Computer Science Review 58 (2025) 100780

Contents lists available at ScienceDirect

Computer Science Review

journal homepage: www.elsevier.com/locate/cosrev

Review article

Denoising and completion filters for human motion software: A survey with code

Enrico Martini [*], Andrea Calanca, Nicola Bombieri

Department of Engineering for Innovation Medicine, University of Verona, Verona, Italy

ARTICLE INFO

Keywords:
Human motion analysis
Human pose estimation
Filtering
Denoising
Refinement
Completion

ABSTRACT

Software platforms for human motion analysis are increasingly utilized across various fields, from Healthcare to Industry 5.0. However, the inherent inaccuracies of these platforms often lead to noisy observations of human poses or periods of missing information. As a result, data filtering for denoising or completion is a fundamental step before data analysis. Over the years, different techniques have been proposed, from general-purpose solutions based on low-pass filters to more advanced and embedded approaches based on state observers rather than deep learning. This survey presents the current state-of-the-art filtering solutions for denoising and completing data generated by software platforms for human motion analysis. It focuses on 3D positional data extrapolated through marker-based or marker-less motion capture systems. The survey proposes a concise taxonomy based on filter technology and application assumptions. For each class, it summarizes the basic concepts and reports application feedback collected from the literature. The survey also includes implementation codes or links to the authors' original codes, enabling readers to quickly reproduce all the algorithms in different experimental settings (https://github.com/PARCO-LAB/mocap-refinement).

Contents

* Corresponding author.
E-mail address: enrico.martini@univr.it (E. Martini).

https://doi.org/10.1016/j.cosrev.2025.100780
Received 21 October 2024; Received in revised form 6 May 2025; Accepted 13 June 2025
Available online 1 July 2025

1. Introduction

The understanding, analysis, and evaluation of human movements are playing an increasingly important role in people's professional and personal lives and are presently a significant topic in research on human–machine interaction [1–3]. Various technologies have been developed to capture the human motion. Some rely on reflective *markers* placed on the human body and a multi-camera optical system that uses triangulation of the reflected light to calculate the 3D position of each marker [4,5]. Others rely on RGB or RGB-D camera sensors and estimate the human pose through *inference* on Deep Neural Networks (DNN) without the need for markers [6,7].

Aside from implementation technology, platforms for human motion analysis extrapolate *keypoints* that represent the spatial coordinates of the main human body joints. Compared to video data, skeletons are a compact representation that naturally enables quantitative information retrieval on human actions. It is consistent across subjects and application scenarios and independent of changing backgrounds or viewpoints in the video. Based on the skeleton pose data, kinematic and semantic features can be easily derived, which are compatible with existing motion analysis practices in high-level vision tasks [8–11], sports [12], industrial [13,14], and medical applications [15].

Both marker-less and marker-based *motion capture* (mocap) systems have an intrinsic inaccuracy, often leading to noisy keypoints, although varying degrees. The inaccuracy sources of marker-less human pose estimation (HPE) systems are of various types, including imprecise or inconsistent annotations in the training dataset, poor image or depth data quality, rare poses, dropped frames, or heavy occlusions in the real scene [16,17]. Misplacements, occlusions, and dropped markers are common noise sources in marker-based mocap systems [18]. The combination of such noise sources, along with temporary desynchronizations between the optics and the elaboration software, leads to periods during which the human pose information is even missing [19].

Furthermore, keypoint coordinates are often utilized to extract more complex information about human poses and motion, such as joint angles and angular velocity. However, errors in the initial measurements are amplified when mathematical differentiation is used to extrapolate velocities and even further when a second differentiation is calculated to determine body accelerations [20]. Consequently, data filtering for keypoint denoising or skeleton completion is crucial before data analysis.

Various filtering techniques have been proposed for 3D human motion data. To the best of our knowledge, the literature does not currently provide comprehensive reviews that classify and explain these different solutions. This survey aims to fill that gap by presenting an overview of denoising and completion filters that can be applied to refine 3D human motion data generated by marker-based or marker-less mocap systems. The survey focuses on designing, optimizing, and combining filtering methods. It begins with an overview of the different measurement errors. It proposes a taxonomy of the different filtering techniques, where each class is explained in terms of key concepts, target errors, and important application results from the literature.

This paper also provides an open-source library containing each reviewed filtering algorithm's implementation code (or links to the authors' original code). The repository also includes the precision of the filtering results obtained using the *Human3.6M* motion capture dataset [21]. Additionally, the survey explains the most common metrics and standard datasets used in literature to measure the accuracy of motion capture data and the performance of the filtering techniques.

The article is organized as follows. Section 2 presents the basic concepts related to human motion analyses, filtering, and the problem statement. Section 3 presents an overview of the literature analysis and the proposed taxonomy. Sections 4–8.4 present key concepts of each filtering class and major feedback from the literature. Section 12 concludes the work with remarks.

2. Basic concepts and problem statement

This survey focuses on filtering algorithms for motion capture systems (*mocaps*) that take a 3D skeleton of noisy keypoints as input and generate a corresponding 3D skeleton of refined keypoints as output (see Fig. 1). Filtering algorithms for systems that cannot directly measure 3D human positions, such as inertial measurement units, fall outside the scope of this survey. Throughout this survey, we refer to the platform for acquiring the 3D skeleton as the *mocap*, regardless of whether it uses marker-based or marker-less technology.

The filtering methods implement *denoising*, *completion*, or a combination of the two to enhance the accuracy of 3D human pose estimation data. Denoising refers to removing noise from the raw data by eliminating errors or inaccuracies that may have occurred during data capture or processing of a frame. Completion fills in missing data points in the 3D pose data that may have occurred due to occlusions or other factors that prevented certain joints from being captured during the data collection process.

In this context, we define *frame* k as a time instant when the mocap captures the pose of a human body and extrapolates the corresponding spatial information. A *keypoint* is a term used to refer to the 3D position of a physical or virtual marker:

$$kp_i[k] = \begin{bmatrix} p_x[k], & p_y[k], & p_z[k] \end{bmatrix} \tag{1}$$

where $p_x[k], p_y[k], p_z[k]$ are the coordinates on the x-axis, y-axis, and z-axis, respectively.

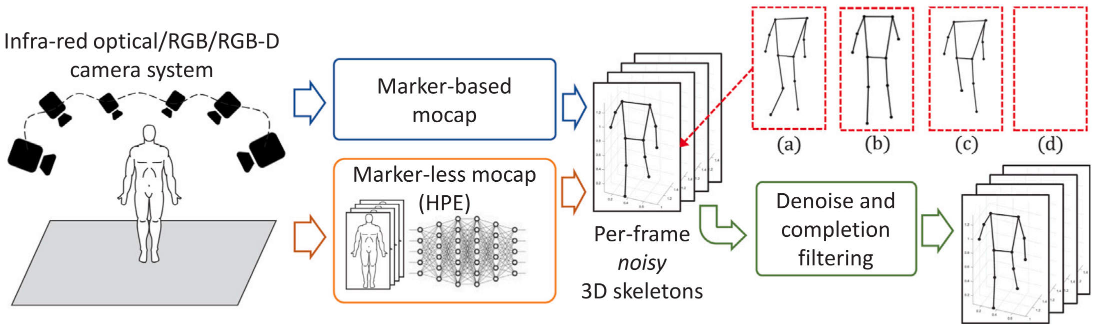
Infra-red optical/RGB/RGB-D camera system
Marker-based mocap
Marker-less mocap (HPE)
Per-frame *noisy* 3D skeletons
Denoise and completion filtering
(a)
(b)
(c)
(d)

Fig. 1. Standard pipeline of a motion capture (mocap) system, from capturing the subject movements to the corresponding representation through skeletons. Marker-based mocap systems typically rely on multiple infrared cameras, while markerless HPE systems rely on RGB or RGB-D cameras. Both systems generate one 3D skeleton per frame, which may have artifacts: inaccurate keypoint localization (a), missing keypoints (b), or a combination of the two (c). For one or a sequence of frames, the skeleton might also not exist (d). Each filtering algorithm discussed in this survey refines the per-frame noisy 3D skeleton by estimating the true position of the keypoints.

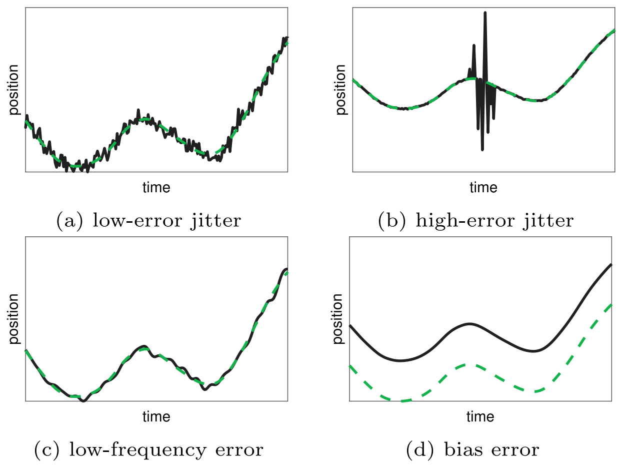

Fig. 2. Overview of the different types of noise.

We define *temporal window* Δ of length l:

$$\Delta_l = \left[k - \frac{l}{2}, k + \frac{l}{2}\right] \tag{2}$$

a sequence of consecutive frames. We define the skeleton sk as the collection of keypoints that belong to the same person:

$$sk_n[k] = \{kp_{i,n}[k] \ : \ i = 1 \ldots |\text{kps}|\} \quad n \in [1, N] \tag{3}$$

where |kps| is the number of keypoints detected through the mocap. N is the maximum number of people detected, distinguishing single-person (e.g., [22–24]) from multi-person mocap systems (e.g., [25–28]). Throughout this survey, we assume one skeleton per frame ($N = 1$).

2.1. Measurement errors

In marker-less mocaps, occlusions, HPE limitations, or inaccuracies of input sensors such as depth cameras can result in noisy or missing keypoints. In marker-based mocaps, a missing keypoint is generally due to occlusions or issues related to the physical markers, such as the markers falling off the subject's skin. In this study, we define *per-frame errors* as the observable and measurable artifacts on a single frame, which include noisy or missing keypoints (as shown in Fig. 1(a)–(d)). When the localization inaccuracy of a keypoint is significant, it is referred to as an *outlier*.

Per-frame errors that persist over temporal windows lead to *jittering*, *low-frequency*, or *bias* errors in the measurement of human motion (see Fig. 2). The jittering problem is especially noticeable in marker-less mocaps, primarily because HPE frameworks do not consider the temporal coherence across successive frames. Other factors that can cause jittering include poor image quality, unusual skeleton poses, or occlusions [29].

Filtering solutions for missing or noisy keypoints differ in their ability to handle the number of missing keypoints and the number of frames in which they occur. This survey refers to these methods as *sequence completion methods*. We refer to all processes that reduce noise and jittering as *denoising methods*.

3. Literature review and classification

Refining the keypoints position is crucial for enhancing the accuracy of motion capture systems, including marker-less and marker-based approaches. During our literature review, we systematically searched the Google Scholar database using various combinations of keywords such as "human motion", "motion capture", and "human pose estimation" along with "denoising" and "completion". We gathered 167 relevant papers and eliminated duplicates, ultimately selecting 114 highly relevant articles. These articles are presented in Table 1.

We classified these contributions into five different classes (see Fig. 3):

- *General purpose*, which refine data using well-established arithmetic algorithms on spatial and temporal data without making specific assumptions about human biomechanics.
- *State observers* utilize prior knowledge about the system (such as keypoint position, velocity, or acceleration in past frames) to refine the motion sequences.
- *Dimensionality reduction*, which takes advantage of motion synergies that represent well-known motion control strategies adopted by the human brain. These algorithms extract a lower-dimensional representation of the motion sequence to refine the corrupted sequences.
- *Deep neural network*, a class of machine learning algorithms. In particular, we focus on Autoencoders, trained to refine noisy or incomplete pose estimations by mapping them to the *latent space* and decoding them back to the original space.
- *Hybrid approaches*. They combine two or more methods of the previous classes, aiming to leverage the complementary strengths of different technologies.

Fig. 4 shows the number of papers published in each category over the past few years. The following sections will focus on each class of filtering algorithms, explaining the underlying principles and the specific types of errors they solve and briefly summarizing the variations offered by each contribution.

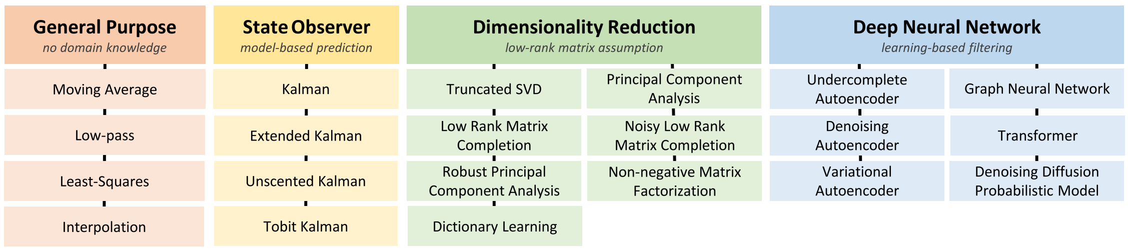

Fig. 3. Proposed classification of the filtering methods.

Table 1
Overview of approaches and the corresponding contributions.

Category	Method	Denoising	Completion	Contributions
General purpose	Moving average	✓		[30–32]
	Low-pass	✓		[33–38]
	Least-Squares	✓		[31,39]
	Interpolation		✓	[32,40–43]
State observer	Kalman	✓	✓	[31,32,39,44–48]
	Extended Kalman	✓	✓	[49,50]
	Unscented Kalman/Particle	✓	✓	[51–59]
Dimensionality reduction	Truncated singular value decomposition	✓		[60]
	Principal component analysis	✓	✓	[29,42,61–66]
	Low-Rank matrix completion		✓	[60,67–71]
	Noisy low-rank matrix completion	✓	✓	[69,72–78]
	Robust principal component analysis	✓		[79–87]
	Non-negative-Matrix factorization	✓		[88]
	Dictionary learning	✓	✓	[89–94]
Deep neural network	Undercomplete autoencoder	✓	✓	[95–97]
	Denoising autoencoder	✓	✓	[98–112]
	Variational autoencoder	✓	✓	[17,113], [114]
	Graph neural network	✓	✓	[115–121]
	Transformer	✓	✓	[122–129]
	Denoising diffusion probabilistic model	✓	✓	[130–135]
	Other architectures	✓	✓	[136–141]

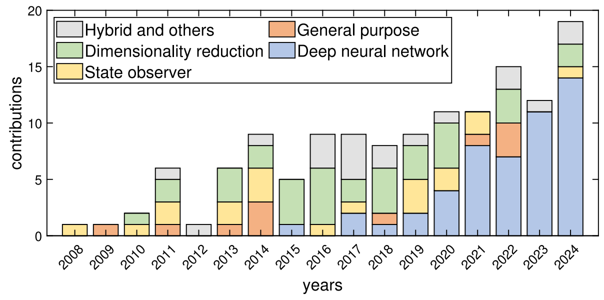

Fig. 4. Number of contributions related to methods for human motion refinement from 2008 to 2024.

4. General purpose

This class groups filtering algorithms universally adopted in signal processing theory and applied to refine human motion software. The filter design and parameters are independent of the input data, and they refine motion data using arithmetic algorithms on spatial or temporal information without making any specific assumptions about human biomechanics or motion synergies. Table 2 shows a comparison between all general-purpose methods.

4.1. Moving average

The moving average is the simplest family of filters that denoises keypoint information by smoothing their positional data in a given temporal window. The most straightforward is the *simple moving average* (SMA) filter. Given the coordinate vector p of a keypoint kp_i in the temporal window Δ_l, the SMA of p at frame k is defined as:

$$\mathrm{SMA}(p,k,\Delta) = \frac{1}{\Delta}\sum_{\delta=-\frac{\Delta}{2}}^{\frac{\Delta}{2}} p[k+\delta] \tag{4}$$

This filter can reduce noise, and its frequency behavior depends on the window length. A larger window can result in smoother outcomes and reduced outliers and jittering. Edwards and Green [31] conducted an analysis and found that an SMA filter with a window length of 5 frames provided the best denoising of upper limb motion data extrapolated by a markerless mocap system

The fundamental distinction among the different moving average filters is the *weighting function*. Whereas in SMA filters the information extrapolated from each frame of the window is weighted equally, *weighted moving average* (WMA) filters assign different weights to different frames. A WMA filter is defined as:

$$\mathrm{WMA}(p,k,\Delta,w) = \sum_{\delta=-\frac{\Delta}{2}}^{\frac{\Delta}{2}} p[k+\delta]\cdot w\left[\delta+\frac{\Delta}{2}\right] \tag{5}$$

where $w \in \mathbb{R}^{1\times\Delta}$ is an array of weights Wei et al. [30] used the WMA filter to denoise several high-velocity actions, such as jumping and dancing, captured from a marker-based mocap.

Exponential moving average (EMA) is a WMA filter in which the weight assigned to each frame used for the average calculation has an exponentially decaying value. Unlike SMA and WMA filters, EMA has an infinite window length and is defined recursively as an Infinite Impulse Response (IIR) filter:

$$\mathrm{EMA}(p,k) = \begin{cases} p[k] & k=0 \\ \alpha p[k] + (1-\alpha)\cdot \mathrm{EMA}(p,k-1) & k>0 \end{cases} \tag{6}$$

Table 2
Comparison of the general-purpose methods.

General purpose method	Advantages	Disadvantages
Moving average and low-pass	∘ Allows frequency control ∘ Easy to implement	∘ May blur motion details
Least squares	∘ High smoothness	∘ Poor at handling high-error jitter ∘ Only suited for slow motions
Interpolation	∘ Suited for for small gaps	∘ Cannot denoise existing keypoints

where $\alpha \in [0,1]$ is a user-defined coefficient to weigh all elements of the coordinate vector. Edwards and Green [31] compared the SMA and EMA filters and quantified the accuracy gain of EMA over SMA in denoising human motion. The Microsoft Kinect SDK uses the Holt double exponential (HDE) smoothing filter as the default filter for human pose estimation. HDE is a smoothing algorithm that recursively implements the EMA filter and tracks the slope of the data to preserve short-term fluctuations [142].

4.2. Low-pass

A low-pass filter allows only signals with a frequency lower than a chosen cutoff to pass through, while a *moving average* filter is a low-pass FIR filter with no direct control over the cutoff frequency. The *Butterworth filter* [143] is the most commonly used low-pass IIR filter in the literature for motion analysis [33–37]. It is the default denoising filter implemented in the user application software of Vicon [4] and Qualisys [144], two of the major marker-based mocap system platforms. Crenna et al. [38] tested the Butterworth filter on different human gestures (e.g., hopping and walking) captured by a marker-based mocap and claimed that it achieved the most reliable performance in removing synthetic white Gaussian noise when compared to SMA.

4.3. Least-squares

The least-squares (LS) filter denoises signals through a curve-fitting algorithm, which smooths trajectories of data samples over time to polynomial functions. LS is particularly well-suited for denoising human motion since it best applies to smooth values that change slowly and human motion frequencies typically fall between 0 and 20 Hz [145].

LS Gaussian is a variant of the LS filter that uses Gaussian functions for the impulse response and is best suited for situations where residuals around the true curve are assumed to be independently and identically distributed with mean zero and constant variance [146]. Li et al. [39] found that LS Gaussian filters achieved the best results in removing outliers along the trajectory of the upper limb during linear and planar motions.

Another filter that employs a similar concept to LS is the *Savitzky–Golay* (SG) filter, which uses closed-form linear coefficients so that the filtering results can be computed with a convolution [147]. Li et al. [39] showed that the SG filter outperforms the LS Gaussian filter in denoising human motion sequences obtained from a single camera marker-less mocap.

4.4. Interpolation

Interpolation is commonly used to recover signal gaps in various applications, including motion capture systems. When a sequence of skeletons in a temporal window has missing keypoints for a certain period of time, interpolation can be applied to estimate the missing keypoints from an incomplete sequence [148]. One of the most popular interpolation techniques is *spline* interpolation, which connects spatially or temporally adjacent points using low-degree polynomial functions to maximize the signal smoothness [149]. Several surveys focused on interpolation to refine motion data can be found in [32, 40–43].

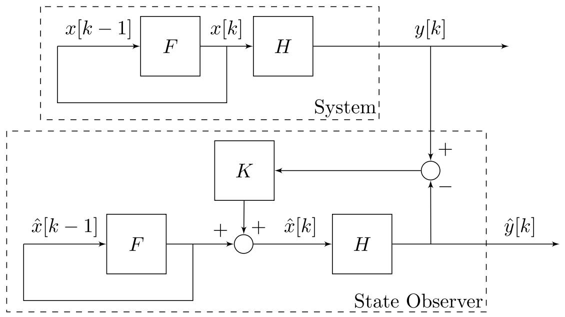

Fig. 5. Filtering through state observer: The observer corrects the keypoint position based on the previous state $\hat{x}[k-1]$. When a new observation $y[k]$ arrives, the observer performs a weighted average between the prediction of the inner state and the state obtained with the noisy measurement.

5. State observers

In control theory, a *state observer* is a system that estimates the internal state of a real system based on measurements of its input and output. The observer assumes an a priori model consisting of a transition function f and an output function h, which may depend on the system inputs. State observers refine signals by utilizing the knowledge contained in the system model (see Fig. 5). Typically, state observers have two phases: *prediction* and *correction*. The state estimate is projected forward in time during the prediction phase based on a priori knowledge. The measured output is used to refine the state estimate in the correction phase. In refining human motion, the system state is often represented by the body joints' position, velocity, acceleration, or angular velocity. The a priori model typically considers aspects related to human biomechanics, such as rigid body dynamics and kinematic constraints. Table 3 shows a comparison between all state-observer methods.

5.1. Kalman filter

The Kalman filter (KF) is one of the most representative and widely used examples of a state observer, especially for linear dynamic systems. It utilizes input/output measurements observed over time and assumes a noise model to estimate a Gaussian distribution of states with minimum variance for each time frame [150]. A KF model assumes that the state at time frame k evolves from the state at the previous time frame according to an a priori model in the form of:

$$x[k] = Fx[k-1] + q[k-1] \tag{7}$$

where F is the state transition matrix, x is the state vector, and $q \sim \mathcal{N}(0, Q)$ represents the process noise.

At frame k, the system output $y[k]$ is expressed as follows:

$$y[k] = Hx[k] + r[k] \tag{8}$$

where H is the observation matrix, which maps the state space onto the output. The matrix $r \sim \mathcal{N}(0, R)$ represents the covariance of the system measurements. Large values in the measurement covariance R, compared to the state covariance Q, indicate that the filter is less confident in the measured data than in the system knowledge

Table 3
Comparison of the state-observer methods.

State observer method	Advantages	Disadvantages
Kalman	∘ Low computational demands ∘ Suited for rectilinear motion ∘ Optimal for gaussian noise	∘ Less accurate in non-rectilinear movements
Extended Kalman	∘ Low computational demands ∘ Allows to describe complex motions	∘ Needs an explicit motion model
Unscented Kalman/Particle	∘ Does not need an explicit motion model	∘ More computationally expensive

In the literature, KF solutions primarily differ based on the choice of the state vector and transition matrices. Finding a transition matrix that effectively follows the evolution of the system state is crucial for implementing an accurate KF and may strongly depend on the application, such as straight or cyclic motion. Aristidou et al. [44], Wu et al. [45], and Hu et al. [48] considered a state as follows:

$$x[k] = \begin{bmatrix} p_x[k] & \dot{p}_x[k] & p_y[k] & \dot{p}_y[k] & p_z[k] & \dot{p}_z[k] \end{bmatrix}^T \tag{9}$$

where p and $\dot{p}$ are the position and velocity of keypoint kp_i, respectively. They assume each keypoint coordinate has *uniform rectilinear motion at constant speed*, and define the transition matrix F as follows:

$$F_x = F_y = F_z = \begin{bmatrix} 1 & dt \\ 0 & 1 \end{bmatrix} \quad F = \begin{bmatrix} F_x & 0 & 0 \\ 0 & F_y & 0 \\ 0 & 0 & F_z \end{bmatrix} \tag{10}$$

where dt is the time elapsed between two frames.

Edwards et al. [31] considered an *uniformly accelerated rectilinear motion* and defined x and F as follows:

$$x_{kp_i} = \begin{bmatrix} p_x & \dot{p}_x & \ddot{p}_x & p_y & \dot{p}_y & \ddot{p}_y & p_z & \dot{p}_z & \ddot{p}_z \end{bmatrix}^T \tag{11}$$

$$F_x = F_y = F_z = \begin{bmatrix} 1 & dt & \frac{1}{2}dt^2 \\ 0 & 1 & dt \\ 0 & 0 & 1 \end{bmatrix} \quad F = \begin{bmatrix} F_x & 0 & 0 \\ 0 & F_y & 0 \\ 0 & 0 & F_z \end{bmatrix} \tag{12}$$

Tripathy et al. [46] modeled the state vector as:

$$x_{kp_i}[k] = \begin{bmatrix} p_x^i[k] & p_x^j[k] & p_x^i[k] & p_x^j[k] & p_x^i[k] & p_x^j[k] \end{bmatrix}^T \tag{13}$$

where i and j are two physically connected joints of the human body. The transition matrix is designed to force a constant distance between such joints over time to reflect the human skeleton biomechanics.

5.2. Kalman filter variants

Although KF is best suited for linear dynamic systems, human motion often involves *non-linear transitions* [151]. To overcome this limitation, variants of KF refine the state transition and output of the system as follows

$$x[k] = f(x[k-1])y[k] = h(x[k]) \tag{14}$$

where f and h are non-linear functions.

The *Extended Kalman Filter* (EKF) is the most commonly used non-linear extension of the standard KF. It works by locally linearizing the transition function at each iteration step. In EKF, matrices F and H in Eqs. (7) and (8) are replaced by non-linear functions f and h, which can better model non-linear dynamics. To compute the optimal gain, EKF linearizes f and h around the current estimate by computing a matrix of partial derivatives [152]. However, if f and h are highly non-linear, EKF may perform poorly since the gain computation covariance propagates through linearization [153]. Some studies have successfully used EKF to denoise markerless mocap data from a single RGB-D camera [50], improve the accuracy of the Kinect SDK tracking [49], and filter the output provided by the Kinect SDK to avoid inconsistent estimations of the limb lengths [47].

The *Unscented Kalman Filter* (UKF) and *particle filter* (PF) are alternative methods to the EKF, which provide a more accurate estimation of the distribution of the state random variable through sampling techniques [154–156]. Larsen et al. [51] used UKF to denoise skeletons provided by a Kinect camera, and reported that UKF yielded more precise results at a lower computational cost than PF. Vartiainen et al. [53] demonstrated that UKF can track instant changes more precisely than EKF in marker-based mocap. Agarwal et al. [52] estimated lower limb dynamics during gait using an RGB camera with UKF. Musunuri et al. [57] showed that UKF was more effective than EKF in reducing nonlinear temporal noise in Kinect RGB-D mocap. Gomes et al. [58] proposed a UKF-based approach for gap-filling in corrupted motion data generated by a marker-based mocap system. Martini et al. [59] proposed a three-step filtering pipeline comprising a spatial node, a temporal node, and a particle filter. The spatial node assesses prediction reliability based on joint relationships, while the temporal node tracks individuals using joint positions and reliability scores. The particle filter combines this information to handle occlusions and ensure smooth predictions.

The *Tobit Kalman Filter* (TKF) is another variation of KF that enables the estimation of the system state even in the presence of missing or noisy measurements [157]. Loumponias et al. [55,56] applied TKF to recover sequences from a markerless mocap system, where occluded keypoints were considered censored measurements. They claim that, compared to EKF and UKF, TKF is more accurate when, due to occlusions, the skeleton is partial.

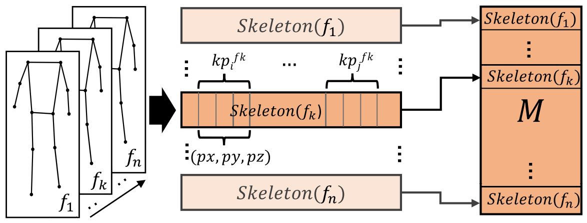

Fig. 6. Matrix representation of human motion in a n-frames temporal window.

6. Dimensionality reduction

In data analysis or data mining, an *attribute* refers to a characteristic or feature of a system that is measured for some *observations* (records) and can vary from one observation to another. These attributes and observations are typically represented in matrices, with the columns representing the attributes and the rows representing the observations, forming a *dataset*. When the number of attributes in the dataset is large, it is referred to as *high-dimensional* [158]. Dimensionality reduction (DR) represents such high-dimensional data in a lower-dimensional subspace to capture the "essence" of the data and separate it from noise.

In human motion refinement, DR methods are applied to solve denoising and completion issues. Fig. 6 shows the representation of human motion in a temporal window of n frames as a matrix M, where each row stores the 3D coordinates of all human keypoints for a frame. Using this notation, each column represents the motion in 3D of a single keypoint throughout the entire temporal window.

As body segments are connected through physical articulations and the motor cortex utilizes motion synergies extensively, human

Table 4
Comparison of the dimensionality reduction methods.

Dimensionality reduction method	Advantages	Disadvantages
Truncated SVD/PCA	◦ Simple implementation	◦ Only captures dominant trends ◦ Fails with non-linear patterns
LRMC	◦ Suited for filling gaps	◦ Works only with clean data
Noisy LRMC/Robust PCA	◦ Handles both gaps and noise ◦ Suited for sparse errors	◦ More computationally expensive
NNMF	◦ Generates interpretable decompositions	◦ May be less accurate
Dictionary learning	◦ Runs in real-time	◦ Needs large training dataset

motion can be efficiently represented in lower dimensions [159–161]. Consequently, human motion data, i.e., keypoint positions over time, is highly correlated and can be approximated using low-rank matrices [60]. This approximation, with substantially fewer rows or columns than the original matrix, captures the essence of human motion. Temporal correlation (e.g., repetitive movement patterns) is identified as *temporal properties*, whereas kinematic synergies of the human body are represented by the correlation between data in columns and are identified as *spatial properties* (e.g., when the hand reaches a target, the elbow and shoulder keypoints move in the same direction). Table 4 shows a comparison between all dimensionality-reduction methods.

6.1. Truncated singular value decomposition

To extract the essential information from the motion matrix and remove noise, the matrix is decomposed into three sub-matrices using the singular value decomposition (SVD) method [162]. Given the motion matrix $M \in \mathbb{R}^{n\times 3m}$ (built as shown in Fig. 6), the SVD of M is:

$$M = U\Sigma V^T \tag{15}$$

where $U \in \mathbb{R}^{n\times n}$ and $V \in \mathbb{R}^{3m\times 3m}$ are orthonormal matrices that contain information about the spatial and temporal properties among keypoints, respectively. The information is represented through the eigenvectors of $M^T M$, which are ordered by importance along the columns in U and V^T, respectively. $\Sigma \in \mathbb{R}^{n\times 3m}$ represents a diagonal scaling matrix, in which the values are the weights (ordered by importance) of the corresponding property.

Starting from the sub-matrices, the original matrix M can be reconstructed as follows:

$$M = \sigma_1 u_1 v_1^T + \sigma_2 u_2 v_2^T + \cdots + \sigma_{3m} u_{3m} v_{3m}^T \tag{16}$$

where σ_i represents the ith element of the Σ diagonal, u_i is the ith column of U, and v_i^T is the ith column of V^T. Eq. (16) represents the motion data where the terms are ordered from left to right by importance (i.e., $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_{3m} \in \Sigma$). Generally, SVD provides a hierarchical representation of the human motion data determined by dominant correlations. It provides a numerically stable matrix decomposition that is guaranteed to exist.

The leftmost vectors of U and V^T contain the principal basis describing the motion matrix, while the rightmost vectors contain minor trends and uncorrelated signals that typically originate from measurement noise. The relevance of the vector pair $u_i v_i$ is determined through the diagonal element $\sigma_i \in \Sigma$, which represents the weight in Eq. (16).

Truncation aims to approximate the observation matrix M with another matrix $\tilde{M}$ of rank r as follows:

$$\tilde{M} = \sigma_1 u_1 v_1^T + \cdots + \sigma_r u_r v_r^T \qquad r < 3\ m \tag{17}$$

where the user defines r to split the main motion trends and noise. The threshold rank r is lowered as the data becomes more noisy. Fig. 7 provides a graphical overview of the truncation process

The effectiveness of this approach is mainly motivated by the *Eckart–Young Theorem* [163]. According to this theorem, the best possible low-rank approximation of matrix M, obtained by minimizing the Frobenius norm of the difference between M and the approximated matrix $\tilde{M}$, is given by the truncated SVD of M. Lai et al. [60] applied the truncated SVD approach to denoise a marker-based mocap data and minimize the error of a motion sequence. Although we only found one reference for this method, it is a starting point for many approaches reported below.

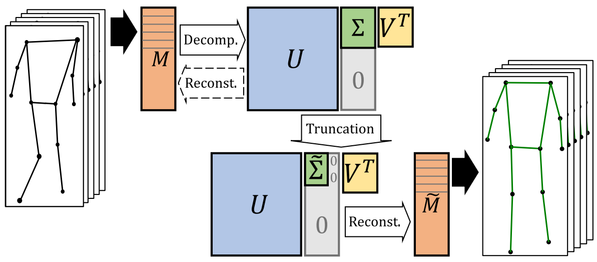

Fig. 7. Overview of decomposition, truncation, and reconstruction phases of dimensionality reduction applied to human motion refinement: sequences of skeletons are stored in the matrix M and decomposed into matrices U, Σ, V^T. The matrix $\tilde{\Sigma}$ is obtained zeroing the rightmost columns of Σ.

6.2. Principal Component Analysis

Principal component Analysis (PCA) is a technique used to extract the most statistically relevant elements of high-dimensional data by utilizing linear correlation between columns.

In the case of human motion data, given a motion matrix $M \in \mathbb{R}^{n\times 3m}$, where n represents the number of frames and m represents the number of keypoints, PCA methods first calculate the *mean motion matrix* $\bar{M} \in \mathbb{R}^{n\times 3m}$:

$$\bar{M} = \begin{bmatrix} \bar{\mu}_1 \\ \vdots \\ \bar{\mu}_n \end{bmatrix} [1 \dots 1]_{1\times 3m} \quad s.t. \quad \bar{\mu}_i = \frac{1}{3m}\sum_{j=1}^{3m} M_{ij} \tag{18}$$

where $\bar{\mu}_i \in \mathbb{R}$ represents the mean value computed along row i. The mean-centered motion matrix $B \in \mathbb{R}^{n\times 3m}$ is obtained by a simple subtraction:

$$B = M - \bar{M} \tag{19}$$

The principal components (PCs) matrix $\Lambda \in \mathbb{R}^{n\times 3m}$ of B is obtained through SVD over B:

$$B = U\Sigma V^T = \Lambda V^T \rightarrow \Lambda = BV \tag{20}$$

where $V \in \mathbb{R}^{3m\times 3m}$ is a matrix whose columns are the eigenvectors of $B^T B$. V represents an orthonormal transform that projects the data matrix B onto its principal axes, where the first principal axes represent the largest data variability. Reversing Eq. (20), the original data matrix B can be reconstructed as a linear combination of PCs, using the matrix V^T:

$$B = \Lambda V^T \tag{21}$$

Federolf et al. [64] utilized the correlation within motion data to address the problem of missing markers by recovering principal

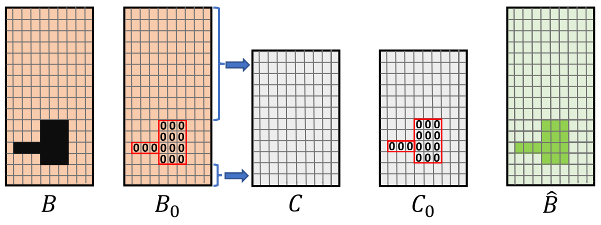

Fig. 8. Representation of the matrices used to perform sequence completion: B is the incomplete motion matrix, B_0 is obtained zeroing all missing values of B, C is obtained maintaining only complete entries of B, and C_0 is obtained zeroing elements of C were the missing values in B are located. $\hat{B}$ is the recovered matrix, whose elements are used to fill the gaps of B.

components from a subset of uncorrupted sequences. Given a mean-centered corrupted motion matrix $B \in \mathbb{R}^{n\times 3m}$, they build matrix $B_0 \in \mathbb{R}^{n\times 3m}$ by zeroing all missing values.

In the first step, they created a matrix $C \in \mathbb{R}^{l\times 3m}$ from the portion of B where all keypoints were available, and matrix $C_0 \in \mathbb{R}^{l\times 3m}$ was built to mimic the corruption pattern of B. Fig. 8 visually represents the involved matrices. Finally, $\hat{B} \in \mathbb{R}^{n\times 3m}$ is the unknown matrix used to fill in the gaps of B.

In the second step, they calculate PCA on B_0, C, C_0:

$$B_0 = \Lambda_0 V_0^T \quad C = \Lambda_C V_C^T \quad C_0 = \Lambda_{C0} V_{C0}^T \tag{22}$$

The eigenvectors matrix V^T is then calculated, which be used to reconstruct C_0 from the uncorrupted PCs of C:

$$C_0 = \Lambda_{C0} V_{C0}^T = \Lambda_C V^T \tag{23}$$

At this point, the method finds the transform $T \in \mathbb{R}^{3m\times 3m}$ between corrupted and uncorrupted eigenvectors V_{C0}^T and V^T:

$$T V_{C0}^T = V^T \tag{24}$$

The method is based on the fact that if T describes a transformation between eigenvectors, then T^{-1} describes the transform between PCs as follows:

$$C_0 = \Lambda_C V^T \qquad \text{from (23)} \tag{25}$$

$$\Lambda_{C0} V_{C0}^T = \Lambda_C T V_{C0}^T \qquad \text{from (22), (24)} \tag{26}$$

$$\Lambda_{C0} = \Lambda_C T \tag{27}$$

$$\Lambda_{C0} T^{-1} = \Lambda_C \tag{28}$$

The computation of T is based on eigenvectors rather than directly on PCs. This is because the dimensions of PCs differ between B_0 to C_0, while the size of the eigenvectors matrix depends only on the number of keypoints. According to this method, the transform T^{-1} between corrupted and uncorrupted PCs is computed by considering the emulated gap (from C to C_0) and is used to correct the real gaps (from $\hat{B}$ to B_0). In particular, it is used to correct corrupted PC of B_0 as follows:

$$\Lambda_{B_0} T^{-1} = \hat{\Lambda} \tag{29}$$

The recovered matrix $\hat{B}$ is estimated as:

$$\hat{B} = \hat{\Lambda} \hat{V}^T \tag{30}$$

$$= \hat{\Lambda} V_C^T \qquad \text{assuming } \hat{V}^T \approx V_C^T \tag{31}$$

$$= \Lambda_0 T^{-1} V_C^T \qquad \text{from (29)} \tag{32}$$

$$= B_0 V_0 T^{-1} V_C^T \qquad \text{from (22)} \tag{33}$$

In [65], the authors propose extending the method described above, where the number of PC-vectors is selected by setting a threshold on the cumulative sum of normalized singular values. This helps to remove noise and preserve valuable information. The effectiveness of the PCA-based approach was tested on various gait datasets with information gaps of up to 70% of the frames. The authors claimed that accuracy only declines when applied to motion sequences with unpredictable movement patterns.

Liu et al. [61] proposed a strategy for recovering different sets of markers across a moderate-to-long period using PCA. They claimed that their method can reconstruct believable movements even when the number of missing markers reaches 50%

Lou et al. [29] implemented a motion denoising technique that uses PCA to learn several motion bases from a dataset and corrects the input with those bases. They claimed that their system obtains good results in filtering marker-based mocap data containing a percentage of outliers below 15%. However, filtering noisy input with completely different motion patterns decreases the quality of the motion.

Shum et al. [63] proposed a real-time PCA-based method to correct skeletons from Kinect SDK. They created a motion database from sequences dimensionally reduced using PCA, and the algorithm corrects corrupted skeletons through such a motion database.

Li et al. [66] presented a PCA-based method to solve the missing keypoint problem. They applied PCA to an entire *training* dataset and stored the results. Once the PCs were obtained, given an incomplete motion sequence M, the algorithm first found the complete matrix M' with the information obtained from the PCA of the complete dataset and then filled the unknown values of M with the estimated information of M'. They found that this method outperformed other state-of-the-art completion methods, such as linear and spline interpolation.

To overcome the limitations of PCA on denoising non-linearly correlated human motion, Tangkuampien et al. [62] used kernel-PCA as a non-linear denoising technique. Kernel-PCA is an extension of PCA, where data is non-linearly mapped via a semi-positive definite kernel function to a feature space before applying linear PCA. As shown in their experiments, kernel techniques with filtering based on greedy algorithms outperform linear PCA in the refinement of Gaussian noise for marker-based mocap data at a very high computational cost.

6.3. Low-rank Matrix Completion

Matrix completion is the process of filling in missing values in a matrix. Low-rank matrix completion (LRMC) is completing an incomplete matrix such that the resulting matrix has a low rank. Given $M \in \mathbb{R}^{n\times 3m}$ motion matrix with gaps (i.e., missing information), LRMC involves finding:

$$\min_{\tilde{M}} rank(\tilde{M}) \quad s.t. \quad \tilde{M}_{i,j} = M_{i,j}, \forall\, (i,j) \in \Omega \tag{34}$$

where $\tilde{M} \in \mathbb{R}^{n\times 3m}$ is the recovered complete matrix, d is the number of joints, and n is the number of frames.

To accomplish this, a logic matrix $\Omega \in \mathbb{R}^{n\times 3m}$ is created to represent the known elements of M (set to 1) and unknown elements (set to 0). LRMC is an effective solution for missing marker issues, as it preserves the spatial–temporal properties of human motion [70]. However, the method has limitations. For example, if an entire row or column is missing, the matrix cannot be recovered without additional information. Additionally, LRMC formulated as in Eq. (34) has the drawback of NP-Hardness in the *rank* function.

To address this issue, *singular value thresholding* (SVT) is an iterative algorithm that approximates the LRMC problem by minimizing the nuclear norm, which replaces the original *rank* function. The SVT problem can be formulated as follows:

$$\min_{\tilde{M}} \|\tilde{M}\|_* \quad s.t. \quad \tilde{M}_{i,j} = M_{i,j}, \forall\, (i,j) \in \Omega \tag{35}$$

where $\|M\|_*$ is the nuclear norm for approximating the rank of M [164].

Lai et al. [60] applied SVT to reconstruct missing joints of a marker-based mocap. They showed that human motion can be effectively recovered starting from a sequence with up to 50% of missing data. Cui et al. [70] proposed an SVT-based method to solve the missing keypoint problem. They showed their method can obtain better accuracy than recovery algorithms implementing full connection neural networks and long-short memory networks.

Tan et al. [67,68] applied SVT to solve the missing keypoint problem, specifically when an entire column or row is missing, including skeleton constraints (i.e., inter-joint distances) in the SVT definition.

Mohaoui et al. [71] solved the LRMC problem through Canonical Polyadic (CP) decomposition. The CP decomposition generalizes the Singular Value Decomposition (SVD) from matrices to higher-order tensors. While SVD expresses a matrix as a sum of rank-one matrices, CP decomposition represents a tensor as a sum of rank-one tensors, each formed by the outer product of vectors from each dimension. Unlike SVD, CP decomposition has no closed-form solution and is typically computed using iterative optimization methods. In this context, the 2D incomplete motion matrix $M \in \mathbb{R}^{m\times 3n}$ is reshaped into the 3D tensor $M' \in \mathbb{R}^{m\times n\times 3}$. Their completion algorithm follows a two-step iterative process. First, it estimates the CP decomposition of M'. Then, using this decomposition, it fills in the missing entries of the tensor.

6.4. Noisy low-rank matrix completion

Noisy low-rank matrix completion (NLRMC) aims to recover a complete low-rank matrix from a noisy observation. NLRMC operates similarly to LRMC, with the additional assumption that the known values are corrupted by noise. The problem of Eq. (34) is modified as follows:

$$\min_{\tilde{M}} rank(\tilde{M}) \quad s.t. \quad \|\pi_\Omega(\tilde{M} - M)\|_F \leq \delta \tag{36}$$

where $\|\cdot\|_F$ is the Frobenius norm, π_Ω is a linear operator that keeps the entries in Ω unchanged zeroing the others, and δ is the noise coefficient [165].

SVT can solve the NLRMC problem by truncating the nuclear norm of M from Eq. (35) through a threshold λ:

$$\min_{\tilde{M}} \lambda\|\tilde{M}\|_* \quad s.t. \quad \tilde{M}_{i,j} = M_{i,j}, \forall\,(i,j) \in \Omega \tag{37}$$

An alternative to SVT is Augmented Lagrange Multiplier (ALM), widely used to solve constrained optimization problems [166,167].

Starting from a corrupted and incomplete motion matrix $M \in \mathbb{R}^{n\times 3m}$ defined as follows:

$$M = \pi_\Omega(E + Z) \tag{38}$$

where $E \in \mathbb{R}^{n\times 3m}$ is the complete and clean observation matrix and $Z \in \mathbb{R}^{n\times 3m}$ is an additive error matrix, ALM leads to an equivalent expression for (34):

$$\min_{E} \|E\|_* + \lambda|\pi_\Omega(Z)|_1 \quad s.t. \quad M = \pi_\Omega(E + Z) \tag{39}$$

where $\|\cdot\|_1$ is $\ell 1$–norm and λ is a weighting coefficient to balance the effects of the two parts. Bautembach et al. [69] presented a comparative study of algorithms to perform LRMC and NLRMC specifically for marker-less mocap data. Li et al. [72] proposed *BoLeRo*, a method that uses ALM to reconstruct corrupted sequences while adhering to bone length constraints. Feng et al. [73] proposed a variation of ALM called *TSMC* to solve the NLRMC, taking into account the temporal smoothness of human motion. However, TSMC is slow when it comes to recovering long motion sequences. To address this issue, Hu et al. [74] proposed a version of TSMC that avoids using SVD. Xia et al. [77] used ALM to fill gaps in sequences recorded with a marker-based mocap system, incorporating kinematic restrictions like bone length and smoothness to provide realistic results. Similarly, Chen et al. [75] added constraints to preserve the spatial–temporal and structural properties embedded in human motion. Hu et al. [78] designed a subspace clustering technique to solve the NLRMC problem. They divided the recovery of missing keypoint problem from a noisy input into several NLRMC sub-problems through clustering and solved them with ALM.

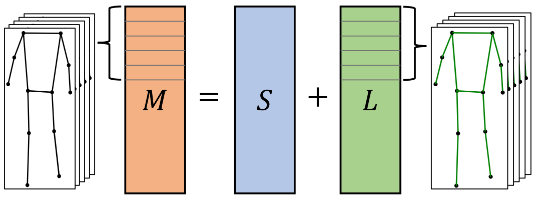

Fig. 9. General structure of motion refinement through RPCA. Noisy motion data is framed as a matrix (M) and then decomposed into sparse matrix S (containing the noise) and low-rank matrix L (containing the clean data).

6.5. Robust Principal Component Analysis

Robust Principal Component Analysis (RPCA) enhances the performance of PCA in the presence of grossly corrupted sequences. The complete observed matrix $M \in \mathbb{R}^{n\times 3m}$ is first decomposed as follows:

$$M = L + S \tag{40}$$

where $L \in \mathbb{R}^{n\times 3m}$ is a low-rank matrix and $S \in \mathbb{R}^{n\times 3m}$ is a sparse matrix containing noise (see Fig. 9).

Following the RPCA notation, classic PCA is defined as follows:

$$\min_{\tilde{L}} \|M - \tilde{L}\|_2 \quad s.t. \quad rank(\tilde{L}) \leq k \tag{41}$$

where k is a positive constant and $\|\cdot\|_2$ denotes the ℓ_2-norm, in this case the largest singular value of $S = M - L$ matrix.

If the noise S is not independent and identically distributed, such as a corrupted entry in M, it can result in an estimated matrix, $\tilde{L}$, that is significantly different from the true L [168]. To overcome this limitation, the RPCA problem is formulated as follows:

$$\min_{L,S} \; rank(L) + \|S\|_0 \quad s.t. \quad M = S + L \tag{42}$$

where both $rank(\cdot)$ and the ℓ_0-norm $\|\cdot\|_0$ terms are non-convex. The problem in Eq. (42), also known as *Principal Component Pursuit*, is relaxed as follows:

$$\min_{L,S} \; \|L\|_* + \lambda\|S\|_1 \quad s.t. \quad M = S + L \tag{43}$$

ALM can efficiently solve the RPCA problem by modifying the linear operator π_Ω in Eq. (39) such that $\pi_\Omega(A) = A$. Wang et al. [80] proposed an RPCA-based method for reconstructing and denoising keypoints from Kinect SDK, approximated using ALM. They also addressed the sub-problem of bone length consistency in [81], where a tree structure describes the skeleton, with each node representing a joint and each edge representing a bone. In [83], the method was tested on both marker-based and marker-less mocap data, and the authors found that the primary limitation of the RPCA decomposition is the unbalanced observation matrix, which reduces the efficiency of rank minimization in capturing the matrix global information. They used Hankel-like augmentation on the matrix to address this issue while maintaining the rank in [82]. Raj et al. [87] introduced two additional constraints C_{tm}, C_{ph} to the objective function of Eq. (42):

$$\min_{L,S} \; rank(L) + \psi_1\|S\|_0 + \frac{\psi_2}{2}C_{tm}(L) + \frac{\psi_3}{2}C_{ph}(L) \tag{44}$$

where ψ_1, ψ_2, ψ_3 are regularization terms. The trajectory movement function $C_{tm}(\cdot)$ calculates the difference between the adjacent rows of the L matrix. Minimizing the Frobenius norm of such a matrix improves the smoothness of the node trajectories. The pair-wise hierarchical function $C_{ph}(\cdot)$ calculates the distance between each parent–child node pair in the skeletal representation. Minimizing the result of $C_{ph}(L)$ limits the undesired drifting of nodes in a frame during recovery.

In [169] Eq. (43) is solved using the accelerated proximal gradient (APG) method, which is a fast iterative approximation algorithm also used in NLRMC. Liu et al. [79] separated the human skeleton into several sub-matrices, assuming each body segment shares the same low-dimensional subspace representation. They used APG to find a complete low-rank matrix approximation from each noisy sub-matrix. In [84–86], alternatives to APG and ALM for solving Eq. (43) are presented, specifically for human motion refinement.

6.6. Non-negative Matrix Factorization

Non-negative Matrix Factorization (NNMF) [170] is an alternative approach to PCA in which both data and components are assumed to be non-negative. Although NNMF has been widely used on images, it applies to many problems. Given a motion matrix $M \in \mathbb{R}_+^{n\times 3m}$, NNMF approximates M to have rank r with two matrices $W \in \mathbb{R}_+^{n\times r}$ and $H \in \mathbb{R}_+^{r\times 3m}$ such that:

$$\min_{W,H} \|WH - M\|_F^2 \quad s.t. \quad W \geq 0 \wedge H \geq 0 \tag{45}$$

The limits imposed on the matrix factors W and H distinguish NNMF from PCA. While PCA requires the columns of W to be orthonormal and the rows of H to be orthogonal to each other, NNMF only requires that all three matrices be positive [171]. In [88], Peng et al. modeled the human skeleton into five low-rank sub-matrices and solved the missing keypoint problem by adapting Eq. (45) using NNMF as follows:

$$\min_{W,H} \|\pi_\Omega(WH - M)\|_F^2 \quad s.t. \quad W \geq 0 \wedge H \geq 0 \tag{46}$$

They quantitatively showed that their NNMF-based method outperforms SVT-based methods [60,68].

6.7. Dictionary Learning

Dictionary Learning (DL) is a machine learning approach that creates a sparse representation of the input data through a linear combination of basic elements. These elements are known as *atoms* and are stored in a *dictionary*.

Given $M \in \mathbb{R}^{n\times 3m}$ motion matrix, methods based on DL minimize the reconstruction error as follows:

$$\min_{D,W} \|M - DW\|_F^2 \quad s.t. \quad \forall i \in [1,n] : \|W_i\|_0 \leq k \tag{47}$$

where $D \in \mathbb{R}^{3\ m\times p}$ is the dictionary, $W \in \mathbb{R}^{n\times p}$ is the sparse weight matrix, and k is the target sparsity threshold for each row of W. If the size p of D is less than the training set samples n, then the dictionary is called *undercomplete*. Dictionaries are typically built using efficient methods such as MOD [172] and K-SVD [173]. The DL approach allows the problem of human motion denoising to be mapped into a general minimization problem. Denoising involves learning motion dictionaries from various clean actions and, at runtime, automatically selecting the best-correlated subset of motion bases to reconstruct the clean motion. Fig. 10 gives an overview of the whole process. Once the dictionary $D \in \mathbb{R}^{3m\times p}$ is built, given $m \in \mathbb{R}^{3m}$ noisy skeleton, the weight vector $w \in \mathbb{R}^p$ can be found as:

$$\min_{w} \|m - Dw\|_F^2 \quad s.t. \quad \|w\|_0 \leq k \tag{48}$$

The approximated vector $\tilde{m} \in \mathbb{R}^{3m}$ is obtained as follows:

$$\tilde{m} = Dw \tag{49}$$

The non-convex minimization problem of Eq. (48) can be approximated with two different norms, depending on the noise type of the corrupted sequence. Xiao et al. [89] claim that a ℓ_2/ℓ_1 denoising model best applies with Gaussian noise:

$$\min_{w} \|m - Dw\|_2^2 + \lambda\|w\|_1 \tag{50}$$

In contrast, a ℓ_1-only denoising model is most suited in case of outlier keypoints:

$$\min_{w} \|m - Dw\|_1 + \lambda\|w\|_1 \tag{51}$$

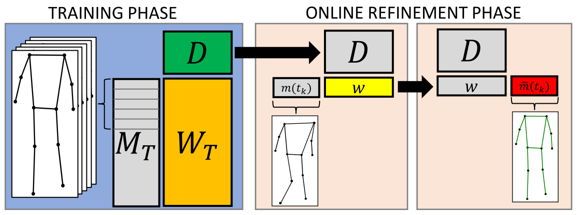

Fig. 10. In refinement through DL, the training phase involves building the dictionary D and the weights W_T from the training set M_T. In the online denoising phase, the vector of weights w that best approximates the corrupted keypoint vector $m(t_k)$ is found using the dictionary D. The weight vector w and dictionary D are then used to compose a clean motion vector $\tilde{m}$.

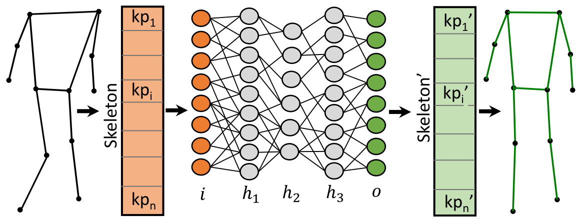

Fig. 11. Structure of a neural network: Input layer (i), hidden layers ($h_1,\dots,h_3$), and output layer (o).

The authors suggest combining the two filters sequentially in the presence of both Gaussian noise and outliers.

Feng et al. [90] divided the human body in parts and then refined each sub-chain through DL. They collected an enormous training dataset and then normalized each coordinate. They trained multiple motion dictionaries, one per body part. After experimental analysis of marker-based mocap sequences, the authors concluded that their DL-based method yields better performance with more stable bone lengths than methods based on KF and LS.

Mei et al. [93] refined, through DL, the corrupted sequences acquired using a monocular 3D HPE [174]. Unlike the other approaches, they trained multiple small dictionaries so that each skeleton is represented by the dictionary with the smallest reconstruction error. They developed a weight mechanism to leverage the temporal smoothness to reconstruct consequent skeletons with similar bases.

Wang et al. [94] refined the outcome of monocular 3D HPEs using DL. After a normalization step, they transform the skeleton into a distance matrix by computing the distances between all pairs of joints. The algorithm takes the distance matrix as input and reconstructs it as a linear combination of bases obtained during the training phase. The reconstructed skeleton is then scaled, aligned, rotated, and translated to match the position and orientation of the initial corrupted skeleton.

Xia et al. [91] demonstrated how to recover a *partial* motion data sequence using DL, which requires a slightly different approach than the standard denoising problem. During runtime, when a subset of keypoints is missing, they extrapolate the weight vector w (originally obtained through Eq. (48)) as follows:

$$\min_{w} \|\Psi(m) - \Psi(D^T)^T w\|_F^2 \quad s.t. \quad \|w\|_0 \leq k \tag{52}$$

where $\Psi(\cdot)$ is a function that maintains only the columns that are not missing in m. Once obtained w, missing keypoints of m are estimated by applying Eq. (49).

7. Deep neural network

This class encompasses methods that leverage deep learning and neural networks to refine human motion data. A neural network is a computational model inspired by the structure of the biological human brain that attempts to identify relationships in a set of data [175]. It

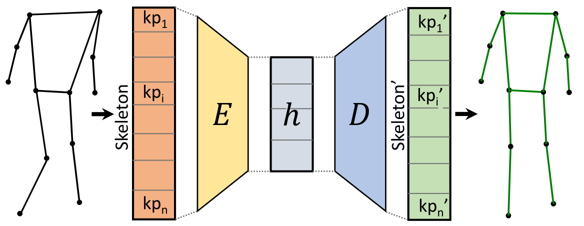

Fig. 12. General structure of an autoencoder. From left to right: input (m), encoder (E), latent space (h), decoder (D), and refined output (m').

consists of an *input layer* (i), one or more *hidden layers* (h), and an *output layer* (o) (see Fig. 11). Each layer processes the inputs through nodes, known as *neurons*. A neuron weights the corresponding inputs based on the task the model attempts to learn and generates an output through a non-linear *activation function* applied to the sum of its inputs. Activation thresholds and weights are set during a training procedure using a *loss function* [176]. Such a cost function is a mathematical function that condenses a complex system's important features into a single scalar value, making it easier to compare and evaluate different results [177].

In recent years, deep learning has been used to refine motion data, with a focus on a particular type of neural network known as autoencoders (AE) (see Fig. 12). AEs are trained to reproduce a given input, typically a set of keypoints that form a human skeleton, with high accuracy. The key idea is to leverage the internal layers of the AE to capture the essential features of human poses while discarding noisy information.

Starting with the input data, which is a set of keypoints representing a $skeleton$ (sk), Autoencoders (AEs) are trained to *accurately reproduce* the input, $skeleton'$ (sk'). The main idea behind using AEs is to utilize the implicit characteristics of internal layers to capture the essential features of human poses while filtering out the noise. Fig. 12 shows the structure of an AE neural network, which consists of encoding layers (E), a hidden layer (h), also known as the *latent space*, and decoding layers (D). During the training phase, the network minimizes a certain loss function, L:

$$L(sk, D(E(sk))) \tag{53}$$

A common cost function used in this context is the *mean squared error*, which penalizes the difference between the input skeleton sk and the output $sk' = D(E(sk))$. There are two main architectural models of AE proposed in the literature: *Feed-forward neural networks* (FFNN) and *recurrent neural networks* (RNN). FFNNs connect the input layer to the output layer through activation functions without forming cycles [175]. The hidden space h at frame k in a FFNN is defined only by the current input:

$$h(k) = f(sk[k], \theta) \tag{54}$$

where θ is the set of weights of the network.

RNN architectures are characterized by feedback connections [176]. In contrast to Eq. (54), the hidden space h at frame k in an RNN is defined not only by the current input but also by the previous state:

$$h(k) = f(h[k-1], sk[k], \theta) \tag{55}$$

Hidden layers implement self-interaction to support time delays and feedback loops. Table 5 shows a comparison between all deep neural network methods.

7.1. Undercomplete autoencoder

An AE can be considered *undercomplete* when its latent space h is smaller than the input skeleton sk. This approach forces the AE to capture the most relevant features of the training data and implicitly performs dimensionality reduction. Undercomplete AE is a generalization of the linear dimensionality reduction method presented in Section 6. When the decoder is linear and the loss function is the mean squared error, an undercomplete AE learns to span the same subspace as PCA [178]. However, unlike PCA, AE allows the model to learn more meaningful generalizations of human motions *non-linear* functions [175]. Bütepage et al. [95] implemented three types of undercomplete AE to recover information about multiple missing keypoints, where they set all joints belonging to the same limb to zero to simulate occlusions. They claimed that all their AE models effectively inferred the missing keypoints with no significant deviation from the ground truth. Xia et al. [96] proposed an undercomplete AE with an additional layer after the encoder, called *local self-expression layer*, to maintain the representation correlations between consecutive skeletons for the same problem. Yuhai et al. [97] proposed U-Bi-LSTM, an undercomplete AE designed explicitly for motion completion, that leverages the underlying structure and temporal dependencies in the motion data through its compressed representation in the bottleneck hidden layer.

7.2. Denoising autoencoder

A *denoising* autoencoder (DAE) is an AE that receives a corrupted skeleton as input and is trained to predict the original *uncorrupted* skeleton. The training phase relies on the minimization of the following loss function L:

$$L(sk, D(E(\xi(sk)))) \tag{56}$$

where $\xi(\cdot)$ is a function that corrupts the input data with a specific type of noise. As a result, rather than simply copying the input skeleton, DAEs are trained to cancel out noise. This is done by implicitly learning the *structure* of sk.

Holden et al. [99] proposed a FFNN DAE to reconstruct the skeleton motion from a marker-less mocap. They showed the method's validity to fix corrupted sequences of human poses using the skeleton information. They used an RGB-D camera and an inertia-based mocap as ground truth. They compared the recovered data with the results obtained by applying PCA and found that AE performs better than PCA in the presence of different motion patterns in the same sequence.

Mall et al. [100] proposed a RNN AE for marker-based mocap data refinement. The RNN architecture exploits temporal relationships between subsequent frames, by processing the input vector with a *look-back* and *look-ahead window*. Although the neural network is originally designed to denoise human motion, the authors demonstrate that gap filling is also possible. In this approach, the missing values are filled with linearly interpolated values, which the RNN then corrects to generate the complete sequence of values. The authors claimed that each frame may be denoised in less than $1\,\mathrm{ms}$, excluding the delay for getting the look-ahead frames.

Kucherenko et al. [98] proposed two different NN architectures, one LSTM-based RNN and one FFNN to denoise and reconstruct corrupted sequences captured with a marker-based mocap. At the training time, their function $\xi(\cdot)$ of Eq. (56) injects additive Gaussian white noise to the network's input. Li et al. [101] proposed a Bidirectional Recurrent Autoencoder (BRA) based on LSTM to learn spatiotemporal patterns and refine noisy sequences. They claimed that their NN can handle a wide variety of noise types at a low time cost. In [102], they proposed another type of BRA that maintains bone-length consistency by keeping the distance between keypoints that are naturally connected by a bone (called *bone-length consistency*). Ji et al. [103] proposed a RNN to estimate the missing keypoint position from sequences captured with a marker-based mocap. Zhu [104] designed a RNN to detect noise and recover a noisy sequence with gaps. Kaufmann et al. [105] proposed a DAE to denoise marker-based mocap sequences. In removing Gaussian noise from the inputs, the model in [99] outperforms the model proposed in [105]. On the other hand, the model in [105] outperforms the model in [99] by far when filling in randomly dropped joints.

Table 5
Comparison of the deep neural network methods.

Deep neural Network method	Advantages	Disadvantages
Undercomplete autoencoder	∘ Good generalization ∘ Interpretable latent space	∘ Cannot reconstruct fine details
Denoising autoencoder	∘ Explicitly trained for noise removal	∘ May underperform in motion completion
Variational autoencoder	∘ Suited for modeling uncertainty	∘ More complex to train ∘ Less interpretable latent space
Graph neural network	∘ Better at capturing spatial relationships	∘ Requires accurate graph modeling
Transformer	∘ Suited for long-term temporal patterns	∘ Requires larger training sets
DDPM	∘ Accurate under extreme corruption ∘ Strong generative capacity	∘ Highest computational cost

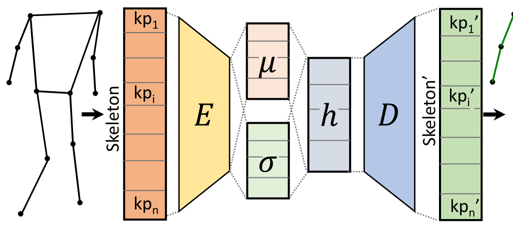

Fig. 13. General structure of a Variational Autoencoder. From left to right: input, encoder (E), mean hidden space (μ), std. dev. hidden space (σ), sampled latent space (h), decoder (D), and refined output.

Lohit et al. [106] predicted the position of missing keypoints online, from both a marker-based mocap and Kinect RGB-D camera. Li et al. [108] proposed a RNN based on LSTM that considers the bone length constraints to recover and denoise sequences acquired with a marker-based mocap. Zeng et al. [107] proposed *SmoothNet*, a FFNN that improves both temporal smoothness and precision simultaneously on 2D HPE, 3D HPE, and SMPL [179]. Zhu et al. [109] proposed a RNN DAE to solve the missing keypoint problem with marker-based mocap sequences. In [110], they implemented an additional loss function based on bone-length consistency to improve the naturalness of the result. In [111,112], they integrated *LSTNet* [180], an LSTM placed between the output of the encoder and the hidden space. After an experimental evaluation of the CMU dataset corrupted with synthetic noise, they claimed that integrating LSTNet helps the DAE to capture long-term features better.

The *Masked autoencoder* is a type of DAE where parts of the input data are replaced with a specific value before being fed into the network. This AE aims to reconstruct the original, unmasked input data from this partially observed input. They are typically used to train Natural Language Processing models (e.g., BERT [181]) that predict masked words based on the context provided by the surrounding ones.

7.3. Variational autoencoder

A variational autoencoder (VAE) is a widespread AE variant in which the encoder learns a probability distribution over the latent space approximating a normal distribution. Fig. 13 shows an overview of the VAE structure. The encoder generates two vectors, one for the mean (μ) and one for the variance (σ), representing a Gaussian distribution over the latent space. The latent code h is then derived from the Gaussian distribution:

$$h = \mu + \sigma \tag{57}$$

The decoder transforms the latent code back to the original input space.

Nakatsuka and Komorita [17] implemented a VAE to refine 3D HPE data from low-resolution images. During training, they first fed the VAE with ground truth data and then added three types of noise: left–right keypoint switching, jittering keypoints, and dropping frames. They claim their model could eliminate jitters and produce smooth movements in real-time, even under large and complex noise-intensity conditions. Chen et al. [113] developed a constrained VAE to denoise motion sequences. They tested the VAE, adding different variances of Gaussian noises to the ground truth of Human3.6M [21]. Fiche et al. [114] proposed Motion-DVAE, a VAE that leverages motion priors to capture the short-term spatiotemporal dependencies. They split the computation into two stages: the learning stage, where an unsupervised framework estimates the parameters of the inference model and then the regression stage.

7.4. Graph neural network

The *Graph Neural Networks* (GNN) is a versatile class of NN designed to operate on graph-structured data, where entities and their relationships are represented as nodes and edges. This representation enables GNNs to capture complex dependencies and interactions in the data effectively.

Nguyen et al. [117] proposed TA-WLS, a method based on GNN to complete missing keypoints in marker-based mocap data. The input is a skeleton sequence with missing information, then reconstructed by the model leveraging the graph's spatiotemporal properties. Pan et al. [116] proposed a GNN to recover the position of missing keypoints in marker-based mocap data. They implemented an LSTM to model the relationships between neighbor markers by computing the variances of pairwise distances between all markers. In this way, neighbor markers will have small variances and thus can be used to recover their position in case of missing information. Choi et al. [120] proposed a directed acyclic graph neural network (GNN) for motion refinement. The model follows an encoder–decoder structure, with each component comprising three directed graph network blocks that update node and edge features based on their neighbors. To capture temporal dynamics, the final block of the encoder and the initial block of the decoder are implemented as bidirectional LSTMs.

The *Graph Convolutional Network* (GCN) is a GNN with convolutional layers. GCN learns node representations better by aggregating information from a node's neighbors. Cui et al. [115] proposed a GCN architecture to recover missing keypoints without training. Yin et al. [137] designed a spatial–temporal graph convolutional network (ST-GCN) to reconstruct corrupted motion sequences captured with marker-based mocap. Xu et al. [121] proposed a model based on ST-GCN to recover the motion information from a sequence of missing frames. He et al. [118] introduced CAIR, a two-step filter for motion completion. Given a sequence of partial skeletons in input, they divided the refinement process into two steps: an encoder–decoder framework fills the gaps in the sequence, and then a GCN denoises the recovered values. Lee et al. [119] introduced a DAE based on GCN for motion completion. Specifically, they employed spectral graph convolution to extract key graph features via spectral filtering. Additionally, they incorporated Laplacian smoothing in the encoder and Laplacian sharpening in the decoder.

7.5. Transformer

The Transformer is a type of NN architecture based on the *self-attention mechanism*. It computes scores for each pair of input elements, weights them based on that score, and then aggregates the weighted elements to form new representations. In contrast to RNNs, this mechanism allows entire data sequences to be processed concurrently rather than sequentially.

Cui et al. [122] proposed a bi-directional attention network (BAN) and LSTM to solve the missing keypoint problem. Xu et al. [123] proposed AuxFormer, a transformer-based learning framework that can denoise, complete, and forecast human motion. The network encodes keypoints, timestamps, and masking information, then implements spatial and temporal attention to model all dependencies. Liu et al. [124] proposed a transformer-based method to recover missing keypoints. It inputs a sequence of partial skeletons and reconstructs the missing information by leveraging the temporal and spatial attention mechanisms. In particular, they used two distinct encoders to extract twice the relationships between the same joint at different time steps and between different joints at the same time step. In their test, their model outperformed an FCNN and an LSTM. Chi et al. [125] introduced PORT, a lightweight transformer-based filter to complete missing keypoints from marker-less mocaps. Since HPEs tend to have low confidence under occlusion, their method can also refine those keypoints, masking and reconstructing them. Although it is a data-driven method, they claim it does not require fine-tuning because the global and local relationships between keypoints learned from a dataset are still valid when applied to other datasets. Björkstrand et al. [126] proposed XMAE, a lightweight masked AE, to recover a complete skeleton given a partial and noisy set of keypoints. They implemented a cross-attention mechanism to learn the spatial relationship between keypoints from the training set. Mascaró et al. [127] presented UNIMASK-M, a transformer-based model for motion completion and forecasting. They employed a pose decomposition approach, deconstructing a single skeleton into patches, along with the standard ViT [182] self-attention encoder and decoder block. He et al. [128,129] proposed a spatiotemporal attention-based graph neural network to denoise motion data. Unlike traditional GCNs, they used the attention mechanism to establish direct connections between distant nodes.

7.6. Denoising Diffusion Probabilistic Model

The Denoising Diffusion Probabilistic Model (DDPM) is a class of generative models that produces high-quality data by iteratively adding and removing noise [183]. The process begins by corrupting the data with Gaussian noise. The model is then trained to reverse this process, gradually removing noise from the data to recover the original complex distribution.

Du et al. [130] proposed AGRoL, a DDPM predicting the full-body pose given the position and orientation of only the head and both hands. Although based on DDPMs, renowned for being heavy, they kept the architecture compact to run it in real-time. Yan et al. [131] proposed a DDPM-based architecture for denoising motion data sequences. Unlike standard DDPMs that sample noise from a standard Gaussian distribution, their method samples from a conditional multivariate Gaussian distribution. The conditioning is based on the paired 2D poses and the initial 3D pose estimates produced by a 3D pose estimator. Zhang et al. [132] presented RoHM, a DDPM-based approach to denoise and complete motion sequences. RoHM consists of two sub-tasks: reconstructing global trajectory and predicting local motion using two distinct diffusion models. This decomposition allows each model to focus on the specific motion characteristics, leading to a more effective reconstruction process. Wang et al. [133] proposed a method to refine skeletons from egocentric videos through DDPM. Their approach leverages joint uncertainty, using a diffusion model to regenerate high-uncertainty joints conditioned on the more reliable, low-uncertainty joints. Bozzini et al. [134,135] presented a lightweight algorithm to refine highly corrupted motion data in real-time. The core is based on the Denoising Diffusion Implicit Model (DDIM), a more efficient sampling method for diffusion models. DDIM can generate high-quality samples in significantly fewer steps than DDPM, leading to substantial speedups.

7.7. Other architectures

Skurowski et al. [136] reviewed various neural network architectures to solve the missing keypoint problem. They implemented two types of FFNN and three types of RNN (i.e., LSTM, BILSTM, GRU). The experimental analysis found that RNNs achieve better results for shorter gaps, while FFNNs outperform RNNs for longer gaps.

Wang et al. [138] proposed an *overcomplete AE* based on LSTM. Differently from *undercomplete AE*, this AE has a latent space h with a higher dimension than the input/output data. They called this architecture *Spatio-temporal Recurrent Neural Network (STRNN)* and showed how STRNN predicts long-duration motions and can be used to denoise corrupted sequences. They recovered the original motion from ground truth sequences from two widespread marker-based mocap datasets randomly perturbed with synthetic noise to test the latter.

D'Eusanio et al. [139] proposed RefiNet, a three-block filter, to improve any HPE's precision using an RGB-D camera. While the first and the third blocks fall outside the scope of the survey, in their second block, they implemented an FFNN composed of a sequence of 4 layers that directly regresses the 3D key points' error.

Wang et al. [140] proposed HiMoReNet, a split-and-recombine method to denoise skeleton sequences. The filter divides all joints into five groups (torso and four limbs) and then applies specific layers to each group to capture spatial and temporal relationships. In the experiments, they claim to outperform [107] in precision because the splitting method helps to distinguish motion patterns for different body parts.

Dang et al. [141] proposed MoManifold, an optimization method that uses manifolds distance to denoise motion data. During training, MoManifold learns an independent implicit surface representing plausible acceleration vectors for each body joint in a high-dimensional space. These acceleration manifolds capture the continuity and structure of realistic human motion, enabling the assessment of motion plausibility. At runtime, the distance between a joint's acceleration and its corresponding learned manifold measures how well the motion aligns with natural human dynamics.

8. Hybrids and other approaches

This class includes all filtering contributions that combine two or more approaches belonging to the previous classes to employ the complementary strengths of various methodologies.

8.1. State observer and evolutionary algorithm

State observers are effective in correcting noisy keypoints and estimating their position, even in the presence of occlusions. However, some formulations (e.g., Eq. (9)) do not consider biomechanical constraints. To address this issue, various approaches in the literature combine state observers with *evolutionary algorithms* to incorporate such constraints. Evolutionary algorithms are optimization techniques inspired by the process of biological evolution. *Differential evolutionary* (DE) optimization, in particular, is a class of stochastic search methods that iteratively seeks to improve a candidate solution concerning a specified quality metric [184].

Das et al. [185,186] improved motion data from a marker-less mocap system with KF and then used DE to minimize the bone length variance. They adopted the transition function and state representation of Eq. (12). The optimization is performed building a graph where each joint is a vertex, and each body segment is an edge. DE attempts to decrease the positional change of each vertex while minimizing the variation of edge lengths. The KF state is then updated with the adjusted measurement.

Tripathy et al. [187] implemented a particle filter to retrieve more realistic anthropometric measurements from a marker-less mocap. They implemented an evolutionary algorithm to reduce the bone length variations. Zhou et al. [188] proposed the combination of a Tobit particle filter and a differential evolutionary algorithm to smooth the results of Kinect SDK, keeping the bone lengths constant.

8.2. *Low-dimensional state observer*

The implementation choices that have the greatest impact on the filtering accuracy of a state observer are the underlying dynamical model and the state (as discussed in Section 5). In the case of Eq. (12), each keypoint coordinate is typically treated independently, which leads to a loss of important correlations between coordinates.

Burke and Lasenby [189] used KF to the low-dimensional space obtained with SVD. They first implemented a mean-centered training set $B^{n\times 3m}$ by subtracting mean column $\mu^{n\times 1}$ from each column of a training set $M^{n\times 3m}$, as in Eq. (19). Then, they obtained $V^{3m\times 3m}$ through SVD on B, as in Eq. (15). Finally, they derived matrix $\tilde{V}^{3m\times d}$ by discarding the $3m-d$ least relevant basis functions from V. This matrix was used to project the input skeleton $m^{1\times 3m}$ into its low dimensional state $x^{1\times d}$:

$$x[k] = \tilde{V}^T(m[k] - \mu) \tag{58}$$

The KF consists of a state transition matrix F set to an identity matrix to simulate random walk motion model. The output matrix H is adapted to de-project the low dimensional state into the space of the input skeleton:

$$\tilde{H} = H\tilde{V} \tag{59}$$

The experiments reduced the low-dimensional state from 47 markers to a vector of size 77. They claim that this method achieves compact error distribution thanks to the temporal smoothing and keypoint correlations.

8.3. *State observer and neural network*

Methods based on state observers are typically considered alternatives to methods based on neural networks. State observers rely on a priori knowledge of the system dynamics and aim to estimate the system's state based on measurements of its inputs and outputs. In contrast, neural networks do not require any explicit definition of the system model, and instead, they learn to approximate the system behavior automatically during the training phase. However, different works in the literature showed that combining the two approaches can lead to better results.

Park et al. [190] improved the skeletons extrapolated by a Kinect, by integrating two RNN DAEs with KF. They recorded a dataset with Kinect and a marker-based mocap as ground truth. One RNNs refines the key keypoint positions, while the other refines the keypoint velocities. In both cases, given the input skeletons from the marker-less mocap, they minimize the bias w.r.t. the ground truth. At run-time, the KF exploits the refined skeleton position and velocity to improve the temporal smoothness. Coskun et al. [191] used deep learning to learn the motion model and all the noise parameters. They created three distinct RNNs with different goals: Predicting the new state, estimating the prediction noise, and estimating the measurement noise. The outputs are then used to compute the correction phase of the KF. As a result, they do not need the process and measurement covariance matrices and the transition function in advance. The experimental evaluations, conducted on the Human3.6M dataset [21], show that this method outperforms traditional filtering techniques such as EMA, KF and AE. Lannan et al. [192–194] proposed to filter the latent space of an undercomplete AE to denoise motion data. In [192], they applied traditional techniques, such as SMA and KF, on the compressed latent space. They reconstructed the motion data by decoding the filtered latent space, claiming that the smoothing filter guidance reduces the noise and produces a smoother outcome than the approach in [99]. In [193] and in [194], they developed a specific Tobit KF to smooth the latent space of the AE. They showed that the filter can improve the output from Kinect SDK in keypoint position, joint angles, and bone lengths. Martini et al. [195] proposed FLK, a real-time filter to perform denoising and completion. FLK combines a Kalman filter with a recurrent neural network to learn the underlying human motion model. These two components enable the filter to effectively handle both jitter and dropped frames.

8.4. *Other approaches*

This section collects the literature contributions to human motion refinement that fall outside the proposed taxonomy.

Tree-based regression. Tree-based regression is a machine learning approach that uses decision trees to predict continuous numerical values. Regression trees split the data into regions based on feature values and predict outcomes by averaging the target values within each region. The model recursively partitions the input space by selecting features and thresholds that minimize prediction error. Random forest regression (RFR) is a machine learning technique to predict a value based on a combination of multiple decision trees. Combining the predictions made by each tree in the forest produces the final prediction. Shen et al. [196] proposed a denoising method based on RFR that meets real-time constraints. Their main idea is to convert a denoising problem into a *bias estimation* problem. They trained an RFR function that estimates the offsets between the keypoints provided by Kinect SDK and the actual positions. They also leveraged temporal smoothness by minimizing an energy function. Skurowski et al. [197] evaluated various tree-based regression methods to solve the missing keypoint problem. They compared the results of CART [198], M5P [199], gradient-boosted trees [200], bagging of trees [201], random forests, FFNN [136], spline interpolation, and LRMC algorithm. They found that, with up to 50 frame gaps, spline interpolation outperformed the other methods. For longer gaps, M5P significantly outperforms the other approaches.

Gaussian process regression. Gaussian process regression (GPR) is a probabilistic machine learning method to predict a value by modeling the relationship between the input and target variables as a Gaussian process. Zhou et al. [202] used GPR to reconstruct corrupted skeletons in presence of self-occlusions, introducing also a temporal consistency term to constrain the velocity variations between successive frames. In [203], they created a mixture of Gaussian processes, by dividing the skeleton space in local regions using clustering algorithm. Chiang et al. [204] captured the joint movement traces of different people by using a motion capture system and a Kinect, mapping with GPR the mocap and Kinect data into a standardized domain.

Skeleton database search. Baumann et al. [205] addressed the missing keypoint problem by storing a collection of clean skeletons in an efficient spatial indexing structure (i.e. kd-tree). They normalized the skeleton w.r.t. global position and orientation, by storing velocities and accelerations. In the presence of missing keypoints, similar keypoints positions are retrieved from the database with the nearest neighbor search. Yasin et al. [206] implemented a fast kd-tree algorithm to complete missing keypoints from sequences. They first construct a knowledge base from the existing clean datasets, building a kd-tree. Then, given a missing keypoint, they implemented a search for nearest neighbors from the GPU-based kd-tree. The work of Plantard et al. [207] follows a similar principle to address the denoising problem. They collected a set of prior clean skeletons to optimize the data provided by Kinect SDK. They proposed a structure, called a *Filtered Pose Graph*, to efficiently search the skeletons. When a new skeleton arrives from the mocap, the algorithm evaluates the reliability of each keypoint. Then, it searches the Filtered Pose Graph for the best candidate skeleton by replacing keypoints with low-reliability scores with those found in the database.

9. Evaluation metrics and datasets

This section summarizes the main measurement metrics and datasets used for evaluating the accuracy of a refinement method. All the contributions discussed in this survey employ one or more of these metrics and datasets.

9.1. Evaluation metrics

The *Mean Absolute Error* (MAE) is a commonly used metric for evaluating one-dimensional signals in the machine learning context. In the human motion refinement context, it evaluates the precision of each coordinate of all keypoints. Given the refined positions $\hat{p}$ of a keypoint coordinate and the corresponding ground truth $\bar{p}$, MAE is defined as:

$$\mathrm{MAE}(\hat{p},\bar{p}) = \frac{1}{n}\sum_{k=0}^{n} \mid \hat{p}[k] - \bar{p}[k] \mid \tag{60}$$

where n is the total number of frames of the motion sequence.

The *Root Mean Squared Error* (RMSE is another commonly used metric to evaluate the accuracy of a refinement method in the machine learning context. Similar to MAE, RMSE measures the precision of each coordinate for each keypoint. It is defined as follows:

$$\mathrm{RMSE}(\hat{p},\bar{p}) = \sqrt{\frac{1}{n}\sum_{k=0}^{n}(\hat{p}[k] - \bar{p}[k])^2} \tag{61}$$

Due to the squaring operation, RMSE is more sensitive to outliers. As a consequence, RMSE values are generally higher than MAE values.

The most used metrics to evaluate the accuracy of 3D HPE is the *Mean Per Joint Precision Error* (MPJPE). It measures the average Euclidean distance (ℓ_2-norm) between the predicted and ground truth positions of each joint across all samples, averaged over all joints. Given a refined motion matrix $\hat{M}$ that stores n frames of m keypoints and its corresponding ground truth motion matrix $\bar{M}$, MPJPE is defined as:

$$\mathrm{MPJPE}(\hat{M},\bar{M}) = \frac{1}{n\cdot m}\sum_{k=0}^{n}\sum_{j=0}^{m} \|\hat{M}_{k,j} - \bar{M}_{k,j}\|_2 \tag{62}$$

The *Mean Per Joint Acceleration Error* (Accel) evaluates the smoothness and jitter errors. Given a refined acceleration matrix $\hat{A}$, obtained deriving two times each keypoint of the motion matrix $\hat{M}$, and its corresponding ground truth acceleration matrix $\bar{A}$, *Accel* is defined as:

$$Accel(\hat{A},\bar{A}) = \frac{1}{(n-2)\cdot m}\sum_{k=2}^{n}\sum_{j=0}^{m} \|\hat{A}_{k,j} - \bar{A}_{k,j}\|_2 \tag{63}$$

Besides smoothness, the consistency of bone lengths is another important aspect that characterizes the efficiency of a refining method. Given two keypoints i, j that are connected by a fixed bone (e.g., elbow and wrist), the bone length is defined as:

$$b(sk) = \|sk_i - sk_j\|_2 \tag{64}$$

where sk is the skeleton defined in Eq. (3).

The *Bone Length Error* (BLE) averages the absolute differences between each predicted bone length and its corresponding ground truth. Given the set of bones B, BLE is defined as:

$$BLE(\hat{sk},\bar{sk}) = \frac{1}{|B|}\sum_{b}^{B} |b(\hat{sk}) - b(\bar{sk})| \tag{65}$$

where $\hat{sk}$ is the skeleton refined and $\bar{sk}$ is the ground-truth skeleton.

9.2. Datasets

The *Carnegie Mellon University Motion Capture Database* (CMU) [208] is the oldest and most used dataset in the context of human motion refinement. It consists of 2605 trials, captured using a marker-based mocap. HDM05 [209] contains more than 2337 sequences performed by 5 actors, captured with a marker-based mocap. Human3.6M [21] is one of the most widely adopted datasets for 3D HPE. It comprises 3.6 million frames obtained through 4 RGB cameras at 50 Hz. It includes 3D skeletons acquired with a marker-based mocap. It is one of the datasets most frequently used in the literature to test HPE software, both single-camera and multi-camera. The *Berkeley Multimodal Human Action Database* (MHAD) contains about 82 min of videos, divided into 11 actions performed by 12 actors. The recording setup consists of a marker-based mocap, 4 stereo-vision cameras, and 2 Kinects.

10. Quantitative comparison

We conducted a quantitative comparison to evaluate the effectiveness of different filtering methods. Table 7 provides an overview of these results using the Human3.6M dataset, injected with four different types of noise (see Table 6).

In *low error jitter*, the function $\xi_{\mathrm{LEJ}}(\cdot)$, which corrupts the input data, is modeled as:

$$\xi_{\mathrm{LEJ}}(x) = x + e_\alpha(2e-1), e \sim \mathcal{N}(0,1) \tag{66}$$

where x is the clean input data and e_α is a constant ($e_\alpha = 100$ mm). In *high error jitter*, the function $\xi_{\mathrm{HEJ}}(\cdot)$ is modeled as:

$$\xi_{\mathrm{HEJ}}(x) = \xi_{\mathrm{LEJ}}(x) + B_{0.1}e_\beta(2e-1), e \sim \mathcal{N}(0,1) \tag{67}$$

where e_β is a constant set to 500 mm. $B_N \sim \mathrm{Bernoulli}(N)$ is an indicator random variable that determines whether high noise is added to the system, with $B = 1$ indicating the presence of noise and $B = 0$ indicating no noise. In the experiments, each data point has a 10% probability of being corrupted with this type of error. In *missing keypoints*, the function $\xi_{\mathrm{MK}}(\cdot)$ is modeled as:

$$\xi_{\mathrm{MK}}(x) = \begin{cases} x & \text{if } B_{0.9} \\ nan & \text{otherwise} \end{cases} \tag{68}$$

where each data point has a 10% probability of being masked. In *missing keypoints with error*, the function $\xi_{\mathrm{MKE}}(\cdot)$ is modeled as:

$$\xi_{\mathrm{MKE}}(x) = \begin{cases} \xi_{\mathrm{LEJ}}(x) & \text{if } B_{0.9} \\ nan & \text{otherwise} \end{cases} \tag{69}$$

In addition to accuracy, latency is reported for each method, representing the processing time of a single frame, offering insight into their applicability for real-time use cases.

We used subjects 9 and 11 for the test, and the other subjects for training the learned methods. We set the temporal window to 20 frames for all methods and briefly fine-tuned each parameter to minimize the error. Further implementation details can be found in the GitHub repository.[1] All experiments are run on a desktop setup (AMD Ryzen 9 7950X, 64 GB RAM DDR5, Nvidia RTX 4090)

The results confirm that all tested methods successfully denoise corrupted input data. General-purpose filters provide effective noise suppression, while state observers manage both jitter and missing keypoints simultaneously. Their strong performance across varied distortion conditions highlights their value in practical motion refinement pipelines. In contrast, dimensionality reduction methods suffer when using a short temporal window, making them more appropriate for offline processing. Deep neural network approaches are highly dependent on the training data, network architecture, and noise type; thus,

[1] https://github.com/PARCO-LAB/mocap-refinement.

Table 6
Quantitative comparison of different methods on the Human3.6M dataset corrupted with synthetic Gaussian noise and occlusions.

Category	Method	Low error jitter		High error jitter		Missing keypoints		Missing keypoints with error		Latency
		MPJPE (mm)	Accel (mm/s^2)	MPJPE (mm)	Accel (mm/s^2)	MPJPE (mm)	Accel (mm/s^2)	MPJPE (mm)	Accel (mm/s^2)	time (ms)
Baseline		96.1	229.96	146.31	381.87	–	–	–	–	
GeneralPurpose	SMA	74.6	11.5	85.0	19.7	–	–	–	–	<1
	WMA	76.8	4.4	88.7	7.0	–	–	–	–	<1
	EMA	55.4	85.8	89.6	140.8	–	–	–	–	<1
	Butterworth	24.8	3.0	41.5	5.0	–	–	–	–	~2
	Savitzky–Golay	63.3	116.7	102.1	194.5	–	–	–	–	<1
	Interpolation	–	–	–	–	0.3	1.1	244.8	634.0	~3
	LSG	62.7	17.2	78.4	19.4	–	–	–	–	<1
StateObserver	Kalman filter (0th)	46.4	29.4	66.0	48.3	30.1	2.8	48.7	29.6	<1
	Kalman filter (1st)	42.3	33.0	65.9	54.2	21.4	2.4	44.2	33.2	<1
	Kalman filter (2nd)	43.1	35.7	68.8	58.7	19.0	2.4	45.0	36.0	<1
DimensionalityReduction	Truncated SVD	64.0	143.7	98.3	226.6	–	–	–	–	<1
	LRMC	–	–	–	–	4.9	16.7	92.5	219.7	<1
	Noisy LRMC	–	–	–	–	32.3	109.4	111.1	267.8	<1
	RPCA	79.8	182.0	90.2	207.7	–	–	–	–	<1
Deep NeuralNetwork	Undercomplete AE	83.6	197.7	128.0	331.0	–	–	–	–	~3
	Denoising AE	59.5	89.5	92.0	75.6	–	–	–	–	~3
	RNN AE	61.4	84.7	92.0	75.6	–	–	–	–	~5

Table 7
Comparison between the different refinement classes.

Category	Suggested for	Main **advantages**	Main disadvantages	Computational Complexity
General purpose	Low-error jitter; missing keypoints; real-time applications	Simple to implement; computationally efficient; does not need training	Requires information of the past and/or future trajectory; works only at keypoints coordinate level; needs manual tuning	Linear in the number of frames; works in real-time
State observer	Low and high error jitter; missing keypoints; real-time applications	Computationally efficient; does not need training	Needs a nominal motion and noise model; needs manual tuning	Quadratic to cubic in state dimensions; works in real-time
Dimensionality reduction	High-error jitter; missing keypoints; offline applications	Refines multiple keypoints simultaneously; does not need manual tuning	Learns only linear motion correlations; may need training data	Quadratic to cubic in data size (frames $\times$ joints); works best offline
Deep neural network	High error jitter	Refines multiple keypoints simultaneously; learns complex motion correlations and does not need manual tuning	Needs training data; computationally heavy; can introduce bias error depending on the training dataset	Varies from moderate to very high; needs GPU

a thorough exploration of these factors could enhance their refinement capabilities. For instance, Zeng et al. [107] reported an 85% improvement on the *Accel* metric on Human3.6M by designing a model architecture specifically optimized for smooth motion data from a targeted set of HPEs.

11. Discussion and future directions

This survey provided a thorough overview of filtering methods for denoising and completing 3D human motion data. The application requirements must be carefully considered before choosing the best filtering technique. Several factors determine the most appropriate filtering technique for refining mocap data, including the specific type of noise and the computational cost.

Table 7 compares the different classes among the proposed taxonomy. General purpose filters are the easiest and computationally efficient methods to reduce low-error jittering and missing keypoints. However, they do not consider the underlying structure of human motion, which may result in unnatural movements. State observers take a more sophisticated approach based on prior knowledge of human biomechanics. These methods can effectively handle denoising and completion tasks but require accurate motion and noise models. Dimensionality reduction techniques take advantage of the low dimensionality of human motion caused by coordination patterns and biomechanical constraints. While these methods are beneficial for dealing with missing data, their effectiveness can be influenced by the complexity of the motion. Deep learning techniques have gained popularity due to their ability to extract complex motion patterns from data. While they have produced impressive results, their behavior may be unpredictable when applied to data that may differ from the training dataset. Hybrid approaches that combine elements from various categories represent a promising direction, leveraging the strengths of each technique to overcome the limitations of individual methods.

The rapidly evolving field of human motion refinement continues to present open challenges. Existing methods are primarily tested on single noise types, whereas errors in real-world mocap data can be due to different causes. Despite the impressive performance of deep learning methods in motion refinement tasks, they face several practical challenges that can limit their deployment in real-world applications. Training deep neural networks typically requires large-scale, high-quality annotated datasets, which are time-consuming and expensive, especially when ensuring consistency across diverse motion types and occlusion scenarios. Another critical limitation is the computational complexity of many state-of-the-art models, such as transformers and DDPMs, which makes real-time inference difficult on resource-constrained systems like mobile robots or wearable devices. Consequently, there is a growing need for model light-weighting techniques, such as pruning, quantization, and knowledge distillation, that reduce inference latency and memory footprint while preserving model performance.

Real-world applications may span diverse domains. In medical rehabilitation, for instance, gait analysis relies heavily on accurate joint trajectory data to assess patient progress. In such applications, where real-time performance is not as strict as in other domains, dimensionality reduction methods can be advantageous due to their explainability. In contrast, industrial environments, particularly in human–robot collaboration tasks, require quick and robust refinement to prevent collisions or task errors. For such settings, real-time performance requirements critically influence algorithm selection: methods with low latency (e.g., moving averages, Kalman filters, and lightweight autoencoders) are preferred.

As markerless systems become more common, future research should look to combine multi-person tracking and filtering techniques. Also, investigating filtering methods for non-positional data extrapolated from alternative mocap sources, such as those from inertial measurement units (IMUs), would broaden the applicability of these refinement techniques to a wider range of motion analysis applications.

12. Conclusion

Based on rapid technological advancement, human motion analysis has gained significant attention in research on human–machine interaction. Various techniques have been developed to capture human motion, including marker-based and marker-less motion capture systems. While these systems provide reliable human pose data, they are still subject to intrinsic inaccuracies that often lead to noisy keypoint coordinates. Filtering techniques have been proposed to denoise and complete the 3D human motion data generated by these systems. In practical applications, the filter choice critically impacts the quality of refined motion and the feasibility of deployments in real-time systems and resource-constrained settings, such as in telerehabilitation or human–robot interaction. However, there is currently no comprehensive review in the literature that provides a classification and explanation of these filtering techniques. To fill this gap, this survey provided an overview of denoising and completion filters for 3D human motion data. It started with an overview of the different measurement errors and proposed a taxonomy of the different filtering techniques. For each class, it summarized the basic concepts and reported application feedback collected from the literature. The article also includes an open-source library containing the implementation code of each reviewed filtering algorithm, along with the precision of the filtering results on different motion capture datasets. Future research should focus on developing unified frameworks that balance accuracy, speed, and generalizability, possibly through biomechanics-aware hybrid models or lightweight self-supervised learning strategies.

Glossary

Completion Filling in missing samples (gaps) in motion data; also called imputation or in-painting.

Denoising Any operation that suppresses stochastic noise while keeping the underlying kinematic signal intact.

Filter Generic label for any algorithm that processes a signal to suppress noise or guess missing values.

Gaussian Process (GP) Bayesian non-parametric model that provides both mean prediction and uncertainty for each pose.

Human Motion Analysis Capturing, modeling, synthesizing, and interpreting human movements.

Human Pose Estimation (HPE) Computer-vision pipeline that localizes body key-points in 2-D or 3-D from RGB or depth frames.

Infinite Impulse Response (IIR) Filter Digital filter whose impulse response theoretically lasts forever; includes Butterworth and EMA

Inverse Kinematics (IK) Optimization that finds joint angles producing a desired end-effector pose while respecting kinematic constraints.

Kinematic Constraints Anatomical limits (joint ranges, limb lengths) enforced to keep completed poses physically plausible.

Latent Space Representation of data in a lower dimension, revealing underlying hidden features and patterns that are not directly observable in the original input.

Missing Data Absent samples caused by marker occlusion, dropped frames, or sensor malfunction.

Noise Unwanted random deviations in captured joint positions caused by sensor jitter or tracker mis-detections.

State Observer Model-based estimator (e.g. Kalman, Luenberger) that reconstructs hidden joint velocities or torques.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

We have shared the link to the code in the abstract of the manuscript.

References

[1] M.-C. Hu, C.-W. Chen, W.-H. Cheng, C.-H. Chang, J.-H. Lai, J.-L. Wu, Real-time human movement retrieval and assessment with kinect sensor, IEEE Trans. Cybern. 45 (4) (2015) 742–753, http://dx.doi.org/10.1109/TCYB.2014.2335540.

[2] F. Azhar, C.-T. Li, Hierarchical relaxed partitioning system for activity recognition, IEEE Trans. Cybern. 47 (3) (2017) 784–795, http://dx.doi.org/10.1109/TCYB.2016.2526970.

[3] G. Zhu, L. Zhang, P. Shen, J. Song, S.A.A. Shah, M. Bennamoun, Continuous gesture segmentation and recognition using 3dcnn and convolutional lstm, IEEE Trans. Multimed. 21 (4) (2019) 1011–1021, http://dx.doi.org/10.1109/TMM.2018.2869278.

[4] V. Nexus, Vicon nexus product guide, 2015, URL https://documentation.vicon.com/nexus/v2.2/Nexus1_8Guide.pdf.

[5] J.G. Richards, The measurement of human motion: A comparison of commercially available systems, Hum. Mov. Sci. 18 (5) (1999) 589–602, http://dx.doi.org/10.1016/s0167-9457(99)00023-8.

[6] N. McLaughlin, J. Martinez-del Rincon, P. Miller, 3-D human pose estimation using iterative conditional squeeze and excitation networks, IEEE Trans. Cybern. 52 (1) (2020) 687–699.

[7] C. Zheng, W. Wu, C. Chen, T. Yang, S. Zhu, J. Shen, N. Kehtarnavaz, M. Shah, Deep learning-based human pose estimation: A survey, 2020, http://dx.doi.org/10.48550/ARXIV.2012.13392, URL https://arxiv.org/abs/2012.13392.

[8] M.M. Arzani, M. Fathy, A.A. Azirani, E. Adeli, Switching structured prediction for simple and complex human activity recognition, IEEE Trans. Cybern. 51 (12) (2020) 5859–5870.

[9] C.-Y. Zhang, Y.-Y. Xiao, J.-C. Lin, C.P. Chen, W. Liu, Y.-H. Tong, 3-D deconvolutional networks for the unsupervised representation learning of human motions, IEEE Trans. Cybern. 52 (1) (2020) 398–410.

[10] B. Sun, S. Wang, D. Kong, L. Wang, B. Yin, Real-time human action recognition using locally aggregated kinematic-guided skeletonlet and supervised hashing-by-analysis model, IEEE Trans. Cybern. 52 (6) (2021) 4837–4849.

[11] H. Ma, Z. Yang, H. Liu, Fine-grained unsupervised temporal action segmentation and distributed representation for skeleton-based human motion analysis, IEEE Trans. Cybern. 52 (12) (2021) 13411–13424.

[12] A. Badiola-Bengoa, A. Mendez-Zorrilla, A systematic review of the application of camera-based human pose estimation in the field of sport and physical exercise, Sensors 21 (18) (2021) 5996, http://dx.doi.org/10.3390/s21185996.

[13] C. Zimmermann, T. Welschehold, C. Dornhege, W. Burgard, T. Brox, 3D human pose estimation in RGBD images for robotic task learning, 2018, http://dx.doi.org/10.1109/icra.2018.8462833.

[14] L. Geretti, S. Centomo, M. Boldo, E. Martini, N. Bombieri, D. Quaglia, T. Villa, Process-driven collision prediction in human-robot work environments, in: IEEE 27th International Conference on Emerging Technologies and Factory Automaton, ETFA, 2022, pp. 1–8, http://dx.doi.org/10.1109/etfa52439.2022.9921732.

[15] E. Martini, M. Boldo, S. Aldegheri, N. Valè, M. Filippetti, N. Smania, M. Bertucco, A. Picelli, N. Bombieri, Enabling gait analysis in the telemedicine practice through portable and accurate 3D human pose estimation, Comput. Methods Programs Biomed. 225 (2022) 107016, http://dx.doi.org/10.1016/j.cmpb.2022.107016.

[16] B. Bonnechere, B. Jansen, P. Salvia, H. Bouzahouene, V. Sholukha, J. Cornelis, M. Rooze, S. Van Sint Jan, Determination of the precision and accuracy of morphological measurements using the kinect™ sensor: Comparison with standard stereophotogrammetry, Ergonomics 57 (4) (2014) 622–631.

[17] C. Nakatsuka, S. Komorita, Denoising 3D human poses from low-resolution video using variational autoencoder, IEEE Int. Conf. Intell. Robot. Syst. (2021) 4625–4630, http://dx.doi.org/10.1109/IROS51168.2021.9636144.

[18] G.E. Gorton III, D.A. Hebert, M.E. Gannotti, Assessment of the kinematic variability among 12 motion analysis laboratories, Gait & Posture 29 (3) (2009) 398–402.

[19] A.M. Gonabadi, G.M. Cesar, T.W. Buster, J.M. Burnfield, Effect of gap-filling technique and gap location on linear and nonlinear calculations of motion during locomotor activities, Gait & Posture 94 (2022) 85–92.

[20] M.W. Whittle, Chapter 4 - methods of gait analysis, in: M.W. Whittle (Ed.), Gait Analysis (Fourth Edition), fourth ed., Butterworth-Heinemann, Edinburgh, 2007, pp. 137–175, http://dx.doi.org/10.1016/B978-075068883-3.50009-X.

[21] C. Ionescu, D. Papava, V. Olaru, C. Sminchisescu, Human3.6M: Large scale datasets and predictive methods for 3D human sensing in natural environments, IEEE Trans. Pattern Anal. Mach. Intell. 36 (7) (2014) 1325–1339, http://dx.doi.org/10.1109/TPAMI.2013.248.

[22] Y. He, R. Yan, K. Fragkiadaki, S.-I. Yu, Epipolar transformers, in: Proceedings of the Ieee/Cvf Conference on Computer Vision and Pattern Recognition, 2020, pp. 7779–7788.

[23] S. Chun, S. Park, J.Y. Chang, Learnable human mesh triangulation for 3D human pose and shape estimation, in: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision, 2023, pp. 2850–2859.

[24] Z. Zhang, C. Wang, W. Qiu, W. Qin, W. Zeng, Adafuse: Adaptive multiview fusion for accurate human pose estimation in the wild, Int. J. Comput. Vis. 129 (2021) 703–718.

[25] N.D. Reddy, L. Guigues, L. Pishchulin, J. Eledath, S.G. Narasimhan, Tessetrack: End-to-end learnable multi-person articulated 3d pose tracking, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2021, pp. 15190–15200.

[26] S. Wu, S. Jin, W. Liu, L. Bai, C. Qian, D. Liu, W. Ouyang, Graph-based 3d multi-person pose estimation using multi-view images, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 11148–11157.

[27] J. Zhang, Y. Cai, S. Yan, J. Feng, et al., Direct multi-view multi-person 3d pose estimation, Adv. Neural Inf. Process. Syst. 34 (2021) 13153–13164.

[28] H. Ye, W. Zhu, C. Wang, R. Wu, Y. Wang, Faster VoxelPose: Real-time 3D human pose estimation by orthographic projection, in: Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part VI, Springer, 2022, pp. 142–159.

[29] H. Lou, J. Chai, Example-based human motion denoising, IEEE Trans. Vis. Comput. Graphics 16 (5) (2010) 870–879, http://dx.doi.org/10.1109/TVCG.2010.23.

[30] X. Wei, B. Xiao, Q. Zhang, R. Liu, A rigid structure matching-based noise data processing approach for human motion capture, Proc. - Work. Digit. Media Digit. Content Manag. DMDCM 2011 (2011) 91–96, http://dx.doi.org/10.1109/DMDCM.2011.32.

[31] M. Edwards, R. Green, Low-latency filtering of kinect skeleton data for video game control, ACM Int. Conf. Proceeding Ser. 19-21-Nove (2014) 190–195, http://dx.doi.org/10.1145/2683405.2683453.

[32] M.M. Ardestani, H. Yan, Noise reduction in human motion-captured signals for computer animation based on B-spline filtering, Sensors 22 (12) (2022) http://dx.doi.org/10.3390/s22124629.

[33] D.A. Winter, H. Sidwall, D.A. Hobson, Measurement and reduction of noise in kinematics of locomotion, J. Biomech. 7 (2) (1974) 157–159, http://dx.doi.org/10.1016/0021-9290(74)90056-6.

[34] A. Cappello, P.F.L. Palombara, A. Leardini, Optimization and smoothing techniques in movement analysis, Int. J. Bio-Med. Comput. 41 (3) (1996) 137–151, http://dx.doi.org/10.1016/0020-7101(96)01167-1.

[35] G. Giakas, V. Baltzopoulos, A comparison of automatic filtering techniques applied to biomechanical walking data, J. Biomech. 30 (8) (1997) 847–850, http://dx.doi.org/10.1016/s0021-9290(97)00042-0.

[36] D.A. Winter, Biomechanics and Motor Control of Human Movement, John Wiley & Sons, 2009.

[37] M.W. Whittle, Gait Analysis: an Introduction, Butterworth-Heinemann, 2014.

[38] F. Crenna, G.B. Rossi, M. Berardengo, Filtering biomechanical signals in movement analysis, Sensors 21 (13) (2021) 4580, http://dx.doi.org/10.3390/s21134580.

[39] S. Li, T. Caelli, M. Ferraro, P.N. Pathirana, A novel bio-kinematic encoder for human exercise representation and decomposition - part 2: Robustness and optimisation, in: 2013 Int. Conf. Control. Autom. Inf. Sci., IEEE, 2013, pp. 30–35, http://dx.doi.org/10.1109/ICCAIS.2013.6720525, http://ieeexplore.ieee.org/document/6720524/.

[40] H.J. Woltring, On optimal smoothing and derivative estimation from noisy displacement data in biomechanics, Hum. Mov. Sci. 4 (3) (1985) 229–245, http://dx.doi.org/10.1016/0167-9457(85)90004-1.

[41] S. Jakób, S.-P. Maria, Comparison of interpolation methods based on real human motion data, 2014, http://dx.doi.org/10.12915/pe.2014.10.54.

[42] M. Tits, J. Tilmanne, T. Dutoit, Robust and automatic motion-capture data recovery using soft skeleton constraints and model averaging, PLoS One 13 (7) (2018) 1–21, http://dx.doi.org/10.1371/journal.pone.0199744.

[43] A. Gupta, V.B. Semwal, Occluded gait reconstruction in multi person gait environment using different numerical methods, Multimedia Tools Appl. 81 (16) (2022) 23421–23448, http://dx.doi.org/10.1007/s11042-022-12218-2.

[44] A. Aristidou, J. Cameron, J. Lasenby, Predicting missing markers to drive real-time centre of rotation estimation, in: Articulated Motion and Deformable Objects, Springer Berlin Heidelberg, 2008, pp. 238–247.

[45] Q. Wu, P. Boulanger, Real-time estimation of missing markers for reconstruction of human motion, in: 2011 XIII Symposium on Virtual Reality, IEEE, 2011, pp. 161–168, http://dx.doi.org/10.1109/svr.2011.35.

[46] S.R. Tripathy, K. Chakravarty, A. Sinha, D. Chatterjee, S.K. Saha, Constrained Kalman filter for improving kinect based measurements, in: 2017 IEEE Int. Symp. Circuits Syst., vol. 2018-Septe, IEEE, 2017, pp. 1–4, http://dx.doi.org/10.1109/ISCAS.2017.8050664, URL http://ieeexplore.ieee.org/document/8050664/.

[47] F. Ahmed, A.S. Hossain Bari, B. Sieu, J. Sadeghi, J. Scholten, M.L. Gavrilova, Kalman filter-based noise reduction framework for posture estimation using depth sensor, in: Proc. 2019 IEEE 18th Int. Conf. Cogn. Informatics Cogn. Comput. ICCI*CC 2019, 2019, pp. 150–158, http://dx.doi.org/10.1109/ICCICC46617.2019.9146069.

[48] Y. Hu, Y. Liu, Y. Xu, Y. Wang, Human steering angle estimation in video based on key point detection and Kalman filter, Control. Theory Technol. 20 (3) (2022) 408–417, http://dx.doi.org/10.1007/s11768-022-00100-3.

[49] J. Shu, F. Hamano, J. Angus, Application of extended Kalman filter for improving the accuracy and smoothness of kinect skeleton-joint estimates, J. Engrg. Math. 88 (1) (2014) 161–175.

[50] J. Niu, X. Wang, D. Wang, L. Ran, A novel method of human joint prediction in an occlusion scene by using low-cost motion capture technique, Sensors (Switzerland) 20 (4) (2020) http://dx.doi.org/10.3390/s20041119.

[51] A.B.L. Larsen, S. Hauberg, K.S. Pedersen, Unscented kalman filtering for articulated human tracking, Lect. Notes Comput. Sci. (Incl. Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics) 6688 LNCS (2011) 228–237, http://dx.doi.org/10.1007/978-3-642-21227-7-22.

[52] P. Agarwal, S. Kumar, J. Ryde, J.J. Corso, V.N. Krovi, Estimating human dynamics on-the-fly using monocular video for pose estimation, Robot. Sci. Syst. 8 (July) (2013) 1–8, http://dx.doi.org/10.15607/rss.2012.viii.001,

[53] P. Vartiainen, T. Bragge, J.P. Arokoski, P.A. Karjalainen, Nonlinear state-space modeling of human motion using 2-D marker observations, IEEE Trans. Biomed. Eng. 61 (7) (2014) 2167–2178, http://dx.doi.org/10.1109/tbme.2014.2318354.

[54] R.P. Matthew, S. Seko, R. Bajcsy, J. Lotz, Kinematic and kinetic validation of an improved depth camera motion assessment system using rigid bodies, IEEE J. Biomed. Heal. Inform. 23 (4) (2019) 1784–1793, http://dx.doi.org/10.1109/JBHI.2018.2872834.

[55] K. Loumponias, N. Vretos, P. Daras, G. Tsaklidis, Using Kalman filter and tobit Kalman filter in order to improve the motion recorded by kinect sensor II, Proc. 29th Panhellenic Stat. Conf. 2016 (2016) 322–334, URL https://zenodo.org/record/1073287.

[56] K. Loumponias, N. Vretos, G. Tsaklidis, P. Daras, An improved tobit Kalman filter with adaptive censoring limits, Circuits, Syst. Signal Process. 39 (11) (2020) 5588–5617, http://dx.doi.org/10.1007/s00034-020-01422-w, arXiv:1911.06190,

[57] Y.R. Musunuri, O.S. Kwon, State estimation using a randomized unscented kalman filter for 3d skeleton posture, Electron. 10 (8) (2021) http://dx.doi.org/10.3390/electronics10080971.

[58] D. Gomes, V. Guimarães, J. Silva, A fully-automatic gap filling approach for motion capture trajectories, Appl. Sci. 11 (21) (2021) 9847, http://dx.doi.org/10.3390/app11219847, URL https://www.mdpi.com/2076-3417/11/21/9847.

[59] E. Martini, H. Parekh, S. Peng, N. Bombieri, N. Figueroa, A robust filter for marker-less multi-person tracking in human-robot interaction scenarios, in: 2024 33rd IEEE International Conference on Robot and Human Interactive Communication, ROMAN, IEEE, 2024, pp. 424–429.

[60] R. Lai, P. Yuen, K. Lee, Motion capture data completion and denoising by singular value thresholding, Proc. Eurograph. Assoc. (2011) 1–4, URL http://www.comp.hkbu.edu.hk/~yqlai/images/egfinal.pdf.

[61] G. Liu, L. McMillan, Estimation of missing markers in human motion capture, Vis. Comput. 22 (9–11) (2006) 721–728, http://dx.doi.org/10.1007/s00371-006-0080-9.

[62] T. Tangkuampien, D. Suter, Human motion de-noising via greedy kernel principal component analysis filtering, Proc. - Int. Conf. Pattern Recognit. 3 (2006) 457–460, http://dx.doi.org/10.1109/ICPR.2006.639.

[63] H.P. Shum, E.S. Ho, Y. Jiang, S. Takagi, Real-time posture reconstruction for microsoft kinect, IEEE Trans. Cybern. 43 (5) (2013) 1357–1369, http://dx.doi.org/10.1109/TCYB.2013.2275945.

[64] P.A. Federolf, A novel approach to solve the 'missing marker problem' in marker-based motion analysis that exploits the segment coordination patterns in multi-limb motion data, in: R. Balasubramaniam (Ed.), PLoS One 8 (10) (2013) e78689, http://dx.doi.org/10.1371/journal.pone.0078689.

[65] Ø. Gløersen, P. Federolf, Predicting missing marker trajectories in human motion data using marker intercorrelations, in: M. Cebecauer (Ed.), PLoS One 11 (3) (2016) e0152616, http://dx.doi.org/10.1371/journal.pone.0152616, URL https://dx.plos.org/10.1371/journal.pone.0152616.

[66] Z. Li, H. Yu, H.D. Kieu, T.L. Vuong, J.J. Zhang, PCA-based robust motion data recovery, IEEE Access 8 (1) (2020) 76980–76990, http://dx.doi.org/10.1109/ACCESS.2020.2989744.

[67] C.-H. Tan, J. Hou, L.-P. Chau, Human motion capture data recovery using trajectory-based matrix completion, Electron. Lett. 49 (12) (2013) 752–754.

[68] C.H. Tan, J.H. Hou, L.P. Chau, Motion capture data recovery using skeleton constrained singular value thresholding, Vis. Comput. 31 (11) (2015) 1521–1532, http://dx.doi.org/10.1007/s00371-014-1031-5.

[69] D. Bautembach, I. Oikonomidis, A. Argyros, A comparative study of matrix completion and recovery techniques for human pose estimation, ACM Int. Conf. Proceeding Ser. (2018) 23–30, http://dx.doi.org/10.1145/3197768.3197791.

[70] Q. Cui, B. Chen, H. Sun, Nonlocal low-rank regularization for human motion recovery based on similarity analysis, Inf. Sci. (Ny) 493 (2019) 57–74, http://dx.doi.org/10.1016/j.ins.2019.04.031.

[71] S. Mohaoui, A. Dmytryshyn, CP decomposition-based algorithms for completion problem of motion capture data, Pattern Anal. Appl. 27 (4) (2024) 133.

[72] L. Li, J. McCann, N. Pollard, C. Faloutsos, BoLeRO: A principled technique for including bone length constraints in motion capture occlusion filling, Comput. Animat. 2010 - ACM SIGGRAPH / Eurographics Symp. Proc. SCA 2010 (2010) 179–188.

[73] Y. Feng, J. Xiao, Y. Zhuang, X. Yang, J.J. Zhang, R. Song, Exploiting temporal stability and low-rank structure for motion capture data refinement, Inf. Sci. (Ny) 277 (2014) 777–793, http://dx.doi.org/10.1016/j.ins.2014.03.013.

[74] W. Hu, Z. Wang, X. Yang, J.J. Zhang, An efficient bilinear factorization based method for motion capture data refinement, in: Current Trends in Computer Science and Mechanical Automation, vol. 2, De Gruyter Open, 2017, pp. 533–547, http://dx.doi.org/10.1515/9783110584998-055.

[75] B. Chen, H. Sun, G. Xia, L. Feng, B. Li, Human motion recovery utilizing truncated schatten p-norm and kinematic constraints, Inf. Sci. (Ny). 450 (2018) 89–108, http://dx.doi.org/10.1016/j.ins.2018.02.052.

[76] R. Imamura, M. Okuda, MOCAP signal interpolation using low-rank matrix recovery, in: 2018 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference, APSIPA ASC, IEEE, 2018, pp. 871–874, http://dx.doi.org/10.23919/apsipa.2018.8659679.

[77] G. Xia, H. Sun, B. Chen, Q. Liu, L. Feng, G. Zhang, R. Hang, Nonlinear low-rank matrix completion for human motion recovery, IEEE Trans. Image Process. 27 (6) (2018) 3011–3024, http://dx.doi.org/10.1109/TIP.2018.2812100.

[78] W. Hu, X. Zhu, T. Wang, Y. Yi, G. Yu, Discrete subspace structure constrained human motion capture data recovery, Appl. Soft Comput. 129 (2022) 109617, http://dx.doi.org/10.1016/j.asoc.2022.109617.

[79] X. Liu, Y.M. Cheung, S.J. Peng, Z. Cui, B. Zhong, J.X. Du, Automatic motion capture data denoising via filtered subspace clustering and low rank matrix approximation, Signal Process. 105 (2014) 350–362, http://dx.doi.org/10.1016/j.sigpro.2014.06.009.

[80] M. Wang, K. Li, F. Wu, Y.-k. Lai, J. Yang, 3-D motion recovery via low rank matrix analysis, 2016.

[81] K. Li, M. Wang, Y.-K. Lai, J. Yang, F. Wu, 3-D motion recovery via low rank matrix restoration on articulation graphs, in: 2017 IEEE International Conference on Multimedia and Expo, ICME, IEEE, 2017, http://dx.doi.org/10.1109/icme.2017.8019486.

[82] J. Yang, J. Shi, Y. Zhu, K. Li, C. Hou, 3D motion recovery via low rank matrix restoration with Hankel-like augmentation, Proc. - IEEE Int. Conf. Multimed. Expo 2020-July (2020) http://dx.doi.org/10.1109/ICME46284.2020.9102858.

[83] J. Yang, X. Guo, K. Li, M. Wang, Y.K. Lai, F. Wu, Spatio-temporal reconstruction for 3D motion recovery, IEEE Trans. Circuits Syst. Video Technol. 30 (6) (2020) 1583–1596, http://dx.doi.org/10.1109/TCSVT.2019.2907324.

[84] M.S. Subodh Raj, S.N. George, l 1/2 regularized RPCA technique for 3D human action recovery, in: 2020 IEEE 17th India Counc. Int. Conf., IEEE, 2020, pp. 1–5, http://dx.doi.org/10.1109/INDICON49873.2020.9342124, URL https://ieeexplore.ieee.org/document/9342124/.

[85] W. Hu, Y. Lu, J. Ren, A fixed-point proximity algorithm for recovering low-rank components from incomplete observation data with application to motion capture data refinement, J. Comput. Appl. Math. 410 (2022) 114224, http://dx.doi.org/10.1016/j.cam.2022.114224.

[86] M.S. Subodh Raj, S.N. George, A fast non-convex optimization technique for human action recovery from misrepresented 3D motion capture data using trajectory movement and pair-wise hierarchical constraints, J. Ambient. Intell. Humaniz. Comput. (0123456789) (2022) http://dx.doi.org/10.1007/s12652-022-04349-z.

[87] M.S. Raj, S.N. George, A fast and efficient approach for human action recovery from corrupted 3-D motion capture data using QR decomposition-based approximate SVD, IEEE Trans. Human- Mach. Syst. 54 (4) (2024) 395–405, http://dx.doi.org/10.1109/THMS.2024.3400290.

[88] S.-j. Peng, G.-f. He, X. Liu, H.-z. Wang, Hierarchical block-based incomplete human mocap data recovery using adaptive nonnegative matrix factorization, Comput. Graph. 49 (2015) 10–23, http://dx.doi.org/10.1016/j.cag.2015.04.004, URL https://linkinghub.elsevier.com/retrieve/pii/S0097849315000436.

[89] J. Xiao, Y. Feng, M. Ji, X. Yang, J.J. Zhang, Y. Zhuang, Sparse motion bases selection for human motion denoising, Signal Process. 110 (2015) 108–122, http://dx.doi.org/10.1016/j.sigpro.2014.08.017.

[90] Y. Feng, M. Ji, J. Xiao, X. Yang, J.J. Zhang, Y. Zhuang, X. Li, Mining spatial-temporal patterns and structural sparsity for human motion data denoising, IEEE Trans. Cybern. 45 (12) (2015) 2693–2706, http://dx.doi.org/10.1109/TCYB.2014.2381659.

[91] G. Xia, H. Sun, G. Zhang, L. Feng, Human motion recovery jointly utilizing statistical and kinematic information, Inf. Sci. (Ny). 339 (2016) 189–205, http://dx.doi.org/10.1016/j.ins.2015.12.041.

[92] M. Wang, K. Li, F. Wu, Y.-K. Lai, J. Yang, 3-D motion recovery via low rank matrix analysis, in: 2016 Visual Communications and Image Processing, VCIP, 2016, pp. 1–4, http://dx.doi.org/10.1109/VCIP.2016.7805473.

[93] J. Mei, X. Chen, C. Wang, A. Yuille, X. Lan, W. Zeng, Learning to refine 3D human pose sequences, Proc. - 2019 Int. Conf. 3D Vis. 3DV 2019 (2019) 358–366, http://dx.doi.org/10.1109/3DV.2019.00047,

[94] C. Wang, H. Qiu, A.L. Yuille, W. Zeng, Learning basis representation to refine 3D human pose estimations, Proc. AAAI Conf. Artif. Intell. 33 (01) (2019) 8925–8932, http://dx.doi.org/10.1609/aaai.v33i01.33018925, URL https://ojs.aaai.org/index.php/AAAI/article/view/4920.

[95] J. Bütepage, M.J. Black, D. Kragic, H. Kjellström, Deep representation learning for human motion prediction and classification, Proc. - 30th IEEE Conf. Comput. Vis. Pattern Recognit. CVPR 2017 2017-January (2017) 1591–1599, http://dx.doi.org/10.1109/CVPR.2017.173, arXiv:1702.07486.

[96] G. Xia, P. Xue, H. Sun, Y. Sun, D. Zhang, Q. Liu, Local self-expression subspace learning network for motion capture data, IEEE Trans. Image Process. 31 (2022) 4869–4883, http://dx.doi.org/10.1109/TIP.2022.3189822.

[97] O. Yuhai, A. Choi, Y. Cho, H. Kim, J.H. Mun, Deep-learning-based recovery of missing optical marker trajectories in 3D motion capture systems, Bioengineering 11 (6) (2024) 560.

[98] T. Kucherenko, J. Beskow, H. Kjellström, A neural network approach to missing marker reconstruction in human motion capture, 2018, arXiv:arXiv:1803.02665v4.

[99] D. Holden, J. Saito, T. Komura, T. Joyce, Learning motion manifolds with convolutional autoencoders, SIGGRAPH Asia 2015 Tech. Briefs, SA 2015 (2015) 1–4, http://dx.doi.org/10.1145/2820903.2820918.

[100] U. Mall, G.R. Lal, S. Chaudhuri, P. Chaudhuri, A deep recurrent framework for cleaning motion capture data, 2017, arXiv:1712.03380. URL http://arxiv.org/abs/1712.03380.

[101] S. Li, Y. Zhou, H. Zhu, W. Xie, Y. Zhao, X. Liu, Bidirectional recurrent autoencoder for 3D skeleton motion data refinement, Comput. Graph. 81 (2019) 92–103, http://dx.doi.org/10.1016/j.cag.2019.03.010.

[102] S.J. Li, H.S. Zhu, L.P. Zheng, L. Li, A perceptual-based noise-agnostic 3D skeleton motion data refinement network, IEEE Access 8 (2020) 52927–52940, http://dx.doi.org/10.1109/ACCESS.2020.2980316.

[103] L. Ji, R. Liu, D. Zhou, Q. Zhang, X. Wei, Missing data recovery for human mocap data based on A-LSTM and LS constraint, 2020 IEEE 5th Int. Conf. Signal Image Process. ICSIP 2020 (2020) 729–734, http://dx.doi.org/10.1109/ICSIP49896.2020.9339359.

[104] Y. Zhu, Reconstruction of missing markers in motion capture based on deep learning, in: 2020 IEEE 3rd Int. Conf. Inf. Syst. Comput. Aided Educ., IEEE, 2020, pp. 346–349, http://dx.doi.org/10.1109/ICISCAE51034.2020.9236900, URL https://ieeexplore.ieee.org/document/9236900/.

[105] M. Kaufmann, E. Aksan, J. Song, F. Pece, R. Ziegler, O. Hilliges, Convolutional autoencoders for human motion infilling, Proc. - 2020 Int. Conf. 3D Vis. 3DV 2020 (2020) 918–927, http://dx.doi.org/10.1109/3DV50981.2020.00102, arXiv:2010.11531.

[106] S. Lohit, R. Anirudh, P. Turaga, Recovering trajectories of unmarked joints in 3d human actions using latent space optimization, Proc. - 2021 IEEE Winter Conf. Appl. Comput. Vis. WACV 2021 (2021) 2341–2350, http://dx.doi.org/10.1109/WACV48630.2021.00239, arXiv:2012.02043.

[107] A. Zeng, L. Yang, X. Ju, J. Li, J. Wang, Q. Xu, SmoothNet: A plug-and-play network for refining human poses in videos, in: European Conference on Computer Vision, Springer, 2022.

[108] J. Li, Graph matching for marker labeling and missing marker reconstruction with bone constraint by LSTM in optical motion capture, IEEE Access 9 (2021) http://dx.doi.org/10.1109/ACCESS.2021.3060385.
[109] Y. Zhu, Refining method of mocap data based on LSTM, in: 2022 IEEE 2nd Int. Conf. Data Sci. Comput. Appl., IEEE, 2022, pp. 740–743, http://dx.doi.org/10.1109/ICDSCA56264.2022.9988554, URL https://ieeexplore.ieee.org/document/9988554/.
[110] Y. Zhu, F. Zhang, Z. Xiao, Attention-based recurrent autoencoder for motion capture denoising, J. Internet Technol. 23 (6) (2022) 1325–1333, http://dx.doi.org/10.53106/160792642022112306015.
[111] Y.-Q.Z. Yong-Qiong Zhu, Y.-M.C. Yong-Qiong Zhu, F.Z. Ye-Ming Cai, Motion capture data denoising based on LSTNet autoencoder, J. Internet Technol. 23 (1) (2022) 011–020, http://dx.doi.org/10.53106/160792642022012301002, URL http://ericdata.com/tw/detail.aspx?no=902988.
[112] Y. Zhu, Y. Cai, Predicting missing markers in mocap data using LSTNet, in: Proceedings of the 7th International Conference on Cyber Security and Information Engineering, 2022, pp. 947–952.
[113] X. Chen, C. Wang, X. Lan, N. Zheng, W. Zeng, Neighborhood geometric structure-preserving variational autoencoder for smooth and bounded data sources, IEEE Trans. Neural Netw. Learn. Syst. 33 (8) (2022) 3598–3611, http://dx.doi.org/10.1109/TNNLS.2021.3053591.
[114] G. Fiche, S. Leglaive, X. Alameda-Pineda, R. Séguier, Motion-DVAE: Unsupervised learning for fast human motion denoising, Proc. - MIG 2023 16th ACM SIGGRAPH Conf. Motion, Interact. Games (2023) http://dx.doi.org/10.1145/3623264.3624454, arXiv:2306.05846.
[115] Q. Cui, H. Sun, Y. Kong, X. Sun, Deep human dynamics prior, in: Proc. 29th ACM Int. Conf. Multimed., ACM, New York, NY, USA, 2021, pp. 4371–4379, http://dx.doi.org/10.1145/3474085.3475581, URL https://dl.acm.org/doi/10.1145/3474085.3475581.
[116] X. Pan, B. Zheng, X. Jiang, G. Xu, X. Gu, J. Li, Q. Kou, H. Wang, T. Shao, K. Zhou, X. Jin, A locality-based neural solver for optical motion capture, Proc. - SIGGRAPH Asia 2023 Conf. Pap. SA 2023 (2023) http://dx.doi.org/10.1145/3610548.3618148, arXiv:2309.00428.
[117] M. Duc Nguyen, T. Linh Hoang, V. Cuong Ta, Reconstructing missing joints in 3D human motion with temporal-structural awareness graph neural network, Proc. - Int. Conf. Knowl. Syst. Eng. KSE (2023) 1–6, http://dx.doi.org/10.1109/KSE59128.2023.10299418.
[118] R. He, Y. Tang, X. Li, J. Lu, Context-aware inpainter-refiner for skeleton-based human motion completion, Proc. - Int. Conf. Image Process. ICIP (2023) 296–300, http://dx.doi.org/10.1109/ICIP49359.2023.10222862.
[119] W. Lee, S. Park, T. Kim, Denoising graph autoencoder for missing human joints reconstruction, IEEE Access 12 (2024) 57381–57389.
[120] C. Choi, J. Lee, H.-J. Chung, J. Park, B. Park, S. Sohn, S. Lee, Directed graph-based refinement of three-dimensional human motion data using spatial-temporal information, Int. J. Precis. Eng. Manuf.- Smart Technol. 2 (2024) 33–46.
[121] L. Xu, Q. Wang, C. Yang, Spatial-temporal graph U-net for skeleton-based human motion infilling, in: 2024 IEEE International Conference on Industrial Technology, ICIT, IEEE, 2024, pp. 1–6.
[122] Q. Cui, H. Sun, Y. Li, Y. Kong, A deep bi-directional attention network for human motion recovery, IJCAI Int. Jt. Conf. Artif. Intell. 2019-Augus (2019) 701–707, http://dx.doi.org/10.24963/ijcai.2019/99.
[123] C. Xu, R.T. Tan, Y. Tan, S. Chen, X. Wang, Y. Wang, Auxiliary tasks benefit 3D skeleton-based human motion prediction, Proc. IEEE Int. Conf. Comput. Vis. (2023) 9475–9486, http://dx.doi.org/10.1109/ICCV51070.2023.00872, arXiv:2308.08942.
[124] J. Liu, J. Liu, P. Li, A method of human motion reconstruction with sparse joints based on attention mechanism, Proc. - 2023 2023 IEEE Int. Conf. Bioinforma. Biomed. BIBM 2023 (2023) 2647–2654, http://dx.doi.org/10.1109/BIBM58861.2023.10385759.
[125] H.G. Chi, S. Chi, S. Chan, K. Ramani, Pose relation transformer refine occlusions for human pose estimation, Proc. - IEEE Int. Conf. Robot. Autom. 2023-May (Icra) (2023) 6138–6145, http://dx.doi.org/10.1109/ICRA48891.2023.10161259.
[126] D. Björkstrand, J. Sullivan, L. Bretzner, G. Loy, T. Wang, Cross-attention masked auto-encoder for human 3D motion infilling and denoising, Br. Mach. Vis. Conf. (2023) 1–13.
[127] E.V. Mascaró, H. Ahn, D. Lee, A unified masked autoencoder with patchified skeletons for motion synthesis, in: Proceedings of the AAAI Conference on Artificial Intelligence, 2024, pp. 5261–5269.
[128] T. He, T. Yang, S. Konomi, Human motion enhancement and restoration via unconstrained human structure learning, Sensors 24 (10) (2024) 3123.
[129] T. He, T. Yang, S. Konomi, Optimizing motion completion with unconstrained human skeleton structure learning, in: 2024 IEEE International Conference on Systems, Man, and Cybernetics, SMC, IEEE, 2024, pp. 3255–3262.
[130] Y. Du, R. Kips, A. Pumarola, S. Starke, A. Thabet, A. Sanakoyeu, Avatars grow legs: Generating smooth human motion from sparse tracking inputs with diffusion model, Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. 2023-June (2023) 481–490, http://dx.doi.org/10.1109/CVPR52729.2023.00054, arXiv:2304.08577.
[131] D. Yan, Q. Gao, Y. Qian, X. Chen, C. Fu, Y. Leng, D3PRefiner: A diffusion-based denoise method for 3D human pose refinement, 2024, arXiv preprint arXiv:2401.03914.
[132] S. Zhang, B.L. Bhatnagar, Y. Xu, A. Winkler, P. Kadlecek, S. Tang, F. Bogo, Rohm: Robust human motion reconstruction via diffusion, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2024, pp. 14606–14617.
[133] J. Wang, Z. Cao, D. Luvizon, L. Liu, K. Sarkar, D. Tang, T. Beeler, C. Theobalt, Egocentric whole-body motion capture with fisheyevit and diffusion-based motion refinement, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2024, pp. 777–787.
[134] C. Bozzini, M. Boldo, E. Martini, N. Bombieri, Late breaking results: A real-time diffusion-based filter for human pose estimation on edge devices, in: Proceedings of the 61st ACM/IEEE Design Automation Conference, 2024, pp. 1–2.
[135] C. Bozzini, M. Boldo, E. Martini, N. Bombieri, A real-time filter for human pose estimation based on denoising diffusion models for edge devices, in: 2024 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS, IEEE, 2024, pp. 10864–10869.
[136] P. Skurowski, M. Pawlyta, Gap reconstruction in optical motion capture sequences using neural networks, Sensors 21 (18) (2021) 6115, http://dx.doi.org/10.3390/s21186115, URL https://www.mdpi.com/1424-8220/21/18/6115.
[137] W. Yin, H. Yin, D. Kragic, Graph-based normalizing flow for human motion generation and reconstruction, IEEE Int. Conf. Robot. Hum. Interact. Commun. (2021) 1–8.
[138] H. Wang, E.S. Ho, H.P. Shum, Z. Zhu, Spatio-temporal manifold learning for human motions via long-horizon modeling, IEEE Trans. Vis. Comput. Graphics 27 (1) (2021) 216–227, http://dx.doi.org/10.1109/TVCG.2019.2936810, arXiv:1908.07214.
[139] A. D'Eusanio, A. Simoni, S. Pini, G. Borghi, R. Vezzani, R. Cucchiara, Depth-based 3D human pose refinement: Evaluating the refinet framework, Pattern Recognit. 171 (2023) 185–191, http://dx.doi.org/10.1016/j.patrec.2023.03.005.
[140] Z. Wang, J. Wang, N. Ge, J. Lu, HiMoReNet: A hierarchical model for human motion refinement, IEEE Signal Process. Lett. 30 (2023) 868–872, http://dx.doi.org/10.1109/LSP.2023.3295756.
[141] Z. Dang, T. Fan, B. Zhao, X. Shen, L. Wang, G. Zhang, Z. Cui, MoManifold: Learning to measure 3D human motion via decoupled joint acceleration manifolds, 2024, arXiv preprint arXiv:2409.00736.
[142] P.S. Kalekar, et al., Time series forecasting using holt-winters exponential smoothing, Kanwal Rekhi Sch. Inf. Technol. 4329008 (13) (2004) 1–13.
[143] S. Butterworth, et al., On the theory of filter amplifiers, Wirel. Eng. 7 (6) (1930) 536–541.
[144] Qualisys, Qualisys getting started guide, 2011, URL https://docs.qualisys.com/getting-started/content/37_trajectory_editor_series/37c_how_to_use_the_trajectory_editor_-_smoothing/smoothing_types.htm.
[145] R. Khusainov, D. Azzi, I. Achumba, S. Bersch, Real-time human ambulation, activity, and physiological monitoring: Taxonomy of issues, techniques, applications, challenges and limitations, Sensors 13 (10) (2013) 12852–12902, http://dx.doi.org/10.3390/s131012852.
[146] J.O. Ramsay, B.W. Silverman, Smoothing functional data by least squares, in: Springer Series in Statistics, Springer-Verlag, 2005, pp. 59–79.
[147] J. Luo, K. Ying, J. Bai, Savitzky–golay smoothing and differentiation filter for even number data, Signal Process. 85 (7) (2005) 1429–1434, http://dx.doi.org/10.1016/j.sigpro.2005.02.002.
[148] J.F. Steffensen, Interpolation, Courier Corporation, 2006.
[149] D. Wells, Trajectory gap filling on vicon nexus, 2018, URL https://logemas.com/knowledge-base/faq/trajectory-gap-filling/.
[150] R.E. Kalman, A new approach to linear filtering and prediction problems, Trans. ASME– J. Basic Eng. 82 (Series D) (1960) 35–45.
[151] A.M. Lehrmann, P.V. Gehler, S. Nowozin, Efficient nonlinear Markov models for human motion, in: 2014 IEEE Conference on Computer Vision and Pattern Recognition, IEEE, 2014, pp. 1314–1321, http://dx.doi.org/10.1109/cvpr.2014.171.
[152] E. G.A., Smoothing, Filtering and Prediction - Estimating The Past, Present and Future, InTech, 2012, http://dx.doi.org/10.5772/2706.
[153] S.J. Julier, J.K. Uhlmann, New extension of the Kalman filter to nonlinear systems, in: I. Kadar (Ed.), SPIE Proceedings, SPIE, 1997, pp. 182–193, http://dx.doi.org/10.1117/12.280797.
[154] S.J. Julier, J.K. Uhlmann, Unscented filtering and nonlinear estimation, Proc. IEEE 92 (3) (2004) 401–422.
[155] M.C. VanDyke, J.L. Schwartz, C.D. Hall, et al., Unscented Kalman filtering for spacecraft attitude state and parameter estimation, Adv. Astronaut. Sci. 118 (1) (2004) 217–228.
[156] F. Gustafsson, G. Hendeby, Some relations between extended and unscented Kalman filters, IEEE Trans. Signal Process. 60 (2) (2011) 545–555.
[157] B. Allik, The Tobit Kalman Filter: an Estimator for Censored Data, University of Delaware, 2014.
[158] P. Bühlmann, S. Van De Geer, Statistics for High-Dimensional Data: Methods, Theory and Applications, Springer Science & Business Media, 2011.

[159] J. Chai, J.K. Hodgins, Performance animation from low-dimensional control signals, ACM Trans. Graph. 24 (3) (2005) 686–696, http://dx.doi.org/10.1145/1073204.1073248.

[160] K. Miura, H. Furukawa, M. Shoji, Similarity of human motion: Congruity between perception and data, 2006, http://dx.doi.org/10.1109/icsmc.2006.384875.

[161] M. Ding, G. Fan, Multilayer joint gait-pose manifolds for human gait motion modeling, IEEE Trans. Cybern. 45 (11) (2014) 2413–2424.

[162] V. Klema, A. Laub, The singular value decomposition: Its computation and some applications, IEEE Trans. Autom. Control 25 (2) (1980) 164–176, http://dx.doi.org/10.1109/TAC.1980.1102314.

[163] C. Eckart, G. Young, The approximation of one matrix by another of lower rank, Psychometrika 1 (3) (1936) 211–218, http://dx.doi.org/10.1007/bf02288367.

[164] J.-F. Cai, E.J. Candes, Z. Shen, A singular value thresholding algorithm for matrix completion, 2008, arXiv:0810.3286.

[165] E.J. Candes, Y. Plan, Matrix completion with noise, Proc. IEEE 98 (6) (2010) 925–936, http://dx.doi.org/10.1109/JPROC.2009.2035722.

[166] Z. Lin, M. Chen, Y. Ma, The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices, 2010, arXiv preprint arXiv:1009.5055.

[167] F. Meng, X. Yang, C. Zhou, The augmented Lagrange multipliers method for matrix completion from corrupted samplings with application to mixed Gaussian-impulse noise removal, PLoS One 9 (9) (2014) e108125, http://dx.doi.org/10.1371/journal.pone.0108125.

[168] E.J. Candes, X. Li, Y. Ma, J. Wright, Robust principal component analysis?, 2009, http://dx.doi.org/10.48550/ARXIV.0912.3599, URL https://arxiv.org/abs/0912.3599.

[169] K.-C. Toh, S. Yun, An accelerated proximal gradient algorithm for nuclear norm regularized linear least squares problems, Pac. J. Optim. 6 (615–640) (2010) 15.

[170] D.D. Lee, H.S. Seung, Learning the parts of objects by non-negative matrix factorization, Nature 401 (6755) (1999) 788–791.

[171] T. Hastie, R. Tibshirani, J.H. Friedman, J.H. Friedman, The Elements of Statistical Learning: Data Mining, Inference, and Prediction, vol. 2, Springer, 2009.

[172] K. Engan, S. Aase, J. Hakon Husoy, Method of optimal directions for frame design, in: 1999 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings. ICASSP99 (Cat. No.99CH36258), vol. 5, 1999, pp. 2443–2446 vol.5, http://dx.doi.org/10.1109/ICASSP.1999.760624.

[173] M. Aharon, M. Elad, A. Bruckstein, K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation, IEEE Trans. Signal Process. 54 (11) (2006) 4311–4322, http://dx.doi.org/10.1109/TSP.2006.881199.

[174] J. Martinez, R. Hossain, J. Romero, J.J. Little, A simple yet effective baseline for 3d human pose estimation, in: ICCV, 2017.

[175] S.L. Brunton, J.N. Kutz, Data-Driven Science and Engineering, Cambridge University Press, Cambridge, England, 2019.

[176] I. Goodfellow, Y. Bengio, A. Courville, Deep Learning, MIT Press, 2016, URL http://www.deeplearningbook.org.

[177] R. Reed, R.J. Marks II, Neural Smithing: Supervised Learning in Feedforward Artificial Neural Networks, Mit Press, 1999.

[178] S. Ladjal, A. Newson, C.-H. Pham, A PCA-like autoencoder, 2019, arXiv preprint arXiv:1904.01277.

[179] M. Loper, N. Mahmood, J. Romero, G. Pons-Moll, M.J. Black, SMPL: A skinned multi-person linear model, ACM Trans. Graph. (Proc. SIGGRAPH Asia) 34 (6) (2015) 248:1–248:16.

[180] G. Lai, W.-C. Chang, Y. Yang, H. Liu, Modeling long- and short-term temporal patterns with deep neural networks, 2017, http://dx.doi.org/10.48550/ARXIV.1703.07015, URL https://arxiv.org/abs/1703.07015.

[181] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, 2018, arXiv preprint arXiv:1810.04805.

[182] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, et al., An image is worth 16x16 words: Transformers for image recognition at scale, 2020, arXiv preprint arXiv:2010.11929.

[183] J. Ho, A. Jain, P. Abbeel, Denoising diffusion probabilistic models, Adv. Neural Inf. Process. Syst. 33 (2020) 6840–6851.

[184] R. Storn, K. Price, Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces, J. Global Optim. 11 (4) (1997) 341–359, http://dx.doi.org/10.1023/a:1008202821328.

[185] P. Das, K. Chakravarty, D. Chatterjee, A. Sinha, Improvement in kinect based measurements using anthropometric constraints for rehabilitation, IEEE Int. Conf. Commun. (2017) http://dx.doi.org/10.1109/ICC.2017.7996969.

[186] P. Das, K. Chakravarty, A. Chowdhury, D. Chatterjee, A. Sinha, A. Pal, Improving joint position estimation of kinect using anthropometric constraint based adaptive Kalman filter for rehabilitation, Biomed. Phys. Eng. Express 4 (3) (2018) 035002, http://dx.doi.org/10.1088/2057-1976/aaa371.

[187] S.R. Tripathy, K. Chakravarty, A. Sinha, Constrained particle filter for improving kinect based measurements, Eur. Signal Process. Conf. 2018-Septe (2018) 306–310, http://dx.doi.org/10.23919/EUSIPCO.2018.8553437.

[188] L. Zhou, N. Lannan, G. Fan, Joint optimization of kinematics and anthropometrics for human motion denoising, IEEE Sens. J. 22 (5) (2022) 4386–4399, http://dx.doi.org/10.1109/JSEN.2022.3144946.

[189] M. Burke, J. Lasenby, Estimating missing marker positions using low dimensional Kalman smoothing, J. Biomech. 49 (9) (2016) 1854–1858, http://dx.doi.org/10.1016/j.jbiomech.2016.04.016.

[190] Y. Park, S. Moon, I.H. Suh, Tracking human-like natural motion using deep recurrent neural networks, 2016, http://dx.doi.org/10.48550/ARXIV.1604.04528, URL https://arxiv.org/abs/1604.04528.

[191] H. Coskun, F. Achilles, R. Dipietro, N. Navab, F. Tombari, Long short-term memory Kalman filters: Recurrent neural estimators for pose regularization, Proc. IEEE Int. Conf. Comput. Vis. 2017-Octob (2017) 5525–5533, http://dx.doi.org/10.1109/ICCV.2017.589, arXiv:1708.01885.

[192] N. Lannan, G. Fan, Filter guided manifold optimization in the autoencoder latent space, IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. Work. 2019-June (c) (2019) 949–955, http://dx.doi.org/10.1109/CVPRW.2019.00125.

[193] N. Lannan, L. Zhou, G. Fan, J. Hausselle, Human motion enhancement using nonlinear Kalman filter assisted convolutional autoencoders, Proc. - IEEE 20th Int. Conf. Bioinform. Bioeng. BIBE 2020 (2020) 1008–1015, http://dx.doi.org/10.1109/BIBE50027.2020.00171.

[194] N. Lannan, L. Zhou, G. Fan, Human motion enhancement via tobit Kalman filter-assisted autoencoder, IEEE Access 10 (2022) 29233–29251, http://dx.doi.org/10.1109/ACCESS.2022.3157605.

[195] E. Martini, M. Boldo, N. Bombieri, FLK: A filter with learned kinematics for real-time 3D human pose estimation, Signal Process. 224 (2024) 109598.

[196] Wei Shen, Ke Deng, Xiang Bai, T. Leyvand, Baining Guo, Zhuowen Tu, Exemplar-based human action pose correction and tagging, in: 2012 IEEE Conf. Comput. Vis. Pattern Recognit., vol. 44, IEEE, 2012, pp. 1784–1791, http://dx.doi.org/10.1109/CVPR.2012.6247875, URL http://ieeexplore.ieee.org/document/6247875/.

[197] P. Skurowski, M. Pawlyta, Tree based regression methods for gap reconstruction of motion capture sequences, Biomed. Signal Process. Control. 88 (2024) 105641.

[198] L. Breiman, J. Friedman, R.A. Olshen, C.J. Stone, Classification and Regression Trees, Routledge, 2017.

[199] Y. Wang, I.H. Witten, Inducing model trees for continuous classes, in: Proceedings of the Ninth European Conference on Machine Learning, vol. 9, Citeseer, 1997, pp. 128–137, 1.

[200] J. Ye, J.-H. Chow, J. Chen, Z. Zheng, Stochastic gradient boosted distributed decision trees, in: Proceedings of the 18th ACM Conference on Information and Knowledge Management, 2009, pp. 2061–2064.

[201] L. Breiman, Bagging predictors, Mach. Learn. 24 (1996) 123–140.

[202] L. Zhou, Z. Liu, H. Leung, H.P. Shum, Posture reconstruction using kinect with a probabilistic model, Proc. ACM Symp. Virtual Real. Softw. Technol. VRST (2014) 117–125, http://dx.doi.org/10.1145/2671015.2671021.

[203] Z. Liu, L. Zhou, H. Leung, H.P. Shum, Kinect posture reconstruction based on a local mixture of Gaussian process models, IEEE Trans. Vis. Comput. Graphics 22 (11) (2016) 2437–2450, http://dx.doi.org/10.1109/TVCG.2015.2510000.

[204] A.T. Chiang, Q. Chen, S. Li, Y. Wang, M. Fu, Denoising of joint tracking data by kinect sensors using clustered Gaussian process regression, MMHealth 2017 - Proc. 2nd Int. Work. Multimed. Pers. Heal. Heal. Care, Co- Located MM 2017 (2017) 19–25, http://dx.doi.org/10.1145/3132635.3132642.

[205] J. Baumann, B. Krüger, A. Zinke, A. Weber, Data-driven completion of motion capture data, 2011, http://dx.doi.org/10.2312/PE/VRIPHYS/VRIPHYS11/111-118, URL http://diglib.eg.org/handle/10.2312/PE.vriphys.vriphys11.111-118.

[206] H. Yasin, S. Ghani, B. Krüger, An effective and efficient approach for 3D recovery of human motion capture data, Sensors 23 (7) (2023) 3664, http://dx.doi.org/10.3390/s23073664, URL https://www.mdpi.com/1424-8220/23/7/3664.

[207] P. Plantard, H.P. Hubert, F. Multon, Filtered pose graph for efficient kinect pose reconstruction, Multimedia Tools Appl. 76 (3) (2017) 4291–4312, http://dx.doi.org/10.1007/s11042-016-3546-4.

[208] Carnegie mellon university - CMU graphics lab - motion capture library, 2016, http://mocap.cs.cmu.edu/tools.php. (Accessed 31 March 2023).

[209] M. Müller, T. Röder, M. Clausen, B. Eberhardt, B. Krüger, A. Weber, Documentation Mocap Database HDM05, Tech. Rep. CG-2007-2, Universität Bonn, 2007.