ghorbani et al. [GWE*20] recently pointed out that applying FK is computationally expensive especially for motion sequences that are long or involve numerous joints. To this end, they proposed to hierarchically weight joint angle errors based on their impact on the positions. They set the weight of a joint as its maximum path length down to all of the connected end effectors in an average skeleton. While ablative studies showed improved performances when joint weighting was enabled, the results were not compared to Pavllo et al.'s FK-based approach [PGA18; PFAG20], neither were the choice of the weights assessed.

2.1.3. Hybrid Representations

As mentioned above, both angular and positional approaches for representing human poses have advantages and drawbacks that sometimes depend on the application or viewpoint, dividing researcher communities. For this reason, several works have proposed hybrid representations with the goal of mitigating drawbacks while keeping benefits of both types of representations.

Aberman et al. [ALL*20] presented a novel data-driven approach for retargeting motions between homeomorphic skeletons (see Section 5.2) along with an interesting and elegant representation of human motion illustrated in Figure 3. In this work, both angular and positional information are combined: a static component S consisting of a set of 3D positional offsets describes the skeleton in some arbitrary pose (similar to a T-pose but specific to a pose sequence), while a dynamic component Q specifies the sequence of orientations of each joint along time (with respect to S) represented using unit quaternions. The separation between static and dynamic partial representations enables the authors to design an architecture such that each component is processed in a separate branch. In continuation of Pavllo et al.'s work [PGA18; PFAG20], Aberman et al. [ALL*20] penalised errors in the positional space after performing FK, while also penalising errors in the quaternion space. Following this work, Shi et al. [SAA*20] addressed the reconstruction of 3D kinematic skeletons from 2D keypoints estimated from monocular video while dividing the motion representation into static bone lengths and dynamic joint orientations. To this end, a DNN called *MotioNet* learns to map 2D keypoints to a symmetric static skeleton, represented by its bone lengths and a dynamic sequence of joint rotations (quaternions) which are then combined through FK to get a full kinematic skeleton.

Finally, the success of multiple recent methods in skeletal character animation mixing different pose representations suggests that DNNs benefit from redundant pose information. In addition to joint orientations, researchers fed their models with joint positions [LLL18], joint positions and velocities [HKS17; MSZ*18; ZSKS18; LZCvdP20; SZKZ20; SZKS19] or even joint positions and linear and angular velocities [HKPP20].

2.2. Human Motion Datasets

Another crucial aspect of data-driven approaches is the choice of dataset, from which a DL-based model will learn a deep representation of human motion. In particular, large amounts of high-quality motion data are necessary to constitute so-called benchmark

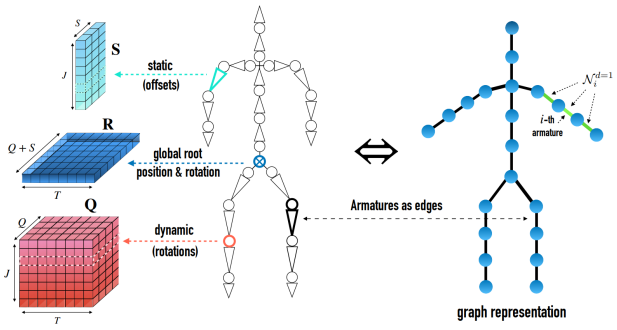

Figure 3: *Representation of skeletal motion data as a graph in [ALL*20]. The nodes of the graph correspond to joints and the edges to armatures. Each of the J armatures holds a time-varying tensor Q modelling the temporal sequence of rotations at its corresponding joint, and a time-independent vector S modelling the bone offset to the parent joint. The global motion of the root joint R is processed separately. Figure from [ALL*20].*

datasets and to provide robust assessment procedures. In this section we introduce a selection of human motion datasets that are the most relevant for this survey. Although many datasets have been proposed, only a small number of them have been repeatedly exploited and even fewer can be considered as standard benchmarks (see Figure 4). Table 1 provides relevant information about these datasets most commonly used in the works presented in this survey.

The two most widely used databases in skeleton-based deep human animation are CMU [Uni03] and Human3.6M [IPOS14]. Both are standard large-scale human motion datasets for learning and evaluation. Despite the fact that the CMU dataset was released about a decade earlier than Human3.6M, they have a comparable size (see Table 1). The main advantage of Human3.6M over CMU is the presence of RGB+D videos synchronized with human pose sequences, making it sometimes more suitable for tasks closer to computer vision such as motion prediction, even though CMU is also often leveraged. To obtain convincing results, several researchers provide evaluations of their work over both CMU and Human3.6M data [BBKK17; LZLL18; LWY*19; GC19; KGB19; MLSL19; CSY20; LCZ*20; ZPK20; CHW*20; LKS*20; LCC*21; CSK*21; BGG*20; ASS*20].

Beyond these two standard human motion databases, a few others are noticeable, e.g. for the types of motion they contain, for the annotations that are included, or even for the environment in which they were captured. A few years after the release of CMU, Müller et al. [MRC*07] proposed HDM05, a public well-documented database of systematically recorded motion capture data. Complementary to CMU which contains a large number of diverse motion sequences, HDM05 is composed of a limited number of specific motion sequences (one hundred) which were executed from 10 to 50 times by five actors. As an example, a cartwheel starting with the left hand has been performed 21 times. More recently, Shahroudy et al. [SLNW16] proposed NTU RGB+D, one

This article has been published in COMPUTER GRAPHICS *Forum* (CGF). https://doi.org/10.1111/cgf.14426

Table 1: *Summary of the main datasets presented in Section 2.2.*

Dataset	URL	Size	Data			Availability
			Joints	Representation	Miscalleneous	
CMU [Uni03]	</>	$3.9 * 10^6$ frames @ 120 Hz → 9.1 h	29	angular		Public
HDM05 [MRC*07]	</>	$3.6 * 10^5$ frames @ 120 Hz → 0.8 h	31	angular		Public
Human3.6M [IPOS14]	</>	$3.6 * 10^6$ frames @ 50 Hz → 20.0 h	32	angular	RGB+D	On request
Holden et al. [HSK16]	</>	$6.0 * 10^6$ frames @ 120 Hz → 13.9 h	21	positional		Public
NTU RGB+D [SLNW16]	</>	$4.0 * 10^6$ frames @ 30 Hz → 37.0 h	25	positional	RGB+D+IR	On request
NTU RGB+D 120 [LSP*20]	</>	$8.0 * 10^6$ frames @ 30 Hz → 74.1 h	25	positional	RGB+D+IR	On request
3DPW [vMHB*18]	</>	$5.1 * 10^4$ frames @ 30 Hz → 0.5 h	23	angular	RGB	Public
AMASS [MGT*19]	</>	$1.8 * 10^7$ frames @ [60, 250] Hz → 41.5 h	52	angular	Body mesh	Public
Mixamo [Ado]	</>	$2.7 * 10^5$ frames @ 30 Hz → 2.5 h	52	angular	Body mesh	Public

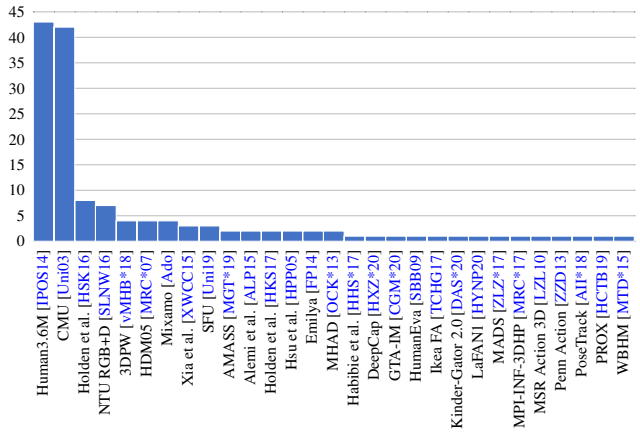

Figure 4: *Histogram of the number of papers using a given dataset among the works in skeleton-based deep human animation covered in this survey.*

of the largest datasets for 3D skeleton-based action recognition. It contains 60 action classes as well as RGB and infrared (IR) videos and depth map sequences (D) synchronized with motion sequences. Later on, Liu et al. [LSP*20] added another 60 classes to constitute NTU RGB+D 120, roughly doubling the size of the dataset. However, the motion sequences in these two datasets are represented only by joint positions which limits their use in skeleton-based human animation. Since motion capture systems need dedicated environment e.g. for a multicamera setup, human motion datasets are mostly captured in the lab, resulting in a lack of in the wild data. To this end, von Marcard et al. [vMHB*18] proposed a pose estimation method, leveraging inertial measurement units in addition to a hand-held camera, accurate enough to capture a new dataset called 3DPW consisting of human pose sequences in the wild synchronized with RGB videos. It contains challenging sequences including walking in the city, going up-stairs, or taking the bus for a total of more than 51000 frames.

A special need of data occurs from the use-case of retargeting (see Section 5.2), which consists in transferring the movements of a character to another one with a different morphology, i.e. motion data with diversity among morphologies. Unfortunately, most human motion datasets feature only a very limited number of subjects with minor morphological differences. As a result the so-called Mixamo Dataset [Ado] is often used to train or evaluate models for retargeting. Indeed Mixamo is a computer graphics technology company developing services for 3D character animation including an online animation store with downloadable animation sequences performed by numerous 3D character models with varied morphologies. This diversity of character models is crucial to the task of retargeting.

Finally, existing datasets were also gathered to build larger human motion databases. Holden et al. [HSK16] constructed a dataset by collecting CMU [Uni03], HDM05 [MRC*07], MHAD [OCK*13] and Xia et al. [XWCC15] in addition to internal motion capture sequences. This data was retargeted to a uniform skeleton structure and resampled to 120 frames per second. More recently, Mahmood et al. [MGT*19] proposed AMASS, which unifies 15 different optical marker-based motion capture datasets (including CMU [Uni03], HDM05 [MRC*07], SFU [Uni19], HumanEva [SBB09]). The size of AMASS, initially around 42 hours of data, is still increasing. Motion sequences in AMASS are parameterised using the Skinned Multi-Person Linear (SMPL) model [LMR*15], a learnt model of human body shape and pose that provides a parameter space from which the skeleton, the joint orientations and the body mesh can be computed.

2.3. Learning Spatial Features

Besides the pose representation and the dataset, the network architecture can also have a significant impact on the deep representation learned. In this section, we present different types of architectures leveraged to learn spatial correlations in motion data. While most DNNs designed to address tasks related to skeletal character animation do not exploit the prior knowledge we have about geometric and structural aspects of the human skeleton, e.g. its symmetry or its hierarchical structure, a few methods proposed various clever architectures to benefit from this prior knowledge. We present them in the following sub-sections, divided into three categories: spatially-structured architectures, *Convolutional Neural Networks* (CNNs) and *Graph Convolutional Networks* (GCNs).

2.3.1. Spatially-Structured Architectures

A first group of approaches to help DNNs learn spatial correlations rely on network architectures structured around the human skeleton, such that the function computed by the network intrinsically encodes human skeleton characteristics. These approaches split the skeleton into body parts and process the corresponding data either in parallel network branches or hierarchically.

This article has been published in COMPUTER GRAPHICS *Forum* (CGF). https://doi.org/10.1111/cgf.14426

In parallel approaches, the main difference is usually related to the targeted task, which conditions the architecture of individual branches. Wang and Neff [WN15] extracted deep motion signatures with an independent autoencoder for each branch (i.e. limb or torso) and concatenated the outputs. Guo and Choi [GC19] relied instead on fully connected layers for each branch, whose outputs are merged using a shared layer to predict the next frame. Nakada et al. [NZC*18] similarly divided the body into separate modules responsible for controlling muscle activations of body parts from preprocessed common visual information. Finally, Jain et al. [JZSS16] predicted future poses with one *Recurrent Neural Network* (RNN) per body part, whose inputs are the neighbouring RNNs predictions at the previous timestamps as well as their own previous predictions.

Hierarchical approaches can be either top-down or bottom-up. Wang et al. [WHSZ19] proposed a spatial encoder that separates the human pose into five body parts, then encodes and merges them two by two recursively. Both Li et al. [LWY*19] and Bütepage et al. [BBKK17] used a similar top-down approach with a finer human pose split at the input. In contrast, Aksan et al. [AKH19] proposed a bottom-up scheme where the human pose is predicted step by step from the root joint (e.g. pelvis) to the end effectors, i.e. the root is first predicted and then the other joints are recursively predicted using neighbouring predictions as additional inputs.

2.3.2. Convolutional Neural Networks

Another type of architecture sometimes employed to model spatial correlations relies on 2D convolutions, i.e. in the spatial and temporal domains. To this purpose, the skeleton graph is flattened along the spatial dimension. CNNs are particularly efficient at learning spatial correlations in data whose structure is regular such as images. However, learning the spatio-temporal dynamics of human joints remains challenging with CNNs because the graph structure of the human skeleton cannot be meaningfully flattened along a single dimension. To capture the spatial correlations of joints from different limbs, Li et al. [LZLL18] proposed to enlarge the convolutional kernels in the spatial domain. More recently, Zang et al. [ZPK20] proposed to adaptively model the spatial correlations with deformable convolutional kernels whose relative positions of the entries are learned. The problem of learning patterns in irregular data structures like human poses with CNNs can be overcome by extending convolutions to graph-structured data, as we will see next.

2.3.3. Graph Convolutional Networks

To leverage CNNs while properly handling the graph-structure typically used in animation to represent skeletons, *Graph Convolutional Networks* (GCNs), an extension of CNNs, have been recently considered in different frameworks working on human motion data. GCNs come in two different flavours [BBL*17]. Spatial approaches map neighbourhoods of each node in the graph to Euclidean patches on which a convolution is applied. Spectral approaches operate in the Fourier domain of the feature signals sampled on the graph, which depends on the graph Laplacian operator [SNF*13]. By analogy with the convolution theorem, convolution filters are defined as spectral coefficients that are multiplied by the Fourier transforms of the signals.

Aberman et al. [ALL*20] resorted to a simple implementation of spatial GCNs. The supports of convolution kernels around each joint are defined as d-ring neighbourhoods on the skeleton graph in the spatial dimension and extended to the temporal axis to obtain 2D skeleto-temporal convolutions. The operation of such GCNs is limited to skeletons sharing the same topology. However, the motion retargeting network they proposed includes skeletal pooling/unpooling layers, based on the fusion/duplication of the signals of adjacent edges. The pooling layers bring the input topology to a common primal skeleton on which the core processing is performed. The result is transformed back to the original topology by the unpooling layers. Such a network can cope with any topology that is homeomorphic to the primal skeleton.

The other contributions leveraging GCNs [MLSL19; CSY20; LCZ*20; LCC*21] relied on the spectral approach of Kipf and Welling [KW17]. Here, the output F^{l+1} of the graph convolutional layer l fed with a feature signal F^l is formulated as $F^{l+1} = \sigma(\hat{A}F^lW^l)$ where W^l is the tensor of learnable convolution filter weights, $\hat{A}$ depends only on the graph adjacency matrix and σ is a non-linear activation function. In all of the proposed approaches the weights of the adjacency matrix are learnt in addition to the convolution filter. Mao et al. [MLSL19] built their adjacency matrix from a fully connected graph of joints and thereby simultaneously learnt the motion correlations between joints that are physically connected and joints that are far apart but whose motion are dependent, e.g. hands and feet during walking. Cui et al. [CSY20] objected that this scheme may result in unstable training and separately learnt two graph adjacency matrices, one in which the weights of non-connected joints in the skeleton are forced to 0 and another with full joint connectivity. Two papers by Li et al. [LCZ*20; LCC*21] proposed GCNs that operate at multiple scales, the finest scale corresponding to individual joints and the coarser scales to increasingly large body parts. While scales are analysed in parallel branches in [LCC*21], in [LCZ*20] the features at various scales are additionally fused within each graph convolutional block.

2.4. Learning Temporal Features

The temporal dimension of motion data is informative of the nature of the action being performed as well as the way it is performed. In the following sub-sections we review the approaches taken to model human dynamics, most of which rely either on RNNs or CNNs.

2.4.1. Recurrent Neural Networks

RNNs are neural networks designed to process each timestep of a time series one after another, and can thereby handle variable length sequences. They maintain an internal state that captures the temporal context of the signal. RNNs are most of the time based on *Long Short-Term Memory* (LSTM) or *Gated Recurrent Unit* (GRU).

Long Short-Term Memory. A common LSTM is composed of a memory cell that remembers values over arbitrary time intervals, and three gates – an input gate, an output gate and a forget gate – to

This article has been published in COMPUTER GRAPHICS *Forum* (CGF). https://doi.org/10.1111/cgf.14426

regulate the flow of information into and out of the cell and to avoid a common problem with RNNs known as the vanishing (or exploding) gradient problem. In general, the problem is that the gradients used to update the network weights can become extremely small (or large), either preventing the network from further learning or making the network diverge, respectively. In the case of RNNs, the backpropagation through time heavily relies on the chain rule to compute gradients which exponentially decrease (vanishing problem) or increase (exploding problem) if any weight is greater or smaller than 1, respectively.

The memory cell remembers values over arbitrary time intervals, making LSTM effective at capturing both short-term and long-term temporal dependencies. Indeed, LSTMs have proven to be powerful for learning temporal dependencies by achieving state-of-the-art performance in key applications e.g. NLP, machine translation, etc. In human motion related problems, LSTMs have been broadly employed, e.g. in motion prediction [FLFM15; CAW*19; LWJ*19; KGB19; JZSS16] and generation [HAB20; HYNP20; GSAH17; WYZ*20; WHSZ19; WACD20], in both physical [WMR*17; HTS*17; MAP*19; MTA*20] and kinematic [WCX17; LLL18; WCX21] character control, as well as in motion cleaning [ZLX*18] and style transfer [WCAD18].

Gated Recurrent Unit. As an alternative to LSTMs, GRUs have also been widely used, such as in motion prediction [MBR17; TCHG17; GWLM18; GWRM18; PGA18; GMK*19; WAC*19; XLM19; GC19; CPAM20; AAR*20; LCZ*20], in motion generation [YK20; BKL18; ASS*20; GWE*20] or in motion editing [VYCL18; JL20]. GRUs rely on a gating mechanism similar to LSTMs in order to avoid the vanishing gradient problem but have only two gates, a reset gate and an update gate. As a result, GRUs use fewer parameters and therefore less memory, are computationally less expensive [GC19] and thus train faster than LSTMs. They can process entire motion datasets [MBR17] instead of training action-specific models [FLFM15; JZSS16]. However, as shown by Weiss et al. [WGY18], the LSTM is strictly stronger than the GRU as it can easily perform unbounded counting, while the GRU cannot. Thus, LSTMs seem more accurate than GRUs on longer sequences. In summary, the choice between LSTM and GRU depends on the processed data and the considered application.

Besides pure LSTM or GRU, extensions [TMLZ18] or combinations of both [WAC*19] have been used to model human dynamics. *Bidirectional LSTMs* (BiLSTMs) stack two LSTMs running forward and backward, respectively. As a result, temporal information is processed and preserved in both directions, i.e. past and future, which is helpful on certain tasks. For instance, BiLSTMs are leveraged to map an example motion into an embedding within an imitation learning framework [WMR*17], to build a long plausible motion sequence from a set of short motion clips [XXN*20] or even to refine 3D motion data [LZZ*19; LZZL20].

2.4.2. Temporal Convolutions

CNNs provide an alternative to RNNs for learning temporal patterns in motion data. They can be either 1D along the temporal dimension, or use 2D spatio-temporal convolutions. Stacking several convolutional layers can efficiently capture both short and long range temporal patterns since lower and higher layers will capture dependencies between nearby and distant frames, respectively. CNN-based approaches are more computationally efficient than RNN-based ones because they process whole motion segments at once rather than frame by frame, allowing greater parallelism. However, CNN-based architectures often contain elements that do not allow variable length inputs (e.g. a few fully connected layers after convolutions) limiting their use in practice. As a result, they are quite rare in motion synthesis [HSK16; HGM19; DHS*19], prediction [BBKK17; PFAG20; CSY20; CSK*21] or character control [HBM*20] with regard to RNNs but more frequent in other tasks where fixed-length motion sequences are more suitable, e.g. motion editing [LCC19; ZYC*20; SGXT20; SAA*20; KPKH20; ALL*20; HHKK17; AWL*20; DAS*20; LAT21] (see Section 5).

2.4.3. Miscellaneous

Other approaches to better model the temporal flow of human motions include motion phase representation, spectral decomposition of motion data and spatio-temporal attention. For instance, several authors investigated the representation and learning of the phase of movements in the context of kinematic character control, which is detailed in section 4.1. In the character controller network proposed by Holden et al. [HKS17], the network weights are computed as a spline function of the phase, whose control points, representing network weights configurations during the human locomotion cycle, are learned. Other works [ZSKS18; SZKS19; SZKZ20; LZCvdP20] use a gating network instead of the cyclic phase to blend expert weights, resulting in a mixture-of-experts scheme. In the context of motion prediction, Mao et al. [MLSL19; MLS20] and Cai et al. [CHW*20] learned temporal correlations in the frequency domain by applying at the input and the output of their network a *Discrete Cosine Transform* (DCT) and its inverse, respectively.

3. Motion Synthesis

Synthesising motion data is of strong practical interest to the media and entertainment industry. Besides generating realistic and diverse sequences, a key challenge is to be able to control various aspects of the motion using high-level parameters. In this section we distinguish the tasks of short-term (Section 3.1) and long-term (Section 3.2) motion prediction, and motion generation (Section 3.3). The first focuses on the short-term deterministic extrapolation of motion, i.e. up to one or two seconds, from a past conditioning clip, while the second operates beyond this temporal horizon, with the purpose of generating plausible and diverse continuation of the observed motion. Indeed, reproducing ground-truth animations quickly becomes impossible because of the stochastic nature of human motion [FLFM15; PGA18; ZLX*18]. On the contrary, the ability to generate diverse sequences is often a desired feature in long-term motion prediction. In the third category, the goal is to synthesise from non-historical parameters pose sequences that are both consistent with the distribution of samples in a reference training dataset, and sufficiently diverse to capture its variations. We restrict our survey to general-purpose approaches and exclude works targeting application-specific contexts such as the synthesis of gestures from speech or dance animations from music. We summarise the methods presented in this section in Table 2.

This article has been published in COMPUTER GRAPHICS *Forum* (CGF). https://doi.org/10.1111/cgf.14426

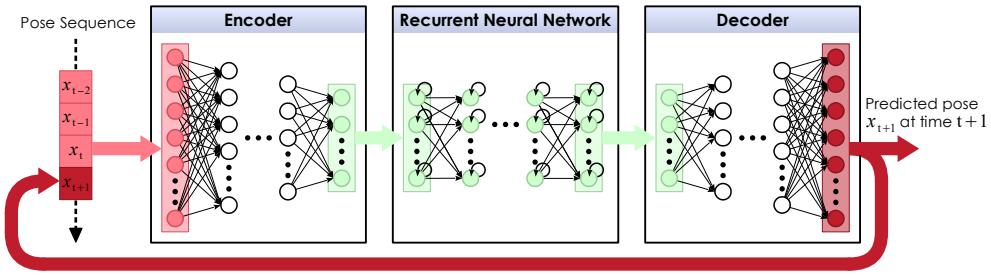

Figure 5: *Illustration of the* Encoder-Recurrent-Decoder *(ERD) architecture originally proposed by Fragkiadaki et al. [FLFM15] and leveraged in both short and long term motion prediction in its original formulation and modified/extended versions. An RNN captures motion dynamics in a latent space. The encoder and decoder feedforward DNNs map skeletal poses to this latent representation and back.*

3.1. Short-term Prediction

Short-term motion prediction consists in observing the motion on a given temporal horizon and predicting the motion for the near future. Let us note X_t the 3D pose of a skeleton at time t: when observing the set of poses $X_{t-T_p},\dots,X_t$, we aim at predicting future poses $Y_{t+1},\dots,Y_{t+T_f}$ where T_p and T_f denote the past and future temporal observation and prediction windows, respectively. Motion prediction is useful for many applications including robotic navigation in crowds, human-robot interaction, video surveillance, virtual reality experiences or cloud gaming. In particular, the capability to predict the 3D positions of human end effectors rather than trajectory provides a much richer and helpful information.

Classical approaches have been developed roughly between 2000 and 2015 using techniques such as spatio-temporal autoregressive models, Hidden Markov Models or Gaussian processes. These "classical" techniques are out of the scope of this section. With the development of DL techniques, as well as the production of larger mocap databases (as presented in section 2.2), it has become a natural path to use DL for human motion prediction. There are multiple difficulties in this particular task of predicting human motion with DL, namely:

- When the motion is stationary and periodic, the repeating patterns can be discovered and used for future prediction. Unfortunately, human motion in real-world scenario is rarely stationary. The first challenge is thus to design deep models that account for aperiodic and non-stationary motion.
- Articulated human motion can only be efficiently predicted if spatial and temporal dependencies can be captured. For instance, modelling the dependency between the right arm and the left leg during a walk cycle. The second challenge is to model these spatial and temporal dependencies.
- DL offers the advantage of leveraging large quantities of data. However, the plausibility of the future prediction given the past observations should be explicitly enforced. In particular, Martinez et al. [MBR17] showed that many classical ML approaches can be beaten by a simple baseline: the zero-motion prediction (this can partially be explained by the discontinuity of the first frame prediction). The third challenge is therefore to ensure the plausibility and realism of the predicted motion.
- Humans rarely act in an empty environment. Constraints such as objects, or other humans in the scene, should be taken into account for prediction. The fourth challenge is to predict future motion in a dynamic and constrained environment.

We first review the existing methods, organised w.r.t. framework used. Then, we examine the existing answers that authors have proposed to the four aforementioned challenges.

3.1.1. Classes of methods

Various approaches have been applied to the problem of short-term motion prediction, and are presented according to the framework they built on, namely: *Sequence to Sequence* (Seq2Seq), direct feedforward networks, *Graph Convolutional Networks* (GCNs) and *Generative Adversarial Networks* (GANs).

Sequence to Sequence. Popular and widely used in motion prediction, Seq2Seq-based approaches generally consist in training an RNN in some latent space. Both the RNN and the hidden latent representation are trained jointly. Known issues of these techniques are the convergence to a mean pose, as well as finding the trade-off in exposing the model to its own errors so that it can recover from deviations. We describe hereafter the main solutions proposed in the literature.

Fragkiadaki et al. [FLFM15] pioneered the *Encoder-Recurrent-Decoder* (ERD) architecture, in which an LSTM network is trained in the hidden layer of an autoencoder, enabling the network to learn a representation suited for motion prediction, as illustrated in Figure 5. Papers that pursued this scheme generally opted for a GRU module afterwards.

Some papers incorporated the skeletal structure in the network. Jain et al. [JZSS16] learnt motion dynamics using a spatio-temporal graph modelling body parts, using an RNN architecture, which can be viewed as a structural-RNN architecture. Aksan et al. [AKH19] proposed a structured prediction mechanism, composed of a hierarchy of sub-layers connected to the kinematic chain of the skeleton. This enables an explicit modelling of body part motion prediction at a local and global scale. This structured prediction is compatible with different flavours of joint representation (exponential map or unit quaternions), as well as different types of prediction networks (simple RNN, Seq2Seq, etc.). Li et al. [LWY*19] proposed

This article has been published in COMPUTER GRAPHICS *Forum* (CGF). https://doi.org/10.1111/cgf.14426

an approach where the RNN is trained on a hierarchical decomposition of joints, as well as on two different temporal horizons to maintain global consistency. Yang et al. [YKL21] used an RNN network based on GRUs to predict the lower body motion, given past observations of the upper body.

Although the prediction of joint positions is usually preferred, some authors opted for different pose representations and the associated loss function. To improve recent RNN approaches, Martinez et al. [MBR17] proposed the following improvements: first, instead of feeding unrealistic noise at learning, they incorporated network predictions to drive the network to recover from its own mistakes. Second, they proposed a residual architecture that takes into account first-order derivatives. Hence, they used a GRU in the latent representation, forcing weight sharing between encoder and decoder: motion continuity is enforced through residual connections. Chiu et al. [CAW*19] learned a hierarchical and multi-scale latent representation where an RNN is trained to predict human velocities. Gui et al. [GWLM18] proposed a GRU-based Seq2Seq architecture, incorporating adversarial training to decrease the short-term discontinuity and increase the long-range realism. In addition, they advocated that the loss function should be based on a geodesic loss, leveraging the properties of the Lie structure of orthogonal matrices. Pavllo et al. [PGA18; PFAG20] proposed an autoregressive model based on two GRU layers in the space of quaternions, therefore avoiding the gimbal lock issue at the expense of constraining quaternions to remain on the unit 4-sphere (see Section 2.1.2). The total loss for the pose is composed of a quaternion loss and a positional loss obtained after FK.

Finally, some methods took inspiration from the Seq2Seq framework but replaced the RNN prediction layer: Li et al. [LZLL18] used two encoders to learn latent representations of both short and long term horizons. A convolutional architecture is trained instead of an RNN to predict future poses. Xu et al. [XLM19] proposed a hierarchical Seq2Seq model that encodes each sequence into a latent representation. The prediction is then seen as a completion in this latent representation, obtained by vector addition. A single fully connected completion network is trained to this goal. Gui et al. [GWRM18] leveraged a meta-learning scheme within a Seq2Seq approach using GRUs to adapt the model's parameters to new training examples. To the best of our knowledge, the work of Wang et al. [WAC*19] is the only approach using DRL for motion prediction. The prediction problem is decomposed into K steps, and the progressive prediction is trained using an imitation learning approach.

Direct Feedforward. Less used than Seq2Seq-based methods, direct feedforward techniques aim at computing the prediction network $\mathcal{F}$ directly from the entire set of poses given the past observations: $Y_{t+1},\dots,Y_{t+T_f} = \mathcal{F}(X_{t-T_p},\dots,X_t)$.

Bütepage et al. [BBKK17] trained such a feedforward network after a temporal encoding of human motion. Feature learning is evaluated thanks to an action classifier. Guo and Choi [GC19] proposed a combination of two DNNs for human motion prediction: the first network, called *SkelNet*, assembles the prediction of different body parts, while the second network, called *Skel-TNet*, accounts for temporal dependencies modelled through an RNN. Zang et al. [ZPK20] proposed a deformable spatio-temporal convolution approach that captures in the past sequence the most relevant poses, through a masking mechanism. Based on this temporal attention, parameters for the prediction of the future motion are generated on the fly, as in few-shot learning methods. Cai et al. [CHW*20] leveraged the increasingly popular transformer architecture [KNH*21]. Originally designed for processing data sequences in NLP, it relies on attention mechanisms to better account for long-range dependencies than RNNs. The attention weights learnt in a transformer model capture the relevance of data sequence items to other items that can be far apart. Because these weights modulate the input data prior to convolution, transformer networks can be viewed as extensions of CNNs and RNNs in which the learnt convolution kernels are made data-dependent. This increased expressiveness comes at the price of larger training data volume requirements. More specifically, in [CHW*20] joints are converted using a *Discrete Cosine Transform* (DCT). The decoding of the future pose is done progressively in accordance with the skeleton topology.

Graph Convolutional Networks. As described in Section 2.3.3, *Graph Convolutional Networks* (GCNs) leverage the graph nature of the human skeleton, and have therefore been used to design prediction methods that act on such graphs. They were first used in the context of short-term motion prediction by Mao et al. [MLSL19], who proposed an approach where poses are encoded in a latent space using DCT decomposition. This representation is then fed to a GCN for feedforward prediction, while the connectivity of the graph is learned through the adjacency matrix, which enables to learn the temporal and spatial dependencies between body joints. They also proposed an attention mechanism applied to the DCT coefficients of the observed sequence [MLS20], where the main idea is to focus on relevant history information to predict the future poses. Approaches based on GCNs have also been proposed to capture skeletal and motion information at different scales. For instance, Li et al. [LCZ*20; LCC*21] used GCNs to capture the skeleton structure over different scales, ranging from individual joints to increasingly large body parts. A cross-scale fusion block then merges the features over scales and feeds this representation into a GRU-based decoder [LCZ*20]. As an alternative, scales can be analysed in parallel branches to extract features used both for action recognition and motion prediction [LCC*21]. The predicted motion category is then used in the motion prediction network. Similarly, in order to capture relevant motion information at different scales, Lebaily et al. [LKS*20] proposed a temporal inception module, using 1D temporal convolutions at different scales, combined with a GCN to predict the future poses. Unlike previous approaches that represent skeletal connections with a single adjacency matrix, Cui et al. [CSY20] relied on a double graph convolutional approach to better learn the dynamic relationships between skeletal joints, therefore learning a connective graph to represent the natural kinematic links of the human skeleton and a global graph to account for the dynamics of non-connected joints. Convolutions over these two graphs are then used to predict future poses. GCNs have also been used to explore the prediction of human-object interactions: Corona et al. [CPAM20] built on the work of Martinez et al. [MBR17] and explicitly incorporated the modelling of such interactions. As adjacency matrices typically vary in time

This article has been published in COMPUTER GRAPHICS *Forum* (CGF). https://doi.org/10.1111/cgf.14426

in these dynamic interactions, the resulting graph representation is learned to capture their dynamics.

3.1.2. Spatial and Temporal Dependencies

One difficulty in motion prediction is to leverage the spatial and temporal dependencies that exist in human motion. In the spatial domain, human joints are directly related to their parents but indirectly related to other joints (e.g. symmetry or regularity in motion). In the temporal domain, the repeating patterns (or information redundancy) can greatly help in predicting the future motion. Modeling the motion over various scales greatly helps for short-term motion prediction. This is closely related to attention mechanisms that have been well studied in the DL community. Researchers started to investigate spatial dependencies first, then rapidly studied both spatial and temporal dependencies especially through temporal attention mechanisms. We here review the different solutions that have been proposed in the literature.

Jain et al. [JZSS16] used a spatio-temporal graph that explicitly models the body structure (spine, arm, and leg) and captures the spatial dependencies. Bütepage et al. [BBKK17] learned a temporal encoding of motion using different time scales and convolutions, while a graph network is used to encode the hierarchical structure of the skeleton. Li et al. [LZLL18] explicitly used two latent representations for short-range and long-range observation. By doing so, they expected to benefit from long-range motion consistency, as well as to improve the dynamics of short-term prediction. Cui et al. [CSK*21] used a temporal attention mechanism, jointly with dilated convolutions, to capture long-range motion features. Capturing temporal and spatial dependencies between body joints has also been explored using GCNs [MLSL19; CSY20; LCZ*20; MLS20; LKS*20], as already detailed in Section 3.1.1, to match the graph nature of the human skeleton. Cai et al. [CHW*20] leveraged the transformers concept that learns the spatial correlations as well as the temporal smoothness of the predicted skeletons.

3.1.3. Plausibility and Realism

Data-driven approaches to predict future motion can face the issue of predicting unrealistic poses in terms of biomechanics. In addition, motion prediction can suffer from a lack of realism whatever the temporal range of prediction. We present here the proposed solutions to these problems.

To improve plausibility, Gui et al. [GWLM18] included two discriminators at training time: a fidelity and a continuity discriminator. The fidelity discriminator assesses whether the prediction is smooth enough, while the continuity discriminator encourages the prediction to be consistent with past observations, thus limiting the discontinuities. It is also possible to rely on a discriminator to discriminate from real sequences, however results show that this improves the prediction by a small margin only [LZLL18]. Cui et al. [CSY20] used the Gram matrix in the loss function to add more consistency between the predicted and ground-truth poses. They also included bone length preservation.

Pavllo et al. [PGA18] proposed to mix a *teacher-forcing* approach (i.e. feeding the network with ground-truth motions) with an approach where the network is fed with its own predictions. The network is first trained with teacher-forcing and progressively switches to its own predictions through a curriculum schedule. This progressive training improves the error and the model stability. Cai et al. [CHW*20] used a dictionary mechanism, similar to a memory cell. This dictionary encodes motion that could be similar but seen in slightly different contexts. Bourached et al. [BGG*20] proposed an out-of-distribution approach, where samples can be augmented thanks to a *Variational Autoencoder* (VAE) learned on Human3.6 [IPOS14] and CMU [Uni03] datasets. Using graph convolutional layers, the VAE models connectivity, positions and temporal frequencies, which helps to limit the distribution shift and to regularise the training. More specifically, this mechanism helps producing larger quantities of plausible training data.

3.1.4. Context, Environment and Interactions

Modelling human motion in an empty environment, without objects or humans, is already a difficult task. However, context, environment and surrounding characters are crucial for predicting motions in real-life scenarios, especially to avoid unrealistic situations, e.g. character collisions or incorrect interactions with the environment. This is an even more complex task, mainly due to the lack of databases and the large variety of situations that can be encountered.

To better predict interactions between a human and all objects of the scene, Corona et al. [CPAM20] proposed to incorporate the context and the interaction between humans and objects in a novel context-aware motion prediction architecture. To do so, a convolutional method over the graph of objects and persons is learned, building on the RNN implementation proposed by Martinez et al. [MBR17]. The graph relationships are also learned, initialised so that the prediction of one object only depends on itself, and progressively accounting for interactions. Similarly, to account for both scene and social contexts in the prediction, Adeli et al. [AAR*20] proposed a Seq2Seq approach with a GRU architecture, where the scene and social contexts are captured independently through the analysis of monocular video. The spatio-temporal features are first extracted by a CNN, then fed to the decoder module to account for context.

3.2. Long-term Prediction

The goal of long-term motion prediction is either to extrapolate a past conditioning clip to the future (i.e. forecasting, see Section 3.2.1) or to interpolate between known past and future character poses, often far apart both temporally and spatially (i.e. in-betweening, see Section 3.2.2). Unlike short-term prediction approaches, the time horizons considered are larger and the synthesised motions are not constrained to reproduce the ground truth. Instead, the goal at large is to generate a diversity of plausible continuations of the original clip.

3.2.1. Forecasting

Most of the approaches in this category are based on the *Encoder-Recurrent-Decoder* (ERD) model originally proposed by Fragkiadaki et al. [FLFM15] for motion prediction and presented in section 3.1. The generation is conditioned by a past window of mo-

This article has been published in COMPUTER GRAPHICS *Forum* (CGF). https://doi.org/10.1111/cgf.14426

tion frames. The temporal context is captured by an RNN operating in the latent space at the output of the encoder, as illustrated in Figure 5. As pointed out by many authors [HHS*17; ZLX*18; LWJ*19; WCX21; GWE*20], future motion predicted by ERD networks tend to converge to a motionless state or to diverge from human motion. This is attributed to several issues that many of the approaches presented below aim at mitigating. Although some of these issues are also relevant to short-term motion prediction (section 3.1), they are exacerbated when targeting longer prediction horizons.

A major downside of the ERD architecture is the accumulation of errors along time at the output of the RNN. As a result, the quality of the generated motion degrades as the prediction moves away from the past conditioning sequence. Wang et al. [WCX21] and Kundu et al. [KGB19] treated the ERD model as the generator of a GAN and added a corresponding discriminator to improve the plausibility of the motion predictions. Kundu et al. [KGB19] concatenated a stochastic component r to the output of the ERD encoder and complemented the discriminator with a critic network that regresses r from the predicted motion sequence. The regressed value is fed back to the ERD decoder to refine the predicted sequence. This encourages the learning of a one-to-one-mapping between the stochastic input to the GAN and the generated motion, thereby minimising the risk that the output collapses to a single mode of the motion distribution, a well-known issue with GANs [GPM*14].

Several authors [HHS*17; GWE*20] ascribed the regression towards a static pose to the ill-posedness of long-term motion prediction and advocated the use of external control signals to disambiguate the task. For instance, Ghorbani et al. [GWE*20] built the RNN cell of their network around a conditional VAE that is conditioned on external constraints, such as action type and character gender, as well as on the hidden state of the RNN at the previous time step. Cao et al. [CGM*20] conditioned their motion prediction scheme on the environment the character moves in, specified using a 2D picture. The past motion history is provided as a sequence of 2D joint heatmaps in the scene. A first conditional VAE network samples a plausible 2D target end location of the predicted motion sequence inside the environment, from which a second network predicts the 3D trajectory of the character center. Finally, a third network promotes this point trajectory into a 3D pose sequence.

The RNNs at the core of the ERD model create other issues. Training an RNN using ground-truth future samples rather than exposing it to its own predictions creates a difference in behaviour between the training and inference stages that is detrimental to performance, a phenomenon referred to as exposure bias [PGA18; GWE*20]. As a mitigation measure, Zhou et al. [ZLX*18] proposed an auto-conditioning network that is trained alternately in open loop on ground-truth motion and in closed loop on its own predictions. Gopalakrishnan et al. [GMK*19] strike a balance between the two training modes in their loss function and gradually increase the weight of the closed-loop term. RNNs have a tendency to overly focus on recent poses and fail to capture long-term temporal dependencies of human motion. To counter this effect, Tang et al. [TMLZ18] augmented their RNN with a temporal attention mechanism that encourages pose predictions to align to previous poses in a pose embedding. To the same purpose, Liu et al. [LWJ*19] proposed a hierarchical RNN cell that captures a global motion context in addition to the RNN hidden state at every frame.

Unlike for short-term motion prediction where the goal is to minimise deviations from the ground truth, diversity and randomness in the generated motion sequences is encouraged by complementing the past sequence input with a stochastic component. In ERD networks this can be achieved by adding random noise to the latent pose representations at the output of the encoder [XLM19], or by stacking a random vector to the latent representation of the RNN [KGB19]. Stochasticity can also be embedded in the RNN cell, as in [GWE*20] where this cell is built around a conditional VAE whose random latent code drives motion generation.

Other works proposed more expressive models of the skeleton or of the motion dynamics to improve the realism of synthesised animations. For instance, representations of the articulated bone structure as quaternions [PGA18] or as 3D rigid transformations in the Lie algebra $\mathfrak{se}(3)$ [LWJ*19] were shown to generate more natural motion sequences. Please refer to Section 2.1 for a detailed presentation of these approaches. Gopalakrishnan et al. [GMK*19] incorporated short-term motion history in their representation of pose by complementing joint angles with their local temporal derivatives up to order 3. The prediction of motion dynamics is cast by Wang et al. [WCX21] in a probabilistic framework. The transition of character pose x_t at time t to the next frame is expressed by a probability density function $p(x_{t+1} \mid x_t)$, modelled as a multivariate *Gaussian Mixture Model* (GMM). Their network builds on two RNN cells that predict the GMM parameters at each time step by maximising its likelihood.

Many authors acknowledged that learning a motion manifold for a large diversity of styles and movements is a difficult problem. To facilitate this task, several authors [GSAH17; LWJ*19; GMK*19; WHSZ19] learned separate sub-networks for modelling the spatial and temporal components of human motion. The prediction network of Gopalakrishnan et al. [GMK*19] is a two-level hierarchical RNN where the upper layer sketches a trajectory for future motion while the lower layer synthesises poses on this trajectory. The decoder in the ERD network architecture of Wang et al. [WHSZ19] features a forward and a backward motion predictor, whose outputs are concatenated to reconstruct the full-length motion sequence. The authors argue that since the backward predictor forces the decoder to pay attention to the last few frames output by the encoder, it encourages the model to first learn short-term temporal correlations before accounting for longer-term dependencies.

Unlike all previous approaches, Hernandez-Ruiz et al. [HGM19] did not rely on an RNN. They instead reformulated motion prediction as a temporal inpainting task and built their model on a GAN. They fed the generator with the past conditioning motion clip, and encoded it into frame-specific latent representations that are processed at multiple scales before being decoded into an output pose sequence.

3.2.2. In-betweening

The task of interpolating motion between starting and ending character poses, often far apart, is referred to as *In-betweening*. It can

This article has been published in COMPUTER GRAPHICS *Forum* (CGF). https://doi.org/10.1111/cgf.14426

be viewed as a long-term prediction task conditioned on past and future contexts.

Yu et al. [YKK*19] proposed a simple scheme for interpolating a motion sequence from starting and ending positions of end-effector joints. Their main objective is computational efficiency, in order to generate terrain-adaptive character motion in real time for video games. They cascaded two fully connected networks: the first interpolates the trajectories of the end effectors in time, the second infers full poses at each frame.

The main purpose of other works is to generate plausible and diverse motion over long transitions bounded by a starting and an ending keyframe. Kaufmann et al. [KAS*20] leveraged a deep convolution denoising autoencoder, in which pooling layers ensure large receptive fields to capture long-range spatial and temporal joint correlations. A curriculum learning scheme feeds the encoder with sequences containing increasingly large temporal gaps to improve the training. Xu et al. [XXN*20] proposed a temporally hierarchical scheme in which the transition segment is split into equal length sub-segments. Trajectory constraints are provided at each sub-segment endpoint. The transition sequence is initialised by sampling a motion clip from the training dataset for each sub-segment. Next, the style of each clip is changed to match the style of a reference sequence throughout the whole transition sequence. This stage leverages a motion autoencoder with separate content and style embeddings, that is trained in an unsupervised way. Style transfer is performed by linearly combining the content latent code of the initial clips and the style latent code of the reference style sequence. Finally, transitions between the endpoints of consecutive sub-segments are generated using forward and backward LSTM networks, and the plausibility of the generated sequence is enhanced by combining the generation network with a discriminator in a GAN framework. Harvey et al. [HYNP20] pointed out that in-betweening between distant keyframes may result in stalling or teleportation artifacts if the temporal evolution of the motion is not monitored during the generation process. To deal with this issue, they proposed an ERD architecture that is fed with the pose representation deltas between the current and the target frame, in addition to the character pose at the current timestep and the end pose. The time-to-arrival is encoded in a sine wave and added, rather than stacked, to the latent code that is input to the RNN, forcing the network to take this piece of information into account during training.

3.3. Generative Synthesis

Unlike motion prediction methods, approaches covered in this section synthesise human motion by processing a random seed with a deep generative network, optionally conditioned on semantic cues typically pertaining to the character trajectory or motion style. They are grouped in the following sub-sections based on the deep generative framework they build on. As an exception, in the scheme proposed by Holden et al. [HSK16] the synthesis process is purely deterministic and driven by high-level cues.

3.3.1. Restricted Boltzmann Machines

Initially proposed by Smolensky [Smo86], *Restricted Boltzmann Machines* (RBMs) build on a parametric expression of the probability density function of the data distribution from which new

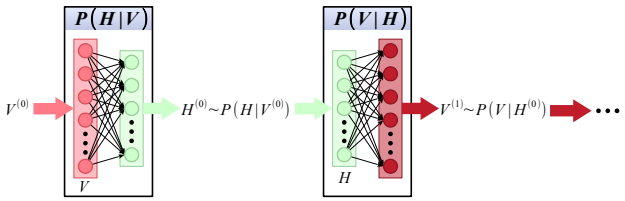

Figure 6: *A* Restricted Boltzmann Machine *(RBM) can be represented by two fully-connected neural networks whose outputs provide the conditional probabilities of hidden variables H given visible variables V and vice-versa. Sampling from these probabilities builds a chain of hidden $H^{(j)}$ and visible $V^{(j)}$ samples that converges to the model distribution. The chain is initiated by drawing $V^{(0)}$ from the training set.*

samples are to be generated. This density depends on visible units V that correspond to the variables to be synthesised and hidden units H. An RBM can be represented as a bipartite graph in which hidden units are conditionally independent given the visible units, and vice-versa. The form of the probability density function $P(H,V)$ in an RBM provides simple closed-form expressions for $P(H|V)$ and $P(V|H)$ as fully connected layers with a sigmoid activation. These layers are usually trained using an approximate gradient ascent scheme known as *Contrastive Divergence* [Hin02]. Once $P(H|V)$ and $P(V|H)$ have been determined, new samples are generated using a *Markov Chain Monte Carlo* (MCMC) sampling process initiated by visible units $V^{(0)}$ drawn from the training dataset. Given $V^{(i)}$, $P(H|V^{(i)})$ is computed and hidden units $H^{(i)}$ are drawn from that distribution. Next, evaluating and sampling from $P(V|H^{(i)})$ yields $V^{(i+1)}$. This alternated sampling scheme between $V^{(i)}$ and $H^{(i)}$ is stopped after K steps, $V^{(K)}$ providing the sought generated sample (see Figure 6).

Conditional RBMs. Taylor et al. [THR06] extended the RBM generative framework to the synthesis of time series by introducing an autoregressive model that conditions the visible and hidden variables at the current time step on the visible variables at past time steps. The formulation of this *Conditional RBM* (CRBM) differs from the original RBM only by additional bias terms in the expressions of $P(H|V)$ and $P(V|H)$. These biases are learnt during training. The optimisation and sampling processes are otherwise unchanged. Motion sequences can be generated using a CRBM by mapping character joint positions or rotations to its visible units. When the training dataset contains different categories of motion, the generated category is dictated in principle by the selection of the initial MCMC sample $V^{(0)}$, although transitions between different types of motion can be forced by adding noise to the hidden states.

Factored CRBMs. The same authors [TH09] proposed the *Factored CRBM* (FCRBM) architecture to better control the motion category or style. It is essentially a gated CRBM consisting of three layers: the visible units at the previous time steps form the input layer, the same units at the current time step are the output layer, and the connections between these two layers are gated using mul-

This article has been published in COMPUTER GRAPHICS *Forum* (CGF). https://doi.org/10.1111/cgf.14426

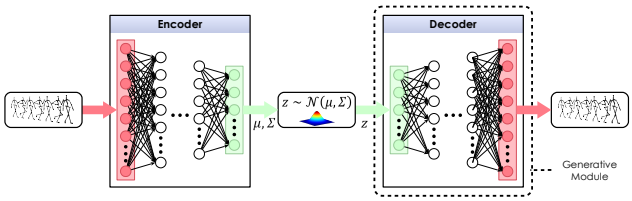

Figure 7: *In a* Variational Autoencoder *(VAE) a motion sequence is mapped by an encoder to a random latent code that is constrained to follow a prior Gaussian distribution. This code is mapped back by a decoder to the input motion data. Once the full network is trained, feeding the decoder with samples drawn from the prior generates stochastic motion sequences.*

tiplicative weights that form a third hidden layer. To obtain a more efficient representation, the third-order interaction tensor between the 3 layers is factored into three matrices of pairwise interactions. The weights in the hidden layer are defined as linear functions of a one-hot vector of motion style labels that control motion synthesis. Transitions between motion styles can be introduced at will by appropriate settings of the motion style labels over the sequence. Alemi et al. [ALP15] applied the FCRBM model to the generation of controlled affective variations of walking motion, based on time-varying valence and arousal labels.

Hierarchical FCRBMs. Chiu and Marsella [CM11] noted that CRBM and FCRBM approaches are prone to overfitting when trying to generalise to a rich set of styles and motion, because the training dataset can only sparsely sample the set of all possible style combinations and transitions. To mitigate this issue, they proposed a *Hierarchical FCRBM* (HFCRBM) model, corresponding to a *Deep Belief Network* with an FCRBM on top of a simplified CRBM where the temporal dependency on past visible states has been removed. The dynamics of the synthesised sequence is controlled exclusively by the upper FCRBM. In their multi-path approach, a separate FCRBM is learnt for each motion style, the output visible layers are linearly combined with predefined weights defining the desired motion style, and the result is fed to the bottom layer of the HFCRBM. Thus, rather than relying on a unique FCRBM to represent all styles, the mixing of styles is performed by mixing individual FCRBM instances within the hidden layer of the HFCRBM.

3.3.2. Variational Autoencoders

Structured as an encoder followed by a decoder, an autoencoder is a DNN whose input and output represent the same data. The encoder output provides an intermediate latent code with a dimension often lower than the input data. Thus, autoencoders provide a scheme for non-linear dimensionality reduction. As illustrated in Figure 7, the distribution of the latent codes in a *Variational Autoencoder* (VAE) [KW14] is further constrained to follow a predefined prior distribution, typically a multivariate normal distribution, endowing the autoencoder with a generative capability. Thus, VAEs provide a convenient way to embed a stochastic component into the generation of human motion.

Various approaches have been proposed to extend the VAE framework to the modelling of temporal sequences. Toyer et al. [TCHG17] relied on a Deep Markov Model, essentially a VAE in which the latent code is conditioned on its value at the previous time step. Habibie et al. [HHS*17] combined a VAE and an RNN. Motion generation is driven by the random latent code samples, as well as by control variables that constrain the trajectory and velocity of the character. The RNN is conditioned on encodings on these control variables. At inference time, its cell state is initialised with the latent code value. The concatenation of cell state outputs at each time step is fed to the decoder to produce the synthesised motion sequence.

Du et al. [DHS*19] built on the motion graph framework proposed by Min and Chai [MC12], in which motion sequences are represented as a graph of motion primitives. The segmentation of motion sequences into primitives is dependent on the type of motion and typically hand-crafted. Autoencoders learn embeddings for each primitive, and the latent codes for these embeddings are further encoded by Conditional VAEs trained on dataset samples for the considered primitives. The conditioning of the primitive-specific VAEs ensures that they reproduce the style of the input motion, which is encoded as a Gram matrix in the embedding space, following prior work in motion style transfer [HSK16] (see Section 5.3). To synthesise a motion sequence, a path is determined in the motion graph based on user-defined trajectory controls, and motion primitives are generated along this path using the VAEs.

Yan et al. [YRV*18] and Aliakbarian et al. [ASS*20] relied on similar network architectures for stochastic motion prediction. Pose sequences are mapped to lower-dimensional features by an encoder and transformed back to motion data by a decoder. The VAE operates on the encoded features, its output is fed to the decoder to synthesise the motion clips. Yan et al. [YRV*18] processed small pose sequences called *motion modes* that capture short-term motion features. During training, their VAE maps a pair of (past, future) mode features to a random latent code (encoder) then to a prediction of the future mode feature (decoder). It thereby captures the transition between the two modes. At inference time, the VAE and RNN decoders generate a stochastic prediction of the future mode, given a past conditioning mode and a draw of the VAE random latent code. Aliakbarian et al. [ASS*20] argued that stacking the past sequence information and the random generating seed in a vector and feeding it to the decoding network leaves the possibility that the stochastic component is assigned low weights during training and is thus effectively ignored by the network. This concern is confirmed by experimental evidence. To avoid this, they proposed a *mix-and-match perturbation* strategy and formed a vector by replacing randomly selected components of the past sequence feature by corresponding components of the VAE latent code. Feeding this vector to the decoder forces it to account for both the past context and the stochastic input.

This article has been published in COMPUTER GRAPHICS *Forum* (CGF). https://doi.org/10.1111/cgf.14426

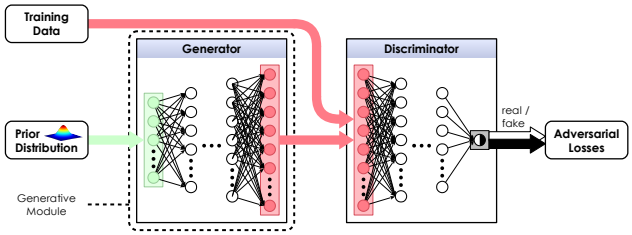

Figure 8: *A* Generative Adversarial Network *(GAN) consists of two networks: the generator produces "fake" motion samples from random seeds, the discriminator tries to differentiate them from "real" ones drawn from the training dataset. The two modules are trained jointly, aiming at an equilibrium where the generator outputs cannot be distinguished from the training data.*

3.3.3. Generative Adversarial Networks

Generative Adversarial Networks (GANs) are a popular alternative to VAEs for generating samples from random seeds. In a GAN, new samples are synthesised by a generator network that operates in conjunction with a discriminator network (see Figure 8). The generator transforms a random seed drawn from a known prior distribution to a sample which aims to be similar to the contents of the training dataset, the discriminator assesses this similarity. The two networks are trained jointly with adversarial losses, the generator being driven to produce samples that the discriminator ultimately should not be able to distinguish from the training dataset samples.

Barsoum et al. [BKL18] pointed out that relying on a GAN for long-term motion prediction has several intrinsic advantages. GANs do not suffer from the temporal error accumulation issues of RNNs and will produce one instance of all possible outputs instead of averaging these possible outputs as ERD networks tend to do. GANs also natively embed a stochastic component. Any motion generation network can be wrapped into a GAN by adding a discriminator network to improve the plausibility of the generated sequences [BKL18; WCX21; HYNP20]. Other works focused their contribution on optimising the architecture of the GAN generator network to facilitate and improve its training. Wang et al. [WACD20] proposed an adversarial autoencoder architecture [MSJG15] where a GAN enforces a prior on the distribution of the latent code of an autoencoder. Yan et al. [YLX*19] fed the generator with a sequence of random samples drawn from a Gaussian process with a predetermined covariance function. Through a series of purely convolutional modules made up of a spatio-temporal upsampling layer followed by a graph convolution on the skeleton features, they gradually increased the spatial and temporal resolution of their output to produce a pose sequence. Wang et al. [WYZ*20] split their generator into separate spatial and temporal sub-networks: the lower layer maps the input random seed and conditioning action label via an RNN to a sequence of low-dimensional latent codes that model temporal transitions. The upper layer decodes each latent code into a skeletal pose. The generator is regularised by a helper action classifier network through a cycle consistency constraint: the conditioning action label fed to the generator should agree with the result of the classification of the motion sequence it produces.

3.3.4. Normalising Flows

Generative models based on *Normalising Flows* [DKB14] synthesise samples by applying a composition of invertible elementary transforms to a latent variable drawn from a known prior distribution. Unlike in GANs or VAEs, owing to the form of the elementary transforms, the probability density function of the generated samples can be computed in closed form. Thus, the generative network can be trained by maximum likelihood optimisation. Henter et al. [HAB20] adapted this framework to the generation of motion sequences. Each transform layer in the generator network is conditioned by a control signal that encodes the past trajectory of the root joint and holds an LSTM unit whose hidden state captures temporal dependencies.

3.3.5. Miscellaneous

Deterministic Generation. Holden et al. [HSK16] conditioned their motion generation approach on purely deterministic constraints that specify the trajectory and velocity of the character. They leveraged an autoencoder operating on temporal chunks of poses to learn a latent manifold of human motion. A separate feedforward network maps the autoencoder embedding to high-level, semantically interpretable controls of the motion. Generating high-dimensional motion embedding samples from a set of low-dimensional control parameters is severely under-constrained. To disambiguate the generation to the largest possible extent, the scope of the approach is restricted to human locomotion, and a comprehensive set of trajectory and foot contact constraints is imposed.

Deep Generative Models Sampling. After training a deep generative model driven by an input random seed, motion sequences are often obtained by drawing independent samples from the seed and feeding them to the generator network. Yuan et Kitani [YK20] proposed a better sampling strategy to enforce diversity in the generated sequences and cover minor modes of their distribution. To this end, they computed correlated random seeds by applying a set of affine transforms to a random variable drawn from a Gaussian distribution. The parameters of the affine transforms are learnt by a front-end network that can be conditioned on the same conditioning data used by the generative model. Each seed generates one motion sequence. Diversity among these sequences is enforced through a repulsion loss term that maximises the dissimilarity between the generated sequences. Another loss term enforces consistency of the random seeds with the prior distribution of the generative model. The relative weights of these two terms define a trade-off between the conflicting goals of diversity and fidelity to the training data distribution.

This article has been published in COMPUTER GRAPHICS *Forum* (CGF). https://doi.org/10.1111/cgf.14426

Table 2: *Summary of the methods presented in Section 3. Miscellaneous data includes hand-crafted, synthetic, proprietary, unspecified or other public datasets.*

Reference	Code	Section			Framework	Dataset							Architecture										Representation
		3.1 Short-term Prediction	3.2 Long-term Prediction	3.3 Generative Synthesis		3DPW [vMHB*18]	CMU [Uni03]	Human3.6M [IPOS14]	HDM05 [MRC*07]	Holden et al. [HSK16]	NTU RGB+D[SLNW16]	Miscellaneous	Fully Connected	Convolutional	Graph Convolutional	Recurrent	Transformer	Autoencoder	RBM	Variational Autoencoder	Adversarial	Normalising Flows	
[FLFM15]		×			DL			×								×							$so(3)$
[JZSS16]	</>	×			DL			×								×							$so(3)$
[BBKK17]		×			DL		×	×					×										$so(3)$
[MBR17]	</>	×			DL			×								×							$so(3)$
[GWLM18]		×			DL			×								×					×		$so(3)$
[GWRM18]		×			DL			×								×							Unknown
[LZLL18]	</>	×			DL		×	×						×							×		$so(3)$
[AKH19]	</>	×			DL			×								×							SO(3)
[CAW*19]	</>	×			DL			×				×				×							3D velocities
[GC19]	</>	×			DL		×	×								×							$so(3)$
[LWY*19]		×			DL		×	×						×									$so(3)$
[MLSL19]	</>	×			DL	×	×	×							×								DCT
[WAC*19]		×			DRL			×								×					×		$so(3)$
[AAR*20]		×			DL						×	×		×		×							3D positions
[CHW*20]		×			DL		×	×									×						DCT
[CPAM20]		×			DL							×				×							3D positions
[CSY20]		×			DL	×	×	×						×	×						×		$so(3)$
[LCZ*20]	</>	×			DL		×	×						×	×	×							$so(3)$
[LKS*20]	</>	×			DL		×	×						×	×								3D positions
[MLS20]	</>	×			DL	×		×				×			×								DCT
[PFAG20]		×			DL			×		×						×							Quaternions
[ZPK20]		×			DL		×	×						×									Unknown
[BGG*20]	</>	×			DL		×	×							×					×			DCT
[CSK*21]		×			DL	×	×	×						×							×		$so(3)$
[LCC*21]		×			DL		×	×			×			×	×	×							$so(3)$
[YKL21]		×			DL		×					×				×							Ad hoc
[PGA18]	</>	×	×		DL			×		×						×							Quaternions
[XLM19]		×	×		DL			×								×							Unknown
[GSAH17]			×		DL			×		×						×							Unknown
[TMLZ18]			×		DL			×								×							$so(3)$
[ZLX*18]	</>		×		DL		×									×							3D positions
[GMK*19]	</>		×		DL			×								×							$so(3)$
[HGM19]	</>		×		DL			×						×							×		3D positions
[KGB19]	</>		×		DL		×	×								×		×			×		$so(3)$
[LWJ*19]	</>		×		DL			×				×				×							$se(3)$
[WHSZ19]			×		DL		×		×	×		×				×							3D positions
[YKK*19]			×		DL							×	×										Unknown
[CGM*20]			×		DL							×		×			×			×			3D positions
[GWE*20]			×		DL							×				×		×		×			Quaternions
[KAS*20]			×		DL					×				×				×					3D positions
[XXN*20]			×		DL		×									×					×		Euler Angles
[HHS*17]			×	×	DL							×		×		×				×			3D positions
[HYNP20]	</>		×	×	DL			×				×				×					×		Quaternions
[WCX21]			×	×	DL		×					×				×					×		Unknown
[THR06]	</>			×	DL		×					×							×				$so(3)$
[TH09]				×	DL		×					×							×				$so(3)$
[CM11]				×	DL		×												×				$so(3)$
[ALP15]				×	DL							×							×				$so(3)$
[HSK16]	</>			×	DL					×				×				×					3D positions
[TCHG17]				×	DL						×	×				×							2D positions
[BKL18]	</>			×	DL			×			×					×					×		3D positions
[YRV*18]	</>			×	DL			×				×				×				×			3D positions
[DHS*19]				×	DL		×		×			×		×						×			3D positions
[YLX*19]				×	DL						×	×			×						×		Unknown
[ASS*20]	</>			×	DL		×	×								×				×			Quaternions
[HAB20]	</>			×	DL		×		×	×		×				×						×	3D positions
[WACD20]	</>			×	DL							×				×		×			×		Euler Angles
[WYZ*20]	</>			×	DL			×			×					×					×		Unknown
[YK20]	</>			×	DL			×				×				×				×			3D positions

This article has been published in COMPUTER GRAPHICS *Forum* (CGF). https://doi.org/10.1111/cgf.14426

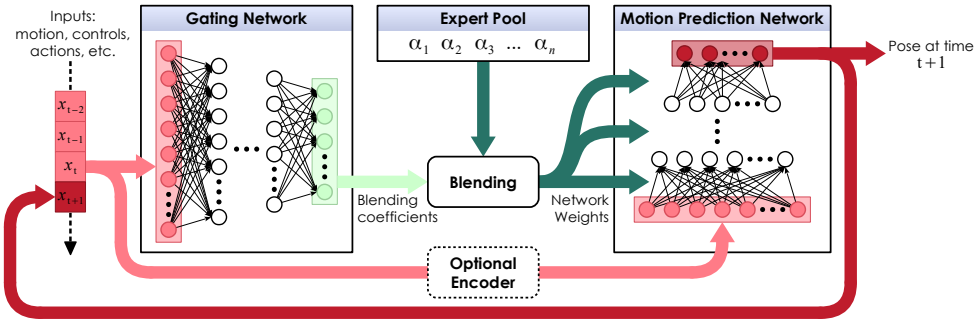

Figure 9: *Overall idea of the mixture-of-experts scheme used in [ZSKS18; SZKS19; SZKZ20; LZZL20]: a gating network first computes blending weights for a number of expert networks, based on a number of features extracted from the current frame x_t. Each expert specialises in a particular movement. They are then blended together to dynamically compute the weights of a motion prediction network, which outputs information relevant to the next frame x_{t+1} including the pose of the animated character. This output is typically fed back into the network (autoregression) at time $t+1$.*

4. Character Control

Controlling character motions that react naturally to user inputs, while accounting for environment constraints as well as biomechanical limitations, is another challenge involved in creating believable virtual characters. In this section, we explore this topic through three main axes: how to make characters move and interact with the virtual environment using kinematic (Section 4.1), physical (Section 4.2) or biomechanical (Section 4.3) control. Finally, we summarise the methods presented in this section in Table 3.

4.1. Kinematics-based

Kinematic approaches typically produce motions as joint angles, based on a set of motion examples and high-level controls (e.g. user inputs, interactions with the environment). Because of the requirement of generating motions in an online fashion when controlling characters in video games or other interactive applications, RNNs and other autoregressive models are often considered to be more appropriate than CNNs, as the future pose is predicted from the previous motion as well as a control signal. For instance, Lee et al. [LLL18] used a four-layer LSTM model for controlling characters playing basketball and tennis. However, as mentioned in Section 3.2, such approaches often tend to fail in the long run, as errors in the prediction are fed back into the input and accumulate, eventually either converging to an average pose or introducing high frequency artifacts. According to Starke et al. [SZKS19], these models also often suffer from low responsiveness due to the large variation of the memory state in the case of interactive character control, as the internal memory state is high dimensional.

To overcome these limitations, Holden et al. [HKS17] proposed the use of a specialised architecture called *Phase-Functioned Neural Network* (PFNN), which provides the phase variable to represent the progression of the motion. In their seminal work, the phase is defined based on alternating foot contacts, and used to generate the weights of the regression network at each frame. A trade-off between compactness and runtime speed can then be achieved by precomputing the phase-function for a number of fixed intervals, then interpolating the precomputed elements at runtime. One major limitation of PFNN is that phase functions need to be manually defined, which can be in some cases extremely complex [ZSKS18; SZKZ20]. Zhang et al. [ZSKS18] therefore proposed to rely on a mixture-of-experts scheme to dynamically compute the weights of a motion prediction network (see Figure 9 for an illustration of the general concept). In their architecture, a gating network first computes blending weights for a number of expert networks, each specialising in a particular movement. This approach was first demonstrated for creating complex quadruped character controllers and then extended by Starke et al. [SZKS19] to compute goal-directed series of motions and transitions, while potentially interacting with the environment. The idea was pushed one step further through the use of local motion phases [SZKZ20], which are defined based on how each body part contacts external objects. Unlike previous approaches where different actions are considered to be synchronised by a single global phase variable, their approach describes each motion by a set of multiple independent and local phases for each bone. It then enables neural networks to learn asynchronous movements of each bone, as well as its interaction with external elements of the virtual environment. While the previous approaches relied on autoregressive DNNs to generate controlled character motions, Ling et al. [LZCvdP20] demonstrated that a VAE using a similar mixture-of-experts scheme is also viable to produce stable high-quality human motions, while being usable in a DRL context to produce goal-directed motions.

Simultaneously, a few approaches explored the creation of controllable expressive human motions from high-level semantic factors. For instance, Alemi and Pasquier [AP17] trained a FCRBM on a dataset of motion capture data containing movements from different subjects, expressions, and trajectories. This model can then be used to generate modulated walking movements in real time. Mason et al. [MSZ*18] explored a similar question from the perspective of generating characters moving in different styles when there is little data available for a new style and proposed a few-shot learning approach. The goal of few-shot learning is to learn (part of) a model able to generalise out of a single or very few examples. In their work, Mason et al. [MSZ*18] adapted a pre-trained

This article has been published in COMPUTER GRAPHICS *Forum* (CGF). https://doi.org/10.1111/cgf.14426

PFNN (modelling style-independent components of the motions), coupled with a set of residual adapters (modelling style-dependent components) learned separately for each new style.

While some of the approaches mentioned above have been demonstrated to be compatible with multi-character control, such character interactions are typically handled by directly including information about the other characters' relative positions [SZKZ20], or indirectly through information about objects both characters are interacting with and the action to be performed [LLL18; SZKZ20]. Wang et al. [WCX17] also proposed to generate character interactions based on the history motion data of both characters, relying on a variant of the ERD architecture to improve animation stability, where multiple LSTM layers constitute the recurrent network.

Finally, despite the impressive advances made by the aforementioned methods, traditional animation approaches are still commonly used in animation pipelines to control human characters because of the quality of the motions produced. However, novel approaches, such as *Learned Motion Matching* [HKPP20], have recently begun to be explored with the goal of breaking down and replacing individual components of animation algorithms by individual specialised neural networks. They balance the advantages of more traditional approaches with the scalability of neural network based models.

4.2. Physics-based

Physics-based approaches generate animations in agreement with physical laws. All but a few methods in this section rely on multibody simulations for physical coherence. DL has also been combined with optimisation-based motion control to generate physically coherent animations. Early works such as Grzeszczuk et al. [GTH98] or Peng et al. [PBvdP16] pioneered the use of deep learning in physically realistic animation. While these were limited to animals, humans were quickly considered as well.

Multibody Simulations. The actuators of a multibody system are typically driven by a DNN trained by *Reinforcement Learning* (RL), where human characters commonly have between 22 and 62 DOFs. In a nutshell, *Reinforcement Learning* (RL) is a framework to learn to control a system by maximising a reward signal. Therefore designing the reward is an important aspect of RL algorithms. The learned model (i.e. the agent) takes as inputs observations (i.e. the states) from the system and outputs actions that impact the system. In the animation context, the states are often a combination of proprioceptive information (e.g. positions, angles, velocities and angular velocities of the joints and the root of the body, end-effector contacts), the phase of the gait, as well as external information (e.g. terrain height, interacting object information) and task information (e.g. path to follow, direction of the goal). Similarly, the reward is often a combination of physics-based penalties (e.g. losing balance, using more energy to produce a motion) and task-specific rewards (e.g. matching a reference motion, satisfying a desired velocity for locomotion). Typically the actions are used to control the character and consist of either torques applied at the joints of the multibody system or target angles for said joints that are converted to torques by *Proportional-Derivative* (PD) controllers. The dynamics of the multibody system is then computed by a physical simulator (e.g. *Bullet* or *MuJoCo*), as illustrated in Figure 10.

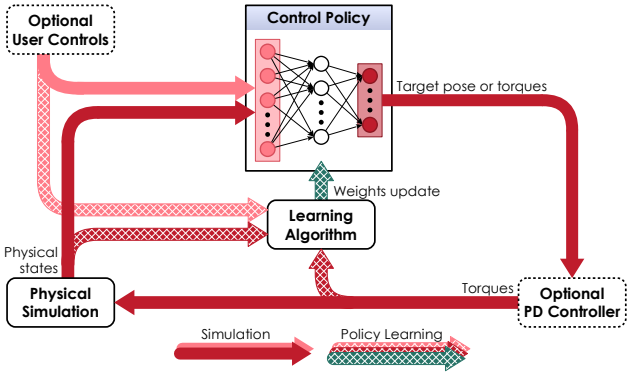

Figure 10: *Illustration of the typical architecture for animation using multi-body simulations. A* Deep Neural Network *(DNN) takes as input information about the state of the character, its surroundings and optionally user inputs such as the desired direction. It then generates either joint torques or target poses fed to a* Proportional-Derivative *(PD) controller. Due to the feedback loop, the model is trained by* Reinforcement Learning *(RL).*

Training a DRL model is different from supervised DL. Rather than using a labeled dataset, the model is typically trained by interacting with the environment and improves at the same time. The most popular DRL algorithm for animation is *Proximal Policy Optimisation* (PPO) [SWD*17]. Common practices to improve training include imitating reference motion capture data; using two controllers: a low-level one for locomotion or low-level actions and a higher one for long term goals; enforcing symmetry of the motion; using a curriculum, that is, learning increasingly difficult tasks; clever state initialisation; early termination based on the lack of end-effector contacts or low rewards received.

Optimisation-based Motion Control. These methods first compute a set of constraints (e.g. footstep positions) based on the environment and a simplified model of the character dynamics (e.g. an inverted pendulum model). The animation is then generated by solving a constrained optimisation problem combining these trajectory and character-level constraints. DL is used here to accelerate and approximate one or several computationally demanding steps.

4.2.1. Character Locomotion

Locomotion was the first task addressed by multibody simulations and DRL. Peng and van de Panne [PvdP17] first compared different action spaces for planar locomotion of bipeds and animals: torques, target joint angles, target joint angle velocities and muscle-activations. They observed that using target joint angles as action space performs well and is the most robust in all cases. It is also the fastest to train in 5 out of 7 cases. Biped locomotion was then quickly considered in 3D. Several works have developed two-level controllers. For instance, Peng et al. [PBYV17] used a high-level controller to output future footsteps, while a low-level controller

This article has been published in COMPUTER GRAPHICS *Forum* (CGF). https://doi.org/10.1111/cgf.14426

trained to imitate reference motions is used to actuate the character. Merel et al. [MTT*17] also constructed low-level controllers that imitate reference motion clips and control actuation. However, multiple low-level controllers are trained using generative adversarial imitation learning. Each controller is trained on a specific skill such as walking, turning, running, and the high-level controller selects the most appropriate instance. Similarly, Merel et al. [MAP*19] trained a character that only perceives its environment through egocentric vision, where a LSTM-based vision-driven high-level controller learns to select among a number of low-level controllers that have learned to imitate short motion clips. Low-level policies can also be combined [PCZ*19] by multiplying their distribution over actions and exponentially weighting them by a weight computed by another neural network. Bergamin et al. [BCHF19] proposed to select short motion capture clips using a motion matching approach. Target angles are then extracted from the selected clip and a subset of these angles are corrected by a DRL model to stabilise the resulting motion. Chentanez et al. [CMM*18] used a similar approach to reproduce a reference clip provided as input, i.e. without including a mechanism for automatic clip selection.

Improving physical character locomotion has also been explored through the use of curricula, training scenarii where the agent is requested to perform a sequence of typically increasingly difficult tasks. For instance, building a curriculum of environments [HTS*17] can help to train an agent that is able to move forward and run, jump, crouch and turn as needed. Xie et al. [XLKvdP20] also compared four environment generating curricula to learn to walk over stepping stones. The resulting policy can then be used to walk over difficult terrain where footstep locations are given. The state includes the height of the pelvis with respect to the lowest foot and the reward includes incentives for placing the foot at the center of a stone and progressing towards the next one. Curricula can also be constructed by assisting the character. Yu et al. [YTL18] trained a model to generate symmetric and low-energy motion to create more realistic motion without using motion capture data. Symmetry is encouraged by an additional loss function. The curriculum is automatically created by including and reducing forces that help with forward motion and left-right balance, leading to different gaits emerging for different target velocities. Abdolhosseini et al. [ALX*19] also addressed learning symmetric gaits by either duplicating observed data by symmetry or by enforcing symmetry in the network. As the approaches discussed so far typically train one model for each morphology, Won and Lee [WL19] developed a parametric DRL controller allowing the shape of the character to be modified during locomotion. At the heart of the training algorithm is an adaptive body shape sampling mechanism that progressively restricts the agent to body shapes it has not mastered yet.

Other types of approaches have also been explored in the community. Rajamäki and Hämäläinen [RH17; RH19] developed a sampling-based approach for locomotion: a *Monte Carlo Tree Search* explores possible actions to maximise rewards over multiple future time steps. In the tree, each child node is one time step ahead of its parent. The impact of a sequence of candidate actions is evaluated by physical simulations. The expansion of the tree (that is, the trajectories explored) is driven in part by GAN models. After a set budget, the most promising action is selected for the next time step. Babadi et al. [BNH19] later leveraged Rajamäki and Hämäläinen's approach to train a DRL network that directly predicts the next action, without looking ahead, as most models in this section do. The Monte Carlo Tree Search approach is used to quickly produce adequate reference motions. Plausible states are extracted from these motions and serve as initial states during training of a DRL model. The authors highlighted a significant reduction in training time.

A few approaches also explored the use of DL-based optimisation techniques to physically animate walking characters. Kwon et al. [KLvdP20] first computed rough successive center of mass and footstep positions using an inverted pendulum on a cart model, refined these estimates and added contact forces, timings and durations using a centroidal dynamics model, then modified these estimates to take external forces into account and finally generated a full-body motion using an IK solver. A DNN was trained to approximate these last two steps for real time performance. Mordatch et al. [MLA*15] trained an RNN model to predict the next pose of a character to generate physically realistic trajectories. Their model was not trained to reproduce the output of a given dynamical system. Instead, they jointly generated physically realistic trajectories and trained the network on these trajectories, which were generated by solving an optimisation problem balancing physical realism (equations of motion, non-penetration, and force complementarity) and the stability of the neural network.

4.2.2. Other Motions

More complex types of motion have also been considered in addition to or instead of locomotion. Peng et al. [PALvdP18] trained DRL models to imitate motion clips. Their approach is applied to locomotion and various acrobatics and martial arts skills. The reward of the training algorithm includes both an imitation term and a task specific one. The model is trained incrementally, starting by the end of the motion. Three approaches are proposed to sequence skills: 1) using the closest clip to compute the imitation reward, 2) using a skill selection vector and 3) learning one policy per skill and using at every time step the policy expected to produce the highest rewards based on the current state. Peng et al. [PKM*18] developed a similar approach based on videos rather than motion clips. For each video clip, 2D and 3D poses are extracted and a 3D reference pose trajectory is reconstructed by optimising in the latent space of an autoencoder. Furthermore, the dataset is augmented by rotations and a curriculum is automatically constructed by another agent trained to propose initial states. Ma et al. [MYT*21] proposed to impose space-time bounds around samples of a trajectory rather than an imitation reward to learn from reference motions. These bounds only allow small deviations in joint angles and in the positions of the center of mass and end effectors. Learning is possible using only a binary reward tied to the respect of these bounds and early termination. Furthermore, to learn different styles, they proposed different additional rewards to steer the model towards animations resulting in a different energy level or convex hull volume. They also sampled initial states to favour states from which the current model achieves relatively lower rewards. Ranganath et al. [RXKZ19] compressed the action space using principal or independent component analysis, reducing the size of the output of the DRL model. Their method is otherwise similar to [PALvdP18].

This article has been published in COMPUTER GRAPHICS *Forum* (CGF). https://doi.org/10.1111/cgf.14426

While some approaches learn a different model for each motion clip, Merel et al. [MHG*19] trained a single controller able to execute many skills. They trained a VAE embedding current and future states from motion capture clips and decoding the action from the embedding and the current state. They then trained a high-level controller to set the value of the latent variable based on the task. On a side note, Wang et al. [WMR*17] learned a similar embedding (although based on previous rather than future states), but used it only for reproducing a given clip, not to automatically generate new animations. Won et al. [WGH20] took as input a motion controller and trained a controller for a physically simulated character that can reproduce all motions of the input controller. At runtime, the trained controller is used to track such a motion. They used a two-level model by clustering generated motions, training one model by cluster and combining them in a mixture. They demonstrated their results on walking, running, acting and performing various dance styles. They also compared additive to multiplicative combination of different reward terms and chose the latter, arguing that it forces all reward terms to be high and that they obtained better results with complex motions such as dancing.

4.2.3. Interactions

DRL has also been used to animate characters interacting with other characters and complex objects. On a side note, some methods already described above have been applied to simple interactions such as pushing an object [PBYV17; PKM*18; PCZ*19]. Liu and Hodgins [LH17] tackled animating a character walking on a ball, balancing on a bongo board or skateboarding by generating control fragments from motion capture data and training a controller to select a generated control fragment at runtime. A control fragment is a short (0.1s) segment of motion capture data and an associated linear control policy. The same authors [LH18] later addressed basketball dribbling. One controller is first learned for the body and legs. Then, a second controller is trained for the arms. Both are based on control fragments extracted from motion capture data. For these two controllers, the actions are corrected by a linear model and a DRL model, respectively. Transitions between control fragments are also learned. Basketball has also been addressed by Park et al. [PRL*19], together with obstacle racing, chicken hopping, fighting with other characters and various other skills. A DRL controller is trained to correct target poses generated by a RNN and then fed to PD controllers. The RNN uses multi-objective character control similar to [LLL18]. Pushing character interaction much further, Haworth et al. [HBM*20] used multi-agent DRL to train a two-level model for crowd animation inspired by Peng et al. [PBYV17], where the low-level controls one character based on two future footstep targets provided by the high-level controller. The latter is specific for each character whereas the former is shared. The input of the high-level controller includes a velocity field of nearby objects and agents.

Merel et al. [MTA*20] extended their previous work [MAP*19] to object manipulation tasks from egocentric vision. It involves a hierarchical controller based on a latent space (similar to [MHG*19]), learning tasks step by step and task variations. Notably, the object is not part of the state. The character can move objects between shelves or catch and throw balls.

Interactions with objects of the scene have also been explored with respect to clothes, in order to animate an upper human body dressing up [CYT*18]. The key ideas include using haptic information in the state, dividing the task into subtasks, using the end states of one subtask as initial states for training the next one and including cloth deformation in the reward.

4.3. Biomechanics-based

Concurrently to joint-actuated approaches for motion control, neural networks have also been explored in the context of musculoskeletal simulation. The general idea is that the complexity of traditional musculoskeletal models usually leads to costly simulations, and that neural network controllers can provide an efficient solution by learning biomimicking controls using data simulated offline with a detailed biomechanical model, to efficiently output optimal muscle activations at runtime.

Lee and Terzopoulos [LT06] originally applied this idea to design novel neuromuscular control models of the human head-neck system (comprising of 7 cervical vertebrae and 72 neck muscles) learned in supervised frameworks, first using shallow neural networks generating both pose and tone control signals for the movements of the head. Then, Nakada and Terzopoulos [NT15] improved head stability in a larger range of situations using deep stacked denoising autoencoders. Nakada et al. [NZC*18] further generalised the concept to full-body human biomechanical simulation (193 bone model actuated by a total of 823 muscles) by designing a sensorimotor control system made of 20 fully connected DNNs. In their framework, 5 DNNs per retina are responsible for extracting visual information required to direct eye/head, arm and leg movements, while 10 additional DNNs are responsible for the neuromuscular control of the 216 neck muscles, of the 29 muscles of each arm, and of the 39 muscles of each leg. More specifically, the limbs and head are driven by one voluntary motor DNN each, responsible for generating efferent activation signals for the corresponding muscles to execute realistic controlled movements, and one reflex motor DNN each, responsible for computing muscle activation adjustments due to low-level muscle dynamics. Lee et al. [LPLL19] also explored full-body musculoskeletal simulation using DRL, with a two-level approach presented to learn simultaneously trajectory mimicking and muscle coordination.

Finally, motion realism brought by such musculoskeletal approaches generally comes at the expense of complex modelling and costly simulations, which might explain to some extent why joint-actuation approaches are still most commonly used. To bridge the gap between both worlds, and "generate human-like motion comparable to muscle-actuation models, while retaining the benefit of simple modelling and fast computation offered by joint-actuation models", Jiang et al. [JVDL19] proposed an approach to formulate the optimal control problem in the joint-actuation space while having an equivalent solution to the same problem in the muscle-actuation space. Both the metabolic energy function and state-dependent torque limits expressed in the joint-actuation space are approximated using fully connected DNNs and learned in a DL framework, and then used in conjunction with a policy learning algorithm built upon previous work [YTL18] (See section 4.2).

This article has been published in COMPUTER GRAPHICS *Forum* (CGF). https://doi.org/10.1111/cgf.14426

Table 3: *Summary of the methods presented in Section 4. Miscellaneous data includes hand-crafted, synthetic, proprietary, unspecified or other public datasets.*

Reference	Code	Section			Framework	Dataset			Architecture					Model & Simulation					
		4.1 Kinematics-based	4.2 Physics-based	4.3 Biomechanics-based		CMU [Uni03]	Holden et al. [HSK16]	Miscellaneous	Fully Connected	Recurrent	RBM	Variational Autoencoder	Adversarial	Simulator	Dimensionality	Humanoid(s)	Other biped(s)	Monoped(s)	Quadruped(s)
[AP17]		×			DL			×			×				3D	×			
[HKS17]	</>	×			DL			×	×						3D	×			
[WCX17]		×			DL	×		×		×					3D	×			
[LLL18]		×			DL			×		×					3D	×			
[MSZ*18]	</>	×			DL	×			×						3D	×			
[ZSKS18]	</>	×			DL			×	×						3D				×
[SZKS19]	</>	×			DL			×	×						3D	×			
[HKPP20]		×			DL			×	×						3D	×			
[LZCvdP20]		×			DL & DRL			×	×			×			3D	×			
[SZKZ20]		×			DL			×	×				×		3D	×			
[MLA*15]			×		DL			×		×				MuJoCo	3D	×			×
[HTS*17]			×		DRL					×				MuJoCo	3D	×	×		×
[LH17]			×		DRL				×					ODE	3D	×			
[MTT*17]			×		DRL	×			×				×	MuJoCo	3D	×			
[PBYV17]	</>		×		DRL			×	×					Bullet	3D		×		
[PvdP17]			×		DRL			×	×					Unknown	2D		×		×
[RH17]	</>		×		DRL				×					ODE	3D	×		×	×
[WMR*17]			×		DRL	×				×		×	×	MuJoCo	3D	×	×		
[CMM*18]			×		DRL		×		×					MuJoCo	3D	×			
[CYT*18]			×		DRL				×					DART	3D	×			
[LH18]			×		DRL			×	×					ODE	3D	×			
[PALvdP18]	</>		×		DRL	×			×					Bullet	3D	×	×		×
[PKM*18]	</>		×		DRL			×	×					Bullet	3D	×			
[YTL18]	</>		×		DRL				×					DART	3D	×			×
[ALX*19]	</>		×		DRL			×	×					Bullet	3D	×	×		
[BCHF19]			×		DRL			×	×					Bullet	3D	×			
[BNH19]	</>		×		DRL			×	×					ODE	3D	×	×		×
[MAP*19]			×		DRL			×		×				MuJoCo	3D	×			
[MHG*19]			×		DRL	×			×			×		MuJoCo	3D	×			
[PCZ*19]			×		DRL			×	×					Bullet	3D	×	×		×
[PRL*19]	</>		×		DL & DRL	×		×	×					DART	3D	×			
[RH19]	</>		×		DRL				×					ODE	3D	×		×	×
[RXKZ19]			×		DRL	×		×	×					Unknown	3D	×			
[WL19]			×		DRL				×					DART	3D	×	×		×
[HBM*20]			×		DRL			×	×					Unknown	3D	×			
[KLvdP20]	</>		×		DL			×	×					Unknown	3D	×	×	×	×
[MTA*20]	</>		×		DRL			×		×				MuJoCo	3D	×			
[WGH20]	</>		×		DL & DRL	×			×					Bullet	3D	×			
[XLKvdP20]	</>		×		DRL				×					Bullet	3D	×	×		
[MYT*21]			×		DRL	×			×					Bullet	3D	×			
[LT06]				×	DL	×			×					Unknown	3D	×			
[NT15]				×	DL			×	×					Unknown	3D	×			
[NZC*18]				×	DL			×	×					Unknown	3D	×			
[JVDL19]	</>			×	DL & DRL			×	×					OpenSim	3D	×			
[LPLL19]	</>			×	DL & DRL			×	×					OpenSim	3D	×			

5. Motion Editing

So far we looked at how motion data can be synthesised, either for predictive or generative purposes, and how to build frameworks for interactively controlling virtual characters. Besides these applications, another important area of character animation is motion editing, which diversifies the creative capabilities of artists. In this section, we focus on DL-based approaches for motion editing problems divided into three topics: motion cleaning (Section 5.1), motion retargeting (Section 5.2) and style transfer (Section 5.3). Finally, we summarise the methods presented in this section in Table 4.

5.1. Cleaning

As mentioned by several authors, projection to and inverse projection from learnt manifolds of human motion can be used to clean motion data, e.g. to perform operations such as denoising, fixing corrupted information or filling in missing motion sequences. Such problems have for instance been explored using RBMs [WN15], convolutional autoencoders [HSKJ15], temporal autoencoders [BBKK17; LAT21], spatio-temporal RNNs [WHSZ19], sequential RNNs [JL20], BiLSTMs [LZZ*19], as well as RNN-based GANs [WCX21]. While most approaches typically clean motion data through direct projection and inverse projection (e.g. [WN15; HSKJ15]), then fix residual errors as a post-process, it is also possible to include additional constraints during training. For instance, it is possible to enforce bone length constraints and smoothness by including specific loss functions [LZZ*19; LZZL20], or to include an additional perceptual loss measuring the difference in high-level features extracted by a pre-trained perceptual autoencoder [LZZL20], which improves overall visual quality at the cost of a slight in-

This article has been published in COMPUTER GRAPHICS *Forum* (CGF). https://doi.org/10.1111/cgf.14426

crease in reproduction error. Lohit et al. [LAT21] also proposed to optimise the latent representation to minimise the error between the reconstructed sequence and the correct information of the input sequence, which they applied to filling missing joint trajectories. While most approaches focus on cleaning directly kinematics skeletal data, Holden [Hol18] proposed an approach producing joint transforms directly from raw marker data, in a way which is robust to errors in the input data. This approach is based on learning a deep denoising feedforward neural network using marker locations synthetically reconstructed from motion capture data, where the marker data is corrupted in terms of occlusions and positional shifts.

Several authors also proposed to train neural networks to automatically detect ground contact events (i.e. whether the feet are in contact with the floor or not), in order to prevent footsliding artefacts. These approaches typically rely on motion data augmented with foot contact information to output foot contact probabilities. To this end, foot contact can be either manually annotated [SCNW19], or automatically computed using different heuristics, usually based on empirical positional [YKL21] or velocity [ZYC*20] thresholds. Using such data, different architectures have been proposed to automatically estimate foot contacts, such as relying on a fully connected neural network [SCNW19], on a temporal CNN with residual connections [ZYC*20], or on an RNN with GRUs followed by linear layers and ReLU activation [YKL21]. At runtime, the network estimates the ground contact information of each pose, which are then often used within an IK framework to remove footskate artifacts [SCNW19; YKL21]. This information can also be included as an additional zero velocity loss within a state-of-the-art method for pose and shape estimation [ZYC*20], or combined with other constraints into a physics-based optimisation [SGXT20]. Shi et al. [SAA*20] also identified the importance of foot contacts to mitigate footskating artifacts, and proposed a network predicting simultaneously joint rotations, global root positions, as well as foot contact labels from estimated 2D joint positions, which are then fed into an integrated FK layer that outputs 3D positions.

5.2. Retargeting

Retargeting [Gle98] refers to the task of transferring the movements of a source character in a skeletal animation sequence to a target character with a different morphology, i.e. to a skeleton with different bone lengths and possibly a different topology. For clarity, and consistently with the literature, in the remainder of this section the source character will be referred to as A and the target character as B. As pointed out by Aberman et al. [ALL*20], there is no formal specification of the task. The purpose at large is to abstract out the dynamics of the source sequence and to reproduce it on a character whose body proportions differ. Effectively, the sought goal is to synthesise a retargeted sequence for the new morphology whose motion mimics the source while remaining visually plausible and natural. Since it is difficult in practice to obtain ground-truth pairs of (source, target) sequences with exactly the same motion, most learning approaches to retargeting, and all the schemes surveyed in this section, rely on unpaired training data without motion correspondences across characters.

In the seminal work of Villegas et al. [VYCL18], the retargeting network is built around two RNNs. An encoder RNN captures the motion context of the source sequence in its hidden state and forwards it to a decoder RNN that outputs each frame of the retargeted sequence. A reference pose of the target skeleton is provided to the decoder. Besides regularisation loss terms, network training is driven by an adversarial loss and a cycle consistency loss. The adversarial loss attempts to minimise discrepancies between the joint velocities of ground truth (true) and retargeted (fake) sequences. The cycle consistency loss ensures that a motion sequence of character A that is retargeted to B and then back to A remains as close as possible to the original.

Lim et al. [LCC19] and Kim et al. [KPKH20] reported that the above approach tends to generate unrealistic motion. To mitigate this issue, Lim et al. [LCC19] retargeted the motion of the root joint separately from the poses at each time step, and combined the results to construct the output sequence. The retargeted poses are computed as joint rotations, represented as quaternions, to be applied to a reference pose of the target skeleton. The cycle consistency loss of Villegas et al. [VYCL18] is replaced by a reconstruction loss associated to self-retargeting to the same character. Kim et al. [KPKH20] argued that CNNs are better suited than RNNs for retargeting because they can more accurately capture the short-term motion dependencies that mostly condition the performance of the task. Accordingly, their scheme relies on a purely convolutional network with temporally dilated convolutions that retargets whole motion sequences in one batch.

Aberman et al. [ALL*20] extended the scope of retargeting to skeletons with different topologies, subject to the condition that all considered topologies are homeomorphic. This implies that they can all be reduced to a common *primal skeleton* by merging pairs of adjacent bones. Their representation of skeletal motion based on *armatures* separates the temporally invariant bone offset vectors that define the character morphology from the time-varying bone rotations at each joint that capture the motion dynamics (see Figure 3). These two components are processed in distinct parallel branches of the retargeting network. Importantly, a motion sequence is modelled as a graph whose edges correspond to armatures. This provides a principled formalism for processing motion data sampled on the skeletal graph. The retargeting network includes three types of modules: space-time graph-convolutional operators acting on spatio-temporal joint neighbourhoods, graph pooling and graph unpooling operators. Graph pooling merges features of two adjacent edges (armatures) into a single feature. Each graph unpooling operator is designed as the inverse of a graph pooling operator, splitting one edge into two adjacent edges whose features are copied from the original feature. An autoencoder, composed of an encoder followed by a decoder, is learnt for every skeletal structure represented in the training dataset. An encoder fed with motion data for character A generates embeddings of the static bone offset and dynamic joint rotation components for this motion, expressed in the common primal skeleton topology. A decoder for character B maps these embeddings to the retargeted motion for this character. Retargeting is achieved by composing the encoder for the source character with the decoder for the target character.

This article has been published in COMPUTER GRAPHICS *Forum* (CGF). https://doi.org/10.1111/cgf.14426

Table 4: *Summary of the methods presented in Section 5. Miscellaneous data includes hand-crafted, synthetic, proprietary, unspecified or other public datasets.*

Reference	Code	Section			Framework	Dataset							Architecture							Representation
		5.1 Cleaning	5.2 Retargeting	5.3 Style Transfer		CMU [Uni03]	Human3.6M [IPOS14]	HDM05 [MRC*07]	Holden et al. [HSK16]	Mixamo [Ado]	NTU RGB+D [SLNW16]	Miscellaneous	Fully Connected	Convolutional	Recurrent	Autoencoder	RBM	Variational Autoencoder	Adversarial	
[HSKJ15]		×			DL	×								×		×				3D positions
[WN15]		×			DL	×										×	×			Unknown
[BBKK17]		×			DL	×	×						×							$\mathfrak{so}(3)$
[Hol18]		×			DL	×						×	×							Euler Angles
[LZZ*19]		×			DL	×									×	×				3D positions
[WHSZ19]		×			DL	×		×	×			×			×					3D positions
[JL20]	</>	×			DL		×								×	×			×	$\mathfrak{so}(3)$
[LZZL20]		×			DL	×						×			×	×				3D positions
[SAA*20]	</>	×			DL	×	×							×					×	3D positions & Quaternions
[SGXT20]		×			DL		×					×		×						2D positions
[ZYC*20]	</>	×			DL		×					×		×						2D positions
[LAT21]		×			DL			×			×			×		×			×	2D positions
[WCX21]		×			DL	×						×			×				×	Unknown
[YKL21]		×			DL	×						×			×					Ad hoc
[SCNW19]		×		×	DL							×	×							3D positions
[VYCL18]	</>		×		DL		×			×					×				×	Quaternions
[LCC19]	</>		×		DL					×				×		×			×	Quaternions
[ALL*20]	</>		×		DL					×				×		×			×	3D positions & Quaternions
[KPKH20]	</>		×		DL		×			×				×					×	Quaternions
[HSK16]	</>			×	DL				×					×		×				3D positions
[HHKK17]				×	DL							×		×		×				3D positions
[WCAD18]				×	DL							×			×	×			×	Unknown
[AWL*20]	</>			×	DL							×		×		×			×	3D positions & Quaternions
[DAS*20]				×	DL							×		×					×	Quaternions
[WACD20]	</>			×	DL							×			×	×			×	Euler Angles

5.3. Style Transfer

Neural Style Transfer refers to algorithms manipulating data such as images, videos or human animations with DNNs to make the stylistic components look like another data sample. Gatys et al. [GEB16] first introduced a method to perform neural style transfer on images, using the Gram matrix of the deep features as the artistic style information of an image. In human animation, style transfer aims at transferring the style from one motion sequence to another whose content is retained, called hereafter style and content motion sequences, respectively. For example, we might want to edit a particular motion by affecting the state of mind of the character (e.g. enthusiastic, sad, angry) while preserving the performed action, e.g. locomotion from point A to point B.

In the context of DL-based skeletal animation, Holden et al. [HSK16] pioneered human motion style transfer using their deterministic generation model (see Section 3.3). In this framework, different types of constraints can be applied on the generated motion, such as trajectory, bone lengths or joint positions, solving a constrained optimisation problem in a learnt human motion manifold. Gradients are back-propagated through an autoencoder representing the manifold to optimise latent representations. Style transfer constitutes a particular case, where both joint positions and style are constrained with respect to the content and style motion sequences, respectively. Moreover, Holden et al. [HSK16] followed prior work in image style transfer [GEB16] by using the Gram matrix in the latent representation as a style similarity measure and thus do not require style annotations. They further improved this approach by training a feedforward network to perform style transfer thousands of times faster [HHKK17]. Gradients are back-propagated to train the feedforward network instead of optimising the latent representation.

Alternatively to the Gram matrix, style annotations can also be used to guide the learning of a style transfer model. Smith et al. [SCNW19] divided the task of style transfer into spatial and temporal style variations networks, both taking as inputs joint positions as well as a one-hot style vector, where the networks are applied consecutively to predict corresponding style variations. However, this approach requires motion data registered with similar poses in different styles. Wang et al. [WCAD18; WACD20] leveraged an LSTM-based Sequential Adversarial Autoencoder whose encoder learns to map motions to separate content and style encodings: two additional discriminators are trained to recover style labels from content and style encodings, respectively. The encoder tries to fool the former and helps the latter in order to free the content encoding from the style information while preserving it in the style encoding. At runtime, both the content and style motion sequences are encoded into separate content and style embeddings. The decoder combines the content embedding of the content motion sequence and the style embedding of the style motion sequence to produce the style transfer output. Following these works, Aberman et al. [AWL*20] also extracted content and style encodings but with two separate encoders. Furthermore, the style encoder learns a common embedding from both 2D and 3D joint positions with a triplet loss, which enables style extraction from videos. The output motion is synthesised from the content encoding while the style

This article has been published in COMPUTER GRAPHICS *Forum* (CGF). https://doi.org/10.1111/cgf.14426

is controlled by the style encoding through temporally invariant *Adaptive Instance Normalisation* (AdaIN). Moreover, a multi-style discriminator assesses the output motion style.

Beyond general-purpose style transfer, Dong et al. [DAS*20] focused on translation from adult to child motions and vice versa. In this particular case, retargeting is not sufficient since it does not capture the natural stylistic differences between adults and children [DAS*20]. This work leverages the capacity of *CycleGAN* [ZPIE17] to learn the mapping between adult and child motion distributions without paired training data, which is critical due to the very limited availability of child data.

6. Discussion

In this state-of-the-art review we explored the recent and promising trends to address challenges in skeleton-based human animation with DL and DRL. After a general overview of pose representations and motion data processing with DNNs, we covered motion synthesis, character control and motion editing based on DL and/or DRL. In this section, we discuss some of the limitations and potential future work directions.

First, a pose/motion representation should characterise the human motion as precisely as possible, while being suitable for optimisation (e.g. avoid the error accumulation problem) and other common applications such as skinning and rigging. However, no simple representation fulfills all these requirements. The success of different approaches [PGA18; PFAG20; ALL*20] suggests that expressing loss functions in the joint positions space is helpful, although the joint orientations are more expressive to represent human poses. To better animate human characters, it is necessary to develop more general pose/motion representation frameworks suitable for learning both temporal and spatial patterns, probably combining the advantages of hierarchical orientations and absolute positions, such as the approach proposed by Aberman et al. [ALL*20] for skeletal convolutions in the case of motion retargeting (see Figure 3). Finally, the spectral domain of human motion remains as of today almost unexplored [MLSL19; MLS20; CHW*20] whereas, e.g. artifacts related to high-frequencies like noise or footskate might be easier to prevent and/or remove in such a domain.

Second, approaches in short-term motion prediction attempt to forecast future motion exactly, with a ground truth being the only solution. Many of them are autoregressive, i.e. (partial) outputs are fed back to the model to predict distant future motion frames. As a result, they suffer from instability and tend to either diverge or converge to a mean pose. This is exacerbated when targeting longer horizons since the problem of deterministic prediction becomes more and more ambiguous. Ill-posedness can be mitigated either by adding contextual information or by modeling and sampling from a distribution of possible future outcomes. Hopefully, both short and long term motion prediction could be achieved by a single powerful stochastic model with a quasi-deterministic short-term behaviour, from which stochasticity could emerge in the long term.

Third, important aspects of generative models in motion synthesis include the quality (e.g. perceptual plausibility, absence of artifacts), the intermodal and intramodal diversity, and the level of representation detail (e.g. ample arm movements versus subtle finger manipulations), but also the ability to control high-level parameters in synthesised motions. Currently, generative models still fail to fully represent the human motion distribution in the considered scope, with all the diversity and modes it is made of. One reason for this is the lack of in-the-wild motion data, mainly because reliable motion capture systems generally require markers to be placed on the subject captured and a dedicated room for a multi-camera setup. Future advances in marker-less motion capture from image/video might help to collect more and more diverse motion data, and hopefully to build strong human motion generative models, maybe directly from image/video. Moreover, generative models would also benefit from motion data with higher skeleton resolution, i.e. from capturing more joints. This would both improve the quality of skinned character animation based on skeletal motion data and enable models to account for more subtle details of human motion, especially those expressed by the hands, the feet and the head. Another promising opening to obtain favorable results with limited amount of in-the-wild data might be self-supervised learning. In such a framework, a representation of the data would first be learnt over in-the-lab data (i.e. the pretext task) then fine-tuned over in-the-wild data (i.e. the downstream task). Although advances in DL allow to model the human motion distribution with increasing diversity and fidelity, general-purpose approaches are most of the time too general and lack flexibility. Indeed, the difficulty to embed hard constraints in DL, e.g. desired precise action(s) or physical correctness, and the tendency of neural networks to be black boxes prevent animators from adjusting model behaviours to specific needs or from interactively editing the generated sequences.

Fourth, unlike with deep generative models used in motion synthesis, interactivity is at the core of character control with models dynamically reacting to user input flows while trying to provide diverse motions that are accurate, physically realistic and performed like real humans would do in a similar situation. On the one hand physics and biomechanics based approaches rely on the laws of physics – often into multibody physical simulations – with actuation models, while on the other hand kinematics-based approaches directly produce character motion without hard physical constraints. As of today, the former provide humanoid controllers that are mostly correct physically speaking since hard constraints are part of the models themselves. Although these controllers e.g. manage locomotion, jumping and specific actions such as lifting, carrying, or pushing objects, they often fail to imitate human naturalness. It appears to us that the approaches closest to providing natural human motion rely more on real-world data besides their physical model, e.g. the framework proposed by Peng et al. [PKM*18] which consists in imitating motions extracted from video using deep pose estimation or the data-driven controller proposed by Bergamin et al. [BCHF19] where the policy network computes corrective offsets added to reference motions. Approaches in kinematics-based character control are exclusively data-driven, which makes it easier to obtain more natural results. However, the lack of physical models/simulation prevents produced animations from being physically correct. In practice, real-world animation pipelines still mostly prefer traditional rather than automated methods due to their unpredictable behaviours and lower quality, although promising advances have been made with DL

This article has been published in COMPUTER GRAPHICS *Forum* (CGF). https://doi.org/10.1111/cgf.14426

and DRL. Finally, we could expect approaches in character control to increasingly rely on (data-driven) deep generative models together with biomechanical or physical models (e.g. [JVDL19; LPLL19; PRL*19; WGH20]) to bring strengths from both types of approaches and converge to diverse, natural, high-quality and physically-plausible controllers. Further advances on the flexibility or the interactivity at runtime are also expected.

Fifth, motion editing has still not been explored very extensively even though promising methods have been published (e.g. [HSKJ15; HSK16; ALL*20; AWL*20; DAS*20]). Yet this topic is important to empower animators in other fields of human animation. A significant contribution in this respect is the motion editing framework proposed by Holden et al. [HSK16], where low-level motion parameters learnt by a motion modelling network are mapped to high-level, human understandable controls by means of a disambiguation network. This latter network is trained separately and its inputs can be fine-tuned to the editing task at hand. To the best of our knowledge, this work has not been followed up. In our opinion, the main remaining challenges are the interpretability and controllability of the method, especially in a professional content production workflow. Interpretability refers to determining some relationship between the latent representation and the physical control, while controllability refers to the explicit control of animations given some latent representation. To the best of our knowledge, no method has achieved these two goals at the moment. Removing or preventing artifacts is also a critical part of motion editing, e.g. for motion data pre-processing or post-processing in animation workflows. As an example, footskate artifacts are widespread but there is still no successful fully automated method to reliably fix this artifact in motion data. In the future we expect a growing number of works in motion editing at large, and maybe even new topics to emerge like retargeting did 20 years ago.

Lastly, most of the studies covered in this survey suffer from a lack of perceptual and user studies, which are crucial. For instance, in motion prediction most works compare their results to others using statistical metrics, such as the mean squared error over predicted joint angles, which are not suited to evaluate how plausible or natural predicted motions would be perceived. Other topics also mostly skip perceptual and user studies although sometimes no ground truth exists, e.g. character control. This is particularly striking in motion generation where ground truth is equivocal and the inception score is mainly used to assess the diversity of synthesised motions. We therefore expect more perceptual and user studies as well as performance metrics involving perceptual cues in the future.

To conclude this survey, we have presented a summary of the advances made over the last few years in human animation based on deep neural networks, either trained using DL or DRL. These approaches are skyrocketing today in the field of animation, and will probably increasingly find their place in real-world applications in the entertainment industry to animate synthetic characters.

Acknowledgements

This work was supported by the European Commission under European Horizon 2020 Programme, grant number 951911 - AI4Media.

Acronyms

BiLSTM	Bidirectional LSTM . . . 8, 21
CNN	Convolutional Neural Network . . . 6-8, 10, 11, 17, 22
CRBM	Conditional RBM . . . 13, 14
DCT	Discrete Cosine Transform . . . 8, 10
DL	Deep Learning . . . 1, 2, 4, 5, 9, 11, 18-20, 23-25
DNN	Deep Neural Network . . . 1, 2, 4-6, 8, 10, 14, 17-20, 22, 24
DOF	Degree of Freedom . . . 2-4, 18
DRL	Deep Reinforcement Learning . . . 1, 2, 4, 10, 17-20, 24, 25
ERD	Encoder-Recurrent-Decoder . . . 8, 9, 11-13, 15, 18
FCRBM	Factored CRBM . . . 13, 14, 17
FK	Forward Kinematics . . . 4, 5, 10, 22
GAN	Generative Adversarial Network . . . 9, 12, 13, 15, 19, 21
GCN	Graph Convolutional Network . . . 6, 7, 9-11
GMM	Gaussian Mixture Model . . . 12
GRU	Gated Recurrent Unit . . . 7-11, 22
HFCRBM	Hierarchical FCRBM . . . 14
IK	Inverse Kinematics . . . 3, 19, 22
LSTM	Long Short-Term Memory . . . 7-9, 13, 15, 17, 18, 23
MCMC	Markov Chain Monte Carlo . . . 13
ML	Machine Learning . . . 2, 9
NLP	Natural Language Processing . . . 1, 8, 10
PD	Proportional-Derivative . . . 18, 20
PFNN	Phase-Functioned Neural Network . . . 17
RBM	Restricted Boltzmann Machine . . . 13, 21
RL	Reinforcement Learning . . . 18
RNN	Recurrent Neural Network . . . 6-15, 17, 19-22
Seq2Seq	Sequence to Sequence . . . 9-11
VAE	Variational Autoencoder . . . 11, 12, 14, 15, 17, 19

References

[AAR*20] ADELI, VIDA, ADELI, EHSAN, REID, IAN, et al. "Socially and Contextually Aware Human Motion and Pose Forecasting". *IEEE Robotics and Automation Letters* 5.4 (July 2020), 6033–6040. DOI: 10.1109/LRA.2020.3010742 8, 11, 16.

[Ado] ADOBE. *Mixamo*. Accessed: 2021-09-16. URL: https://www.mixamo.com 6, 23.

[AII*18] ANDRILUKA, MYKHAYLO, IQBAL, UMAR, INSAFUTDINOV, ELDAR, et al. "PoseTrack: A Benchmark for Human Pose Estimation and Tracking". *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. IEEE Computer Society, June 2018, 5167–5176. DOI: 10.1109/CVPR.2018.00542 6.

[AKH19] AKSAN, EMRE, KAUFMANN, MANUEL, and HILLIGES, OTMAR. "Structured Prediction Helps 3D Human Motion Modelling". *Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. IEEE Computer Society, Oct. 2019, 7143–7152. DOI: 10.1109/ICCV.2019.00724 7, 9, 16.

[ALL*20] ABERMAN, KFIR, LI, PEIZHUO, LISCHINSKI, DANI, et al. "Skeleton-Aware Networks for Deep Motion Retargeting". *ACM Transactions on Graphics* 39.4 (July 2020). DOI: 10.1145/3386569.3392462 4, 5, 7, 8, 22–25.

[ALP15] ALEMI, OMID, LI, WILLIAM, and PASQUIER, PHILIPPE. "Affect-expressive movement generation with factored conditional Restricted Boltzmann Machines". *Proceedings of the IEEE International Conference on Affective Computing and Intelligent Interaction*. IEEE Computer Society, Sept. 2015, 442–448. DOI: 10.1109/ACII.2015.7344608 4, 6, 14, 16.

[ALX*19] ABDOLHOSSEINI, FARZAD, LING, HUNG YU, XIE, ZHAOMING, et al. "On Learning Symmetric Locomotion". *Proceedings of the ACM SIGGRAPH Conference on Motion, Interaction and Games*. Association for Computing Machinery, Oct. 2019. DOI: 10.1145/3359566.3360070 19, 21.

[AP17] ALEMI, OMID and PASQUIER, PHILIPPE. "WalkNet: A Neural-Network-Based Interactive Walking Controller". *Proceedings of the International Conference on Intelligent Virtual Agents*. Springer International Publishing, Aug. 2017, 15–24. DOI: 10.1007/978-3-319-67401-8_2 4, 17, 21.

[AP19] ALEMI, OMID and PASQUIER, PHILIPPE. "Machine Learning for Data-Driven Movement Generation: a Review of the State of the Art". *arXiv e-prints* (Mar. 2019). eprint: 1903.08356 2.

[ASS*20] ALIAKBARIAN, SADEGH, SALEH, FATEMEH SADAT, SALZMANN, MATHIEU, et al. "A Stochastic Conditioning Scheme for Diverse Human Motion Prediction". *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. June 2020, 5223–5232. DOI: 10.1109/CVPR42600.2020.00527 4, 5, 8, 14, 16.

[AWL*20] ABERMAN, KFIR, WENG, YIJIA, LISCHINSKI, DANI, et al. "Unpaired Motion Style Transfer from Video to Animation". *ACM Transactions on Graphics* 39.4 (2020). DOI: 10.1145/3386569.3392469 4, 8, 23, 25.

[BBKK17] BÜTEPAGE, JUDITH, BLACK, MICHAEL J., KRAGIC, DANICA, and KJELLSTRÖM, HEDVIG. "Deep Representation Learning for Human Motion Prediction and Classification". *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. IEEE Computer Society, July 2017, 1591–1599. DOI: 10.1109/CVPR.2017.173 4, 5, 7, 8, 10, 11, 16, 21, 23.

[BBL*17] BRONSTEIN, MICHAEL, BRUNA, JOAN, LECUN, YANN, et al. "Geometric Deep Learning: Going beyond Euclidean data". *IEEE Signal Processing Magazine* 34.4 (Nov. 2017), 18–42. DOI: 10.1109/MSP.2017.2693418 7.

[BCHF19] BERGAMIN, KEVIN, CLAVET, SIMON, HOLDEN, DANIEL, and FORBES, JAMES RICHARD. "DReCon: Data-Driven Responsive Control of Physics-Based Characters". *ACM Transactions on Graphics* 38.6 (Nov. 2019). DOI: 10.1145/3355089.3356536 19, 21, 24.

[BGG*20] BOURACHED, ANTHONY, GRIFFITHS, RYAN-RHYS, GRAY, ROBERT, et al. "Generative Model-Enhanced Human Motion Prediction". *NeurIPS Workshop on Interpretable Inductive Biases and Physically Structured Learning*. Dec. 2020. eprint: 2010.11699 5, 11, 16.

[BKL18] BARSOUM, EMAD, KENDER, JOHN, and LIU, ZICHENG. "HP-GAN: Probabilistic 3D Human Motion Prediction via GAN". *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*. IEEE Computer Society, June 2018, 1499–1509. DOI: 10.1109/CVPRW.2018.00191 8, 15, 16.

[BNH19] BABADI, AMIN, NADERI, KOUROSH, and HÄMÄLÄINEN, PERTTU. "Self-Imitation Learning of Locomotion Movements through Termination Curriculum". *Proceedings of the ACM SIGGRAPH Conference on Motion, Interaction and Games*. Association for Computing Machinery, 2019. DOI: 10.1145/3359566.3360072 19, 21.

[CAW*19] CHIU, HSU-KUANG, ADELI, EHSAN, WANG, BORUI, et al. "Action-Agnostic Human Pose Forecasting". *Proceedings of the IEEE Winter Conference on Applications of Computer Vision*. IEEE Computer Society, Jan. 2019, 1423–1432. DOI: 10.1109/WACV.2019.00156 8, 10, 16.

[CGM*20] CAO, ZHE, GAO, HANG, MANGALAM, KARTTIKEYA, et al. "Long-Term Human Motion Prediction with Scene Context". *Proceedings of the European Conference on Computer Vision*. Springer International Publishing, Sept. 2020, 387–404. DOI: 10.1007/978-3-030-58452-8_23 6, 12, 16.

[CHW*20] CAI, YUJUN, HUANG, LIN, WANG, YIWEI, et al. "Learning Progressive Joint Propagation for Human Motion Prediction". *Proceedings of the European Conference on Computer Vision*. Springer International Publishing, Sept. 2020. DOI: 10.1007/978-3-030-58571-6_14 5, 8, 10, 11, 16, 24.

[CM11] CHIU, CHUNG-CHENG and MARSELLA, STACY. "A style controller for generating virtual human behaviors". *Proceedings of the International Conference on Autonomous Agents and Multiagent Systems*. International Foundation for Autonomous Agents and Multiagent Systems, May 2011, 1023–1030. URL: http://www.ifaamas.org/Proceedings/aamas2011/papers/D7_G77.pdf 4, 14, 16.

[CMM*18] CHENTANEZ, NUTTAPONG, MÜLLER, MATTHIAS, MACKLIN, MILES, et al. "Physics-Based Motion Capture Imitation with Deep Reinforcement Learning". *Proceedings of the ACM SIGGRAPH Conference on Motion, Interaction and Games*. Association for Computing Machinery, Nov. 2018. DOI: 10.1145/3274247.3274506 19, 21.

[CPAM20] CORONA, ENRIC, PUMAROLA, ALBERT, ALENYÀ, GUILLEM, and MORENO-NOGUER, FRANCESC. "Context-Aware Human Motion Prediction". *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. IEEE Computer Society, June 2020, 6990–6999. DOI: 10.1109/CVPR42600.2020.00702 8, 10, 11, 16.

[CSK*21] CUI, QIONGJIE, SUN, HUAIJIANG, KONG, YUE, et al. "Efficient human motion prediction using temporal convolutional generative adversarial network". *Information Sciences* 545 (Feb. 2021), 427–447. DOI: 10.1016/j.ins.2020.08.123 4, 5, 8, 11, 16.

[CSY20] CUI, QIONGJIE, SUN, HUAIJIANG, and YANG, FEI. "Learning Dynamic Relationships for 3D Human Motion Prediction". *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. June 2020, 6519–6527. DOI: 10.1109/CVPR42600.2020.00655 4, 5, 7, 8, 10, 11, 16.

[CYT*18] CLEGG, ALEXANDER, YU, WENHAO, TAN, JIE, et al. "Learning to Dress: Synthesizing Human Dressing Motion via Deep Reinforcement Learning". *ACM Transactions on Graphics* 37.6 (Dec. 2018). DOI: 10.1145/3272127.3275048 20, 21.

[DAS*20] DONG, YUZHU, ARISTIDOU, ANDREAS, SHAMIR, ARIEL, et al. "Adult2child: Motion Style Transfer using CycleGANs". *Proceedings of the ACM SIGGRAPH Conference on Motion, Interaction and Games*. Association for Computing Machinery, Oct. 2020. DOI: 10.1145/3424636.3426909 6, 8, 23–25.

[DHS*19] DU, HAN, HERRMANN, ERIK, SPRENGER, JANIS, et al. "Stylistic Locomotion Modeling and Synthesis Using Variational Generative Models". *Proceedings of the ACM SIGGRAPH Conference on Motion, Interaction and Games*. Association for Computing Machinery, Oct. 2019. DOI: 10.1145/3359566.3360083 3, 8, 14, 16.

[DKB14] DINH, LAURENT, KRUEGER, DAVID, and BENGIO, YOSHUA. "NICE: Non-linear Independent Components Estimation". *arXiv e-prints* (Oct. 2014). eprint: 1410.8516 15.

[FLFM15] FRAGKIADAKI, KATERINA, LEVINE, SERGEY, FELSEN, PANNA, and MALIK, JITENDRA. "Recurrent Network Models for Human Dynamics". *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*. IEEE Computer Society, Dec. 2015, 4346–4354. DOI: 10.1109/ICCV.2015.494 4, 8, 9, 11, 16.

[FP14] FOURATI, NESRINE and PELACHAUD, CATHERINE. "Emilya: Emotional Body Expression in Daily Actions Database". *Proceedings of the International Conference on Language Resources and Evaluation (LREC)*. European Language Resources Association, May 2014. URL: http://www.lrec-conf.org/proceedings/lrec2014/summaries/334.html 6.

[GC19] GUO, XIAO and CHOI, JONGMOO. "Human Motion Prediction via Learning Local Structure Representations and Temporal Dependencies". *Proceedings of the AAAI Conference on Artificial Intelligence* 33.1 (July 2019). DOI: 10.1609/aaai.v33i01.33012580 4, 5, 7, 8, 10, 16.

[GEB16] GATYS, LEON A., ECKER, ALEXANDER S., and BETHGE, MATTHIAS. "Image Style Transfer Using Convolutional Neural Networks". *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. IEEE Computer Society, June 2016, 2414–2423. DOI: 10.1109/CVPR.2016.265 23.

[Gle98] GLEICHER, MICHAEL. "Retargetting Motion to New Characters". *Proceedings of the Annual Conference on Computer Graphics and Interactive Techniques*. Association for Computing Machinery, 1998, 33–42. DOI: 10.1145/280814.280820 1, 22.

[GMK*19] GOPALAKRISHNAN, ANAND, MALI, ANKUR, KIFER, DAN, et al. "A Neural Temporal Model for Human Motion Prediction". *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. IEEE Computer Society, June 2019, 12108–12117. DOI: 10.1109/CVPR.2019.01239 4, 8, 12, 16.

[GP12] GEIJTENBEEK, THOMAS and PRONOST, NICOLAS. "Interactive Character Animation Using Simulated Physics: A State-of-the-Art Review". *Computer Graphics Forum* 31.8 (Sept. 2012), 2492–2515. DOI: 10.1111/j.1467-8659.2012.03189.x 2.

[GPM*14] GOODFELLOW, IAN, POUGET-ABADIE, JEAN, MIRZA, MEHDI, et al. "Generative Adversarial Nets". *Proceedings of the International Conference on Neural Information Processing Systems*. Curran Associates Inc., Dec. 2014, 2672–2680. URL: http://papers.nips.cc/paper/5423.pdf 12.

[Gra98] GRASSIA, F. SEBASTIAN. "Practical Parameterization of Rotations Using the Exponential Map". *Journal of Graphics Tools* 3.3 (Apr. 1998), 29–48. DOI: 10.1080/10867651.1998.10487493 3, 4.

[GSAH17] GHOSH, PARTHA, SONG, JIE, AKSAN, EMRE, and HILLIGES, OTMAR. "Learning Human Motion Models for Long-Term Predictions". *Proceedings of the International Conference on 3D Vision*. IEEE Computer Society, Oct. 2017, 458–466. DOI: 10.1109/3DV.2017.00059 8, 12, 16.

[GTH98] GRZESZCZUK, RADEK, TERZOPOULOS, DEMETRI, and HINTON, GEOFFREY. "NeuroAnimator: Fast Neural Network Emulation and Control of Physics-Based Models". *Proceedings of the ACM Computer Graphics and Interactive Techniques*. Association for Computing Machinery, July 1998, 9–20. DOI: 10.1145/280814.280816 18.

[GWE*20] GHORBANI, SAEED, WLOKA, CALDEN, ETEMAD, ALI, et al. "Probabilistic Character Motion Synthesis using a Hierarchical Deep Latent Variable Model". *Computer Graphics Forum* (Oct. 2020). DOI: 10.1111/cgf.14116 4, 5, 8, 12, 16.

[GWLM18] GUI, LIANG-YAN, WANG, YU-XIONG, LIANG, XIAODAN, and MOURA, JOSÉ M. F. "Adversarial Geometry-Aware Human Motion Prediction". *Proceedings of the European Conference on Computer Vision*. Springer International Publishing, Sept. 2018, 823–842. DOI: 10.1007/978-3-030-01225-0_48 4, 8, 10, 11, 16.

[GWRM18] GUI, LIANG-YAN, WANG, YU-XIONG, RAMANAN, DEVA, and MOURA, JOSÉ M. F. "Few-Shot Human Motion Prediction via Meta-Learning". *Proceedings of the European Conference on Computer Vision*. Springer International Publishing, Sept. 2018, 441–459. DOI: 10.1007/978-3-030-01237-3_27 8, 10, 16.

[HAB20] HENTER, GUSTAV EJE, ALEXANDERSON, SIMON, and BESKOW, JONAS. "MoGlow: Probabilistic and controllable motion synthesis using normalising flows". *ACM Transactions on Graphics* 39.6 (Nov. 2020). DOI: 10.1145/3414685.3417836 3, 8, 15, 16.

[HBM*20] HAWORTH, BRANDON, BERSETH, GLEN, MOON, SEONGHYEON, et al. "Deep Integration of Physical Humanoid Control and Crowd Navigation". *Proceedings of the ACM SIGGRAPH Conference on Motion, Interaction and Games*. Association for Computing Machinery, Oct. 2020. DOI: 10.1145/3424636.3426894 8, 20, 21.

[HCTB19] HASSAN, MOHAMED, CHOUTAS, VASILEIOS, TZIONAS, DIMITRIOS, and BLACK, MICHAEL J. "Resolving 3D Human Pose Ambiguities With 3D Scene Constraints". *Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. IEEE Computer Society, Oct. 2019, 2282–2292. DOI: 10.1109/ICCV.2019.00237 6.

[HGM19] HERNANDEZ RUIZ, ALEJANDRO, GALL, JÜRGEN, and MORENO, FRANCESC. "Human Motion Prediction via Spatio-Temporal Inpainting". *Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. IEEE Computer Society, Oct. 2019, 7133–7142. DOI: 10.1109/ICCV.2019.00723 8, 12, 16.

[HHKK17] HOLDEN, DANIEL, HABIBIE, IKHSANUL, KUSAJIMA, IKUO, and KOMURA, TAKU. "Fast Neural Style Transfer for Motion Data". *IEEE Computer Graphics and Applications (CG&A)* 37.4 (Aug. 2017), 42–49. DOI: 10.1109/MCG.2017.3271464 3, 8, 23.

[HHS*17] HABIBIE, IKHSANUL, HOLDEN, DANIEL, SCHWARZ, JONATHAN, et al. "A Recurrent Variational Autoencoder for Human Motion Synthesis". *Proceedings of the British Machine Vision Conference*. BMVA Press, Sept. 2017, 119.1–119.12. DOI: 10.5244/C.31.119 3, 6, 12, 14, 16.

[Hin02] HINTON, GEOFFREY E. "Training products of experts by minimizing contrastive divergence". *Neural Computation* 14.8 (Aug. 2002), 1771–1800. DOI: 10.1162/089976602760128018 13.

[HKPP20] HOLDEN, DANIEL, KANOUN, OUSSAMA, PEREPICHKA, MAKSYM, and POPA, TIBERIU. "Learned Motion Matching". *ACM Transactions on Graphics* 39.4 (July 2020). DOI: 10.1145/3386569.3392440 5, 18, 21.

[HKS17] HOLDEN, DANIEL, KOMURA, TAKU, and SAITO, JUN. "Phase-Functioned Neural Networks for Character Control". *ACM Transactions on Graphics* 36.4 (July 2017). DOI: 10.1145/3072959.3073663 4–6, 8, 17, 21.

[Hol18] HOLDEN, DANIEL. "Robust Solving of Optical Motion Capture Data by Denoising". *ACM Transactions on Graphics* 37.4 (July 2018). DOI: 10.1145/3197517.3201302 22, 23.

[HPP05] HSU, EUGENE, PILLI, KARI, and POPOVIC, JOVAN. "Style Translation for Human Motion". *Proceedings of the Annual Conference on Computer Graphics and Interactive Techniques*. Association for Computing Machinery, July 2005, 1082–1089. DOI: 10.1145/1186822.1073315 6.

[HSK16] HOLDEN, DANIEL, SAITO, JUN, and KOMURA, TAKU. "A Deep Learning Framework for Character Motion Synthesis and Editing". *ACM Transactions on Graphics* 35.4 (July 2016). DOI: 10.1145/2897824.2925975 3, 6, 8, 13–16, 21, 23, 25.

[HSKJ15] HOLDEN, DANIEL, SAITO, JUN, KOMURA, TAKU, and JOYCE, THOMAS. "Learning Motion Manifolds with Convolutional Autoencoders". *SIGGRAPH Asia Technical Briefs*. Association for Computing Machinery, 2015. DOI: 10.1145/2820903.2820918 3, 21, 23, 25.

[HTS*17] HEESS, NICOLAS, TB, DHRUVA, SRINIVASAN, SRIRAM, et al. "Emergence of Locomotion Behaviours in Rich Environments". *arXiv e-prints* (July 2017). eprint: 1707.02286 8, 19, 21.

[HXZ*20] HABERMANN, MARC, XU, WEIPENG, ZOLLHÖFER, MICHAEL, et al. "DeepCap: Monocular Human Performance Capture Using Weak Supervision". *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. IEEE Computer Society, June 2020, 5051–5062. DOI: 10.1109/CVPR42600.2020.00510 6.

[HYNP20] HARVEY, FÉLIX GINGRAS, YURICK, MIKE, NOWROUZEZAHRAI, DEREK, and PAL, CHRISTOPHER. "Robust Motion In-betweening". *ACM Transactions on Graphics* 39.4 (July 2020). DOI: 10.1145/3386569.3392480 4, 6, 8, 13, 15, 16.

[IPOS14] IONESCU, CATALIN, PAPAVA, DRAGOS, OLARU, VLAD, and SMINCHISESCU, CRISTIAN. "Human3.6M: Large Scale Datasets and Predictive Methods for 3D Human Sensing in Natural Environments". *IEEE Transactions on Pattern Analysis and Machine Intelligence* 36.7 (July 2014), 1325–1339. DOI: 10.1109/TPAMI.2013.248 5, 6, 11, 16, 23.

[JL20] JANG, DEOK-KYEONG and LEE, SUNG-HEE. "Constructing Human Motion Manifold With Sequential Networks". *Computer Graphics Forum* 39.2 (July 2020), 487–496. DOI: 10.1111/cgf.14028 4, 8, 21, 23.

[JVDL19] JIANG, YIFENG, VAN WOUWE, TOM, DE GROOTE, FRIEDL, and LIU, C. KAREN. "Synthesis of Biologically Realistic Human Motion Using Joint Torque Actuation". *ACM Transactions on Graphics* 38.4 (July 2019). DOI: 10.1145/3306346.3322966 20, 21, 25.

This article has been published in COMPUTER GRAPHICS *Forum* (CGF). https://doi.org/10.1111/cgf.14426

[JZSS16] JAIN, ASHESH, ZAMIR, AMIR R., SAVARESE, SILVIO, and SAXENA, ASHUTOSH. "Structural-RNN: Deep Learning on Spatio-Temporal Graphs". *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. IEEE Computer Society, June 2016, 5308–5317. DOI: 10.1109/CVPR.2016.573 4, 7–9, 11, 16.

[KAS*20] KAUFMANN, MANUEL, ASKAN, EMRE, SONG, JIE, et al. "Convolutional Autoencoders for Human Motion Infilling". *Proceedings of the International Conference on 3D Vision*. IEEE Computer Society, Nov. 2020, 918–927. DOI: 10.1109/3DV50981.2020.00102 3, 13, 16.

[KGB19] KUNDU, JOGENDRA NATH, GOR, MAHARSHI, and BABU, R. VENKATESH. "BiHMP-GAN: Bidirectional 3D Human Motion Prediction GAN". *Proceedings of the AAAI Conference on Artificial Intelligence* 33.1 (July 2019). DOI: 10.1609/AAAI.V33I01.33018553 4, 5, 8, 12, 16.

[KGP02] KOVAR, LUCAS, GLEICHER, MICHAEL, and PIGHIN, FRÉDÉRIC. "Motion Graphs". *Proceedings of the Annual Conference on Computer Graphics and Interactive Techniques*. Vol. 21. 3. Association for Computing Machinery, July 2002, 473–482. DOI: 10.1145/566654.566605 1.

[KLvdP20] KWON, TAESOO, LEE, YOONSANG, and van de PANNE, MICHIEL. "Fast and Flexible Multilegged Locomotion Using Learned Centroidal Dynamics". *ACM Transactions on Graphics* 39.4 (July 2020). DOI: 10.1145/3386569.3392432 19, 21.

[KNH*21] KHAN, SALMAN, NASEER, MUZAMMAL, HAYAT, MUNAWAR, et al. "Transformers in Vision: A Survey". *arXiv e-prints* (2021). eprint: 2101.01169 10.

[KPKH20] KIM, SANGBIN, PARK, INBUM, KWON, SEONGSU, and HAN, JUNGHYUN. "Motion Retargetting based on Dilated Convolutions and Skeleton-specific Loss Functions". *Computer Graphics Forum* 39.2 (July 2020), 497–507. DOI: 10.1111/cgf.13947 4, 8, 22, 23.

[KSG*13] KARG, MICHELLE, SAMADANI, ALI-AKBAR, GORBET, ROB, et al. "Body Movements for Affective Expression: A Survey of Automatic Recognition and Generation". *IEEE Transactions on Affective Computing* 4.4 (Nov. 2013), 341–359. DOI: 10.1109/T-AFFC.2013.29 2.

[KW14] KINGMA, DIEDERIK P. and WELLING, MAX. "Auto-Encoding Variational Bayes". *Proceedings of the International Conference on Learning Representations*. Apr. 2014. URL: https://openreview.net/forum?id=33X9fd2-9FyZd 14.

[KW17] KIPF, THOMAS N. and WELLING, MAX. "Semi-Supervised Classification with Graph Convolutional Networks". *Proceedings of the International Conference on Learning Representations*. Apr. 2017. URL: https://openreview.net/forum?id=SJU4ayYgl 7.

[LAT21] LOHIT, SUHAS, ANIRUDH, RUSHIL, and TURAGA, PAVAN. "Recovering Trajectories of Unmarked Joints in 3D Human Actions Using Latent Space Optimization". *Proceedings of the IEEE Winter Conference on Applications of Computer Vision*. IEEE Computer Society, Jan. 2021, 2341–2350. DOI: 10.1109/WACV48630.2021.00239 8, 21–23.

[LCC*21] LI, MAOSEN, CHEN, SIHENG, CHEN, XU, et al. "Symbiotic Graph Neural Networks for 3D Skeleton-based Human Action Recognition and Motion Prediction". *IEEE Transactions on Pattern Analysis and Machine Intelligence* (Jan. 2021), 1–1. DOI: 10.1109/TPAMI.2021.3053765 4, 5, 7, 10, 16.

[LCC19] LIM, JONGIN, CHANG, HYUNG JIN, and CHOI, JIN YOUNG. "PMnet: Learning of Disentangled Pose and Movement for Unsupervised Motion Retargeting". *Proceedings of the British Machine Vision Conference*. BMVA Press, Sept. 2019, 136. URL: https://bmvc2019.org/wp-content/uploads/papers/0997-paper.pdf 4, 8, 22, 23.

[LCZ*20] LI, MAOSEN, CHEN, SIHENG, ZHAO, YANGHENG, et al. "Dynamic Multiscale Graph Neural Networks for 3D Skeleton Based Human Motion Prediction". *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. IEEE Computer Society, June 2020, 211–220. DOI: 10.1109/CVPR42600.2020.00029 4, 5, 7, 8, 10, 11, 16.

[LH17] LIU, LIBIN and HODGINS, JESSICA K. "Learning to Schedule Control Fragments for Physics-Based Characters Using Deep Q-Learning". *ACM Transactions on Graphics* 36.3 (June 2017). DOI: 10.1145/3083723 20, 21.

[LH18] LIU, LIBIN and HODGINS, JESSICA K. "Learning Basketball Dribbling Skills Using Trajectory Optimization and Deep Reinforcement Learning". *ACM Transactions on Graphics* 37.4 (July 2018). DOI: 10.1145/3197517.3201315 20, 21.

[LKS*20] LEBAILLY, TIM, KICIROGLU, SENA, SALZMANN, MATHIEU, et al. "Motion Prediction Using Temporal Inception Module". *Proceedings of the Asian Conference on Computer Vision*. Springer International Publishing, Nov. 2020, 651–665. DOI: 10.1007/978-3-030-69532-3_39 5, 10, 11, 16.

[LLL18] LEE, KYUNGHO, LEE, SEYOUNG, and LEE, JEHEE. "Interactive Character Animation by Learning Multi-Objective Control". *ACM Transactions on Graphics* 37.6 (Dec. 2018). DOI: 10.1145/3272127.3275071 5, 8, 17, 18, 20, 21.

[LMR*15] LOPER, MATTHEW, MAHMOOD, NAUREEN, ROMERO, JAVIER, et al. "SMPL: A Skinned Multi-Person Linear Model". *ACM Transactions on Graphics* 34.6 (Oct. 2015). DOI: 10.1145/2816795.2818013 6.

[LPLL19] LEE, SEUNGHWAN, PARK, MOONSEOK, LEE, KYOUNGMIN, and LEE, JEHEE. "Scalable Muscle-Actuated Human Simulation and Control". *ACM Transactions on Graphics* 38.4 (July 2019). DOI: 10.1145/3306346.3322972 20, 21, 25.

[LSP*20] LIU, JUN, SHAHROUDY, AMIR, PEREZ, MAURICIO, et al. "NTU RGB+D 120: A Large-Scale Benchmark for 3D Human Activity Understanding". *IEEE Transactions on Pattern Analysis and Machine Intelligence* 42.10 (2020), 2684–2701. DOI: 10.1109/TPAMI.2019.2916873 6.

[LT06] LEE, SUNG-HEE and TERZOPOULOS, DEMETRI. "Heads up! Biomechanical Modeling and Neuromuscular Control of the Neck". *ACM Transactions on Graphics* 25.3 (July 2006), 1188–1198. DOI: 10.1145/1141911.1142013 20, 21.

[LWJ*19] LIU, ZHENGUANG, WU, SHUANG, JIN, SHUYUAN, et al. "Towards Natural and Accurate Future Motion Prediction of Humans and Animals". *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. IEEE Computer Society, June 2019, 9996–10004. DOI: 10.1109/CVPR.2019.01024 4, 8, 12, 16.

[LWY*19] LI, YANRAN, WANG, ZHAO, YANG, XIAOSONG, et al. "Efficient convolutional hierarchical autoencoder for human motion prediction". *The Visual Computer* 35.6 (June 2019), 1143–1156. DOI: 10.1007/s00371-019-01692-9 4, 5, 7, 9, 16.

[LZCvdP20] LING, HUNG YU, ZINNO, FABIO, CHENG, GEORGE, and van de PANNE, MICHIEL. "Character Controllers using Motion VAEs". *ACM Transactions on Graphics* 39.4 (July 2020). DOI: 10.1145/3386569.3392422 5, 8, 17, 21.

[LZL10] LI, WANQING, ZHANG, ZHENGYOU, and LIU, ZICHENG. "Action recognition based on a bag of 3D points". *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*. IEEE Computer Society, June 2010, 9–14. DOI: 10.1109/CVPRW.2010.5543273 6.

[LZLL18] LI, CHEN, ZHANG, ZHEN, LEE, WEE SUN, and LEE, GIM HEE. "Convolutional Sequence to Sequence Model for Human Dynamics". *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. IEEE Computer Society, June 2018, 5226–5234. DOI: 10.1109/CVPR.2018.00548 4, 5, 7, 10, 11, 16.

This article has been published in COMPUTER GRAPHICS *Forum* (CGF). https://doi.org/10.1111/cgf.14426

[LZZ*19] LI, SHUJIE, ZHOU, YANG, ZHU, HAISHENG, et al. "Bidirectional recurrent autoencoder for 3D skeleton motion data refinement". *Computers & Graphics* 81 (2019), 92–103. DOI: 10.1016/j.cag.2019.03.010 8, 21, 23.

[LZZL20] LI, SHU-JIE, ZHU, HAI-SHENG, ZHENG, LI-PENG, and LI, LIN. "A Perceptual-Based Noise-Agnostic 3D Skeleton Motion Data Refinement Network". *IEEE Access* 8 (2020), 52927–52940. DOI: 10.1109/ACCESS.2020.2980316 8, 17, 21, 23.

[MAP*19] MEREL, JOSH, AHUJA, ARUN, PHAM, VU, et al. "Hierarchical Visuomotor Control of Humanoids". *Proceedings of the International Conference on Learning Representations*. May 2019. URL: https://openreview.net/forum?id=BJfYvo09Y7 8, 19–21.

[MBR17] MARTINEZ, JULIETA, BLACK, MICHAEL J., and ROMERO, JAVIER. "On Human Motion Prediction Using Recurrent Neural Networks". *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. IEEE Computer Society, July 2017, 4674–4683. DOI: 10.1109/CVPR.2017.497 4, 8–11, 16.

[MC12] MIN, JIANYUAN and CHAI, JINXIANG. "Motion Graphs++: A Compact Generative Model for Semantic Motion Analysis and Synthesis". *ACM Transactions on Graphics* 31.6 (Nov. 2012). DOI: 10.1145/2366145.2366172 1, 14.

[MGT*19] MAHMOOD, NAUREEN, GHORBANI, NIMA, TROJE, NIKOLAUS F., et al. "AMASS: Archive of Motion Capture As Surface Shapes". *Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. IEEE Computer Society, Oct. 2019, 5441–5450. DOI: 10.1109/ICCV.2019.00554 6.

[MHG*19] MEREL, JOSH, HASENCLEVER, LEONARD, GALASHOV, ALEXANDRE, et al. "Neural Probabilistic Motor Primitives for Humanoid Control". *Proceedings of the International Conference on Learning Representations*. May 2019. URL: https://openreview.net/forum?id=BJl6TjRcY7 20, 21.

[MLA*15] MORDATCH, IGOR, LOWREY, KENDALL, ANDREW, GALEN, et al. "Interactive Control of Diverse Complex Characters with Neural Networks". *Proceedings of the International Conference on Neural Information Processing Systems*. The MIT Press, Dec. 2015, 3132–3140. DOI: 10.5555/2969442.2969589 19, 21.

[MLS20] MAO, WEI, LIU, MIAOMIAO, and SALZMANN, MATHIEU. "History Repeats Itself: Human Motion Prediction via Motion Attention". *Proceedings of the European Conference on Computer Vision*. Springer International Publishing, Sept. 2020, 474–489. DOI: 10.1007/978-3-030-58568-6_28 4, 8, 10, 11, 16, 24.

[MLSL19] MAO, WEI, LIU, MIAOMIAO, SALZMANN, MATHIEU, and LI, HONGDONG. "Learning Trajectory Dependencies for Human Motion Prediction". *Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. IEEE Computer Society, Oct. 2019, 9488–9496. DOI: 10.1109/ICCV.2019.00958 5, 7, 8, 10, 11, 16, 24.

[MRC*07] MÜLLER, MEINARD, RÖDER, TIDO, CLAUSEN, MICHAEL, et al. *Documentation Mocap Database HDM05*. Tech. rep. CG-2007-2. Universität Bonn, June 2007. URL: https://resources.mpi-inf.mpg.de/HDM05/07_MuRoClEbKrWe_HDM05.pdf 5, 6, 16, 23.

[MRC*17] MEHTA, DUSHYANT, RHODIN, HELGE, CASAS, DAN, et al. "Monocular 3D Human Pose Estimation In The Wild Using Improved CNN Supervision". *Proceedings of the International Conference on 3D Vision*. IEEE Computer Society, Oct. 2017, 506–516. DOI: 10.1109/3dv.2017.00064 6.

[MSJG15] MAKHZANI, ALIREZA, SHLENS, JONATHON, JAITLY, NAVDEEP, and GOODFELLOW, IAN. "Adversarial Autoencoders". *arXiv e-prints* (Nov. 2015). eprint: 1511.05644 15.

[MSZ*18] MASON, IAN, STARKE, SEBASTIAN, ZHANG, HE, et al. "Few-shot Learning of Homogeneous Human Locomotion Styles". *Computer Graphics Forum* 37.7 (Oct. 2018), 143–153. DOI: 10.1111/cgf.13555 4, 5, 17, 21.

[MTA*20] MEREL, JOSH, TUNYASUVUNAKOOL, SARAN, AHUJA, ARUN, et al. "Catch & Carry: Reusable Neural Controllers for Vision-Guided Whole-Body Tasks". *ACM Transactions on Graphics* 39.4 (July 2020). DOI: 10.1145/3386569.3392474 8, 20, 21.

[MTD*15] MANDERY, CHRISTIAN, TERLEMEZ, ÖMER, DO, MARTIN, et al. "The KIT whole-body human motion database". *Proceedings of the International Conference on Advanced Robotics*. July 2015, 329–336. DOI: 10.1109/ICAR.2015.7251476 6.

[MTT*17] MEREL, JOSH, TASSA, YUVAL, TB, DHRUVA, et al. "Learning human behaviors from motion capture by adversarial imitation". *arXiv e-prints* (July 2017). eprint: 1707.02201 19, 21.

[MYT*21] MA, LI-KE, YANG, ZESHI, TONG, XIN, et al. "Learning and Exploring Motor Skills with Spacetime Bounds". *Computer Graphics Forum* 40.2 (June 2021), 251–263. DOI: 10.1111/cgf.142630 19, 21.

[NT15] NAKADA, MASAKI and TERZOPOULOS, DEMETRI. "Deep Learning of Neuromuscular Control for Biomechanical Human Animation". *Advances in Visual Computing*. Springer International Publishing, Dec. 2015, 339–348. DOI: 10.1007/978-3-319-27857-5_31 20, 21.

[NZC*18] NAKADA, MASAKI, ZHOU, TAO, CHEN, HONGLIN, et al. "Deep Learning of Biomimetic Sensorimotor Control for Biomechanical Human Animation". *ACM Transactions on Graphics* 37.4 (July 2018). DOI: 10.1145/3197517.3201305 7, 20, 21.

[OCK*13] OFLI, FERDA, CHAUDHRY, RIZWAN, KURILLO, GREGORIJ, et al. "Berkeley MHAD: A comprehensive Multimodal Human Action Database". *Proceedings of the IEEE Winter Conference on Applications of Computer Vision*. IEEE Computer Society, Jan. 2013, 53–60. DOI: 10.1109/WACV.2013.6474999 6.

[PALvdP18] PENG, XUE BIN, ABBEEL, PIETER, LEVINE, SERGEY, and van de PANNE, MICHIEL. "DeepMimic: Example-Guided Deep Reinforcement Learning of Physics-Based Character Skills". *ACM Transactions on Graphics* 37.4 (July 2018). DOI: 10.1145/3197517.3201311 19, 21.

[PBvdP16] PENG, XUE BIN, BERSETH, GLEN, and van de PANNE, MICHIEL. "Terrain-Adaptive Locomotion Skills Using Deep Reinforcement Learning". *ACM Transactions on Graphics* 35.4 (July 2016). DOI: 10.1145/2897824.2925881 18.

[PBYV17] PENG, XUE BIN, BERSETH, GLEN, YIN, KANGKANG, and VAN DE PANNE, MICHIEL. "DeepLoco: Dynamic Locomotion Skills Using Hierarchical Deep Reinforcement Learning". *ACM Transactions on Graphics* 36.4 (July 2017). DOI: 10.1145/3072959.3073602 18, 20, 21.

[PCZ*19] PENG, XUE BIN, CHANG, MICHAEL, ZHANG, GRACE, et al. "MCP: Learning Composable Hierarchical Control with Multiplicative Compositional Policies". *Proceedings of the International Conference on Neural Information Processing Systems*. Curran Associates Inc., 2019, 3686–3697. URL: http://papers.nips.cc/paper/8626.pdf 19–21.

[PFAG20] PAVLLO, DARIO, FEICHTENHOFER, CHRISTOPH, AULI, MICHAEL, and GRANGIER, DAVID. "Modeling Human Motion with Quaternion-Based Neural Networks". *International Journal of Computer Vision (IJCV)* 128.4 (Apr. 2020), 855–872. DOI: 10.1007/s11263-019-01245-6 4, 5, 8, 10, 16, 24.

[PGA18] PAVLLO, DARIO, GRANGIER, DAVID, and AULI, MICHAEL. "QuaterNet: A Quaternion-based Recurrent Model for Human Motion". *Proceedings of the British Machine Vision Conference*. BMVA Press, Sept. 2018. URL: https://dblp.org/rec/conf/bmvc/PavlloGA18 4, 5, 8, 10–12, 16, 24.

[PKM*18] PENG, XUE BIN, KANAZAWA, ANGJOO, MALIK, JITENDRA, et al. "SFV: Reinforcement Learning of Physical Skills from Videos". *ACM Transactions on Graphics* 37.6 (Dec. 2018). DOI: 10.1145/3272127.3275014 19–21, 24.

[PP10] PEJSA, TOMISLAV and PANDZIC, IGOR. "State of the Art in Example-Based Motion Synthesis for Virtual Characters in Interactive Applications". *Computer Graphics Forum* 29.1 (Feb. 2010), 202–226. DOI: 10.1111/j.1467-8659.2009.01591.x 2.

[PRL*19] PARK, SOOHWAN, RYU, HOSEOK, LEE, SEYOUNG, et al. "Learning Predict-and-Simulate Policies from Unorganized Human Motion Data". *ACM Transactions on Graphics* 38.6 (Nov. 2019). DOI: 10.1145/3355089.3356501 20, 21, 25.

[PvdP17] PENG, XUE BIN and van de PANNE, MICHIEL. "Learning Locomotion Skills Using DeepRL: Does the Choice of Action Space Matter?": *Proceedings of the ACM SIGGRAPH / Eurographics Symposium on Computer Animation*. Association for Computing Machinery, July 2017. DOI: 10.1145/3099564.3099567 18, 21.

[RH17] RAJAMÄKI, JOOSE and HÄMÄLÄINEN, PERTTU. "Augmenting Sampling Based Controllers with Machine Learning". *Proceedings of the ACM SIGGRAPH / Eurographics Symposium on Computer Animation*. Association for Computing Machinery, July 2017. DOI: 10.1145/3099564.3099579 19, 21.

[RH19] RAJAMÄKI, JOOSE and HÄMÄLÄINEN, PERTTU. "Continuous Control Monte Carlo Tree Search Informed by Multiple Experts". *IEEE Transactions on Visualization and Computer Graphics (TVCG)* 25.8 (Aug. 2019), 2540–2553. DOI: 10.1109/TVCG.2018.2849386 19, 21.

[Rod40] RODRIGUES, OLINDE. "Des lois géométriques qui régissent les déplacements d'un système solide dans l'espace, et de la variation des coordonnées provenant de ces déplacements considérés indépendants des causes qui peuvent les produire". *Journal de Mathématiques Pures et Appliquées* 5 (1840), 380–440. URL: https://eudml.org/doc/234443 3.

[RXKZ19] RANGANATH, AVINASH, XU, PEI, KARAMOUZAS, IOANNIS, and ZORDAN, VICTOR. "Low Dimensional Motor Skill Learning Using Coactivation". *Proceedings of the ACM SIGGRAPH Conference on Motion, Interaction and Games*. Association for Computing Machinery, Oct. 2019. DOI: 10.1145/3359566.3360071 19, 21.

[SAA*20] SHI, MINGYI, ABERMAN, KFIR, ARISTIDOU, ANDREAS, et al. "MotioNet: 3D Human Motion Reconstruction from Monocular Video with Skeleton Consistency". *ACM Transactions on Graphics* 40.1 (Sept. 2020). DOI: 10.1145/3407659 5, 8, 22, 23.

[SBB09] SIGAL, LEONID, BALAN, ALEXANDRU O., and BLACK, MICHAEL J. "HumanEva: Synchronized Video and Motion Capture Dataset and Baseline Algorithm for Evaluation of Articulated Human Motion". *International Journal of Computer Vision (IJCV)* 87.1 (Aug. 2009), 4. DOI: 10.1007/s11263-009-0273-6 6.

[SCNW19] SMITH, HARRISON JESSE, CAO, CHEN, NEFF, MICHAEL, and WANG, YINGYING. "Efficient Neural Networks for Real-Time Motion Style Transfer". *Proceedings of the ACM Computer Graphics and Interactive Techniques* 2.2 (July 2019). DOI: 10.1145/3340254 3, 22, 23.

[SGXT20] SHIMADA, SOSHI, GOLYANIK, VLADISLAV, XU, WEIPENG, and THEOBALT, CHRISTIAN. "PhysCap: Physically Plausible Monocular 3D Motion Capture in Real Time". *ACM Transactions on Graphics* 39.6 (Dec. 2020). DOI: 10.1145/3414685.3417877 8, 22, 23.

[SLNW16] SHAHROUDY, AMIR, LIU, JUN, NG, TIAN-TSONG, and WANG, GANG. "NTU RGB+D: A Large Scale Dataset for 3D Human Activity Analysis". *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. IEEE Computer Society, June 2016, 1010–1019. DOI: 10.1109/CVPR.2016.115 5, 6, 16, 23.

[Smo86] SMOLENSKY, PAUL. "Information processing in dynamical systems: Foundations of harmony theory". *Parallel distributed processing: Explorations in the microstructure of cognition*. The MIT Press, Jan. 1986, 194–281. URL: https://apps.dtic.mil/sti/citations/ADA620727 13.

[SNF*13] SHUMAN, DAVID I, NARANG, SUNIL K., FROSSARD, PASCAL, et al. "The Emerging Field of Signal Processing on Graphs: Extending High-Dimensional Data Analysis to Networks and Other Irregular Domains". *IEEE Signal Processing Magazine* 30.3 (Apr. 2013), 83–98. DOI: 10.1109/MSP.2012.2235192 7.

[SWD*17] SCHULMAN, JOHN, WOLSKI, FILIP, DHARIWAL, PRAFULLA, et al. "Proximal policy optimization algorithms". *arXiv e-prints* (July 2017). eprint: 1707.06347 18.

[SZKS19] STARKE, SEBASTIAN, ZHANG, HE, KOMURA, TAKU, and SAITO, JUN. "Neural State Machine for Character-Scene Interactions". *ACM Transactions on Graphics* 38.6 (Nov. 2019). DOI: 10.1145/3355089.3356505 5, 8, 17, 21.

[SZKZ20] STARKE, SEBASTIAN, ZHAO, YIWEI, KOMURA, TAKU, and ZAMA, KAZI. "Local Motion Phases for Learning Multi-Contact Character Movements". *ACM Transactions on Graphics* 39.4 (July 2020). DOI: 10.1145/3386569.3392450 5, 8, 17, 18, 21.

[TCHG17] TOYER, SAM, CHERIAN, ANOOP, HAN, TENGDA, and GOULD, STEPHEN. "Human Pose Forecasting via Deep Markov Models". *Proceedings of the International Conference on Digital Image Computing: Techniques and Applications*. IEEE Computer Society, Dec. 2017, 1–8. DOI: 10.1109/DICTA.2017.8227441 3, 6, 8, 14, 16.

[TH09] TAYLOR, GRAHAM W. and HINTON, GEOFFREY E. "Factored Conditional Restricted Boltzmann Machines for Modeling Motion Style". *Proceedings of the International Conference on Machine Learning*. Association for Computing Machinery, June 2009, 1025–1032. DOI: 10.1145/1553374.1553505 3, 13, 16.

[THR06] TAYLOR, GRAHAM W., HINTON, GEOFFREY E., and ROWEIS, SAM. "Modeling Human Motion Using Binary Latent Variables". *Proceedings of the International Conference on Neural Information Processing Systems*. The MIT Press, Sept. 2006, 1345–1352. DOI: 10.7551/mitpress/7503.003.0173 3, 13, 16.

[TMLZ18] TANG, YONGYI, MA, LIN, LIU, WEI, and ZHENG, WEI-SHI. "Long-Term Human Motion Prediction by Modeling Motion Context and Enhancing Motion Dynamics". *Proceedings of the International Joint Conferences on Artificial Intelligence*. AAAI Press, July 2018, 935–941. DOI: 10.24963/ijcai.2018/130 4, 8, 12, 16.

[Uni03] UNIVERSITY, CARNEGIE-MELLON. *CMU Graphics Lab Motion Capture Database*. Accessed: 2021-09-16. 2003. URL: http://mocap.cs.cmu.edu 5, 6, 11, 16, 21, 23.

[Uni19] UNIVERSITY, SIMON FRASER. *SFU Motion Capture Database*. Accessed: 2021-09-16. 2019. URL: https://mocap.cs.sfu.ca/ 6.

[vMHB*18] Von MARCARD, TIMO, HENSCHEL, ROBERTO, BLACK, MICHAEL, et al. "Recovering Accurate 3D Human Pose in The Wild Using IMUs and a Moving Camera". *Proceedings of the European Conference on Computer Vision*. Springer International Publishing, Sept. 2018, 614–631. DOI: 10.1007/978-3-030-01249-6_37 6, 16.

[VYCL18] VILLEGAS, RUBEN, YANG, JIMEI, CEYLAN, DUYGU, and LEE, HONGLAK. "Neural Kinematic Networks for Unsupervised Motion Retargetting". *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. IEEE Computer Society, June 2018, 8639–8648. DOI: 10.1109/CVPR.2018.00901 4, 8, 22, 23.

[WAC*19] WANG, BORUI, ADELI, EHSAN, CHIU, HSU-KUANG, et al. "Imitation Learning for Human Pose Prediction". *Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. IEEE Computer Society, Oct. 2019, 7124–7133. DOI: 10.1109/ICCV.2019.00722 4, 8, 10, 16.

[WACD20] WANG, QI, ARTIÈRES, THIERRY, CHEN, MICKAEL, and DENOYER, LUDOVIC. "Adversarial learning for modeling human motion". *The Visual Computer* 36.6–8 (June 2020), 141–160. DOI: 10.1007/s00371-018-1594-7 8, 15, 16, 23.

This article has been published in COMPUTER GRAPHICS *Forum* (CGF). https://doi.org/10.1111/cgf.14426

[WCAD18] WANG, QI, CHEN, MICKAEL, ARTIÈRES, THIERRY, and DENOYER, LUDOVIC. "Transferring Style in Motion Capture Sequences with Adversarial Learning". *Proceedings of the European Symposium on Artificial Neural Network*. ESANN, Apr. 2018. URL: https://hal.archives-ouvertes.fr/hal-02100672/ 8, 23.

[WCW14] WANG, XIN, CHEN, QIUDI, and WANG, WANLIANG. "3D Human Motion Editing and Synthesis: A Survey". *Computational and Mathematical Methods in Medicine* (June 2014). DOI: 10.1155/2014/104535 2.

[WCX17] WANG, YUMENG, CHE, WUJUN, and XU, BO. "Encoder-Decoder Recurrent Network Model for Interactive Character Animation Generation". *The Visual Computer* 33.6–8 (June 2017), 971–980. DOI: 10.1007/s00371-017-1378-5 8, 18, 21.

[WCX21] WANG, ZHIYONG, CHAI, JINXIANG, and XIA, SHIHONG. "Combining Recurrent Neural Networks and Adversarial Training for Human Motion Synthesis and Control". *IEEE Transactions on Visualization and Computer Graphics (TVCG)* 27.1 (Jan. 2021), 14–28. DOI: 10.1109/TVCG.2019.2938520 8, 12, 15, 16, 21, 23.

[WGH20] WON, JUNGDAM, GOPINATH, DEEPAK, and HODGINS, JESSICA K. "A Scalable Approach to Control Diverse Behaviors for Physically Simulated Characters". *ACM Transactions on Graphics* 39.4 (July 2020). DOI: 10.1145/3386569.3392381 20, 21, 25.

[WGY18] WEISS, GAIL, GOLDBERG, YOAV, and YAHAV, ERAN. "On the Practical Computational Power of Finite Precision RNNs for Language Recognition". *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Association for Computational Linguistics, July 2018, 740–745. DOI: 10.18653/v1/P18-2117 8.

[WHSZ19] WANG, HE, HO, EDMOND SHU-LIM, SHUM, HUBERT P. H., and ZHU, ZHANXING. "Spatio-temporal Manifold Learning for Human Motions via Long-horizon Modeling". *IEEE Transactions on Visualization and Computer Graphics (TVCG)* (Aug. 2019). DOI: 10.1109/TVCG.2019.2936810 3, 7, 8, 12, 16, 21, 23.

[WL19] WON, JUNGDAM and LEE, JEHEE. "Learning Body Shape Variation in Physics-Based Characters". *ACM Transactions on Graphics* 38.6 (Nov. 2019). DOI: 10.1145/3355089.3356499 19, 21.

[WMR*17] WANG, ZIYU, MEREL, JOSH, REED, SCOTT, et al. "Robust Imitation of Diverse Behaviors". *Proceedings of the International Conference on Neural Information Processing Systems*. Curran Associates Inc., 2017, 5326–5335. URL: http://papers.nips.cc/paper/7116.pdf 8, 20, 21.

[WN15] WANG, YINGYING and NEFF, MICHAEL. "Deep Signatures for Indexing and Retrieval in Large Motion Databases". *Proceedings of the ACM SIGGRAPH Conference on Motion, Interaction and Games*. Association for Computing Machinery, Nov. 2015, 37–45. DOI: 10.1145/2822013.2822024 7, 21, 23.

[WP95] WITKIN, ANDREW and POPOVIC, ZORAN. "Motion Warping". *Proceedings of the Annual Conference on Computer Graphics and Interactive Techniques*. Association for Computing Machinery, 1995, 105–108. DOI: 10.1145/218380.218422 1.

[WYZ*20] WANG, ZHENYI, YU, PING, ZHAO, YANG, et al. "Learning Diverse Stochastic Human-Action Generators by Learning Smooth Latent Transitions". *Proceedings of the AAAI Conference on Artificial Intelligence* 34.7 (Apr. 2020), 12281–12288. DOI: 10.1609/aaai.v34i07.6911 8, 15, 16.

[XLKvdP20] XIE, ZHAOMING, LING, HUNG YU, KIM, NAM HEE, and van de PANNE, MICHIEL. "ALLSTEPS: Curriculum-driven Learning of Stepping Stone Skills". *Computer Graphics Forum* (Oct. 2020). DOI: 10.1111/cgf.14115 19, 21.

[XLM19] XU, YI TIAN, LI, YAQIAO, and MEGER, DAVID. "Human Motion Prediction Via Pattern Completion in Latent Representation Space". *Proceedings of the IEEE Conference on Computer and Robot Vision*. IEEE Computer Society, 2019, 57–64. DOI: 10.1109/CRV.2019.00016 8, 10, 12, 16.

[XWCC15] XIA, SHIHONG, WANG, CONGYI, CHAI, JINXIANG, and CHAI, JESSICA. "Realtime Style Transfer for Unlabeled Heterogeneous Human Motion". *ACM Transactions on Graphics* 34.4 (July 2015). DOI: 10.1145/2766999 6.

[XXN*20] XU, JINGWEI, XU, HUAZHE, NI, BINGBING, et al. "Hierarchical Style-based Networks for Motion Synthesis". *Proceedings of the European Conference on Computer Vision*. Springer International Publishing, Sept. 2020, 178–194. DOI: 10.1007/978-3-030-58621-8_11 8, 13, 16.

[YK20] YUAN, YE and KITANI, KRIS. "DLow: Diversifying Latent Flows for Diverse Human Motion Prediction". *Proceedings of the European Conference on Computer Vision*. Springer International Publishing, Sept. 2020, 346–364. DOI: 10.1007/978-3-030-58545-7_20 8, 15, 16.

[YKK*19] YU, MOONWON, KWON, BYUNGJUN, KIM, JONGMIN, et al. "Fast Terrain-Adaptive Motion Generation Using Deep Neural Networks". *SIGGRAPH Asia Technical Briefs*. Association for Computing Machinery, 2019, 57–60. DOI: 10.1145/3355088.3365157 13, 16.

[YKL21] YANG, DONGSEOK, KIM, DOYEON, and LEE, SUNG-HEE. "LoBSTr: Real-time Lower-body Pose Prediction from Sparse Upper-body Tracking Signals". *Computer Graphics Forum* 40.2 (June 2021), 265–275. DOI: 10.1111/cgf.142631 10, 16, 22, 23.

[YLX*19] YAN, SIJIE, LI, ZHIZHONG, XIONG, YUANJUN, et al. "Convolutional Sequence Generation for Skeleton-Based Action Synthesis". *Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. IEEE Computer Society, Oct. 2019, 4393–4401. DOI: 10.1109/ICCV.2019.00449 15, 16.

[YRV*18] YAN, XINCHEN, RASTOGI, AKASH, VILLEGAS, RUBEN, et al. "MT-VAE: Learning Motion Transformations to Generate Multimodal Human Dynamics". *Proceedings of the European Conference on Computer Vision*. Springer International Publishing, Sept. 2018, 276–293. DOI: 10.1007/978-3-030-01228-1_17 14, 16.

[YTL18] YU, WENHAO, TURK, GREG, and LIU, C. KAREN. "Learning Symmetric and Low-Energy Locomotion". *ACM Transactions on Graphics* 37.4 (July 2018). DOI: 10.1145/3197517.3201397 19–21.

[ZBL*19] ZHOU, YI, BARNES, CONNELLY, LU, JINGWAN, et al. "On the Continuity of Rotation Representations in Neural Networks". *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. IEEE Computer Society, June 2019, 5738–5746. DOI: 10.1109/CVPR.2019.00589 4.

[ZLX*18] ZHOU, YI, LI, ZIMO, XIAO, SHUANGJIU, et al. "Auto-Conditioned Recurrent Networks for Extended Complex Human Motion Synthesis". *Proceedings of the International Conference on Learning Representations*. May 2018. URL: https://openreview.net/forum?id=r11Q2SlRW 3, 8, 12, 16.

[ZLZ*17] ZHANG, WEICHEN, LIU, ZHIGUANG, ZHOU, LIUYANG, et al. "Martial Arts, Dancing and Sports dataset: A challenging stereo and multi-view dataset for 3D human pose estimation". *Image and Vision Computing* 61 (2017), 22–39. DOI: 10.1016/j.imavis.2017.02.002 6.

[ZPIE17] ZHU, JUN-YAN, PARK, TAESUNG, ISOLA, PHILLIP, and EFROS, ALEXEI A. "Unpaired Image-to-Image Translation Using Cycle-Consistent Adversarial Networks". *Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV)*. IEEE Computer Society, Oct. 2017, 2242–2251. DOI: 10.1109/ICCV.2017.244 24.

[ZPK20] ZANG, CHUANQI, PEI, MINGTAO, and KONG, YU. "Few-shot Human Motion Prediction via Learning Novel Motion Dynamics". *Proceedings of the International Joint Conferences on Artificial Intelligence*. Sept. 2020, 846–852. DOI: 10.24963/ijcai.2020/118 5, 7, 10, 16.

[ZSKS18] ZHANG, HE, STARKE, SEBASTIAN, KOMURA, TAKU, and SAITO, JUN. "Mode-Adaptive Neural Networks for Quadruped Motion Control". *ACM Transactions on Graphics* 37.4 (July 2018). DOI: 10.1145/3197517.3201366 5, 8, 17, 21.

This article has been published in COMPUTER GRAPHICS *Forum* (CGF). https://doi.org/10.1111/cgf.14426

[ZYC*20] ZOU, YULIANG, YANG, JIMEI, CEYLAN, DUYGU, et al. "Reducing Footskate in Human Motion Reconstruction with Ground Contact Constraints". *Proceedings of the IEEE Winter Conference on Applications of Computer Vision*. IEEE Computer Society, Mar. 2020, 448–457. DOI: 10.1109/WACV45572.2020.9093329 8, 22, 23.

[ZZD13] ZHANG, WEIYU, ZHU, MENGLONG, and DERPANIS, KONSTANTINOS G. "From Actemes to Action: A Strongly-Supervised Representation for Detailed Action Understanding". *Proceedings of the 2013 IEEE International Conference on Computer Vision (ICCV)*. IEEE Computer Society, Dec. 2013, 2248–2255. DOI: 10.1109/ICCV.2013.280 6.

This article has been published in COMPUTER GRAPHICS *Forum* (CGF). https://doi.org/10.1111/cgf.14426