Efficient Unsupervised Temporal Segmentation of Motion Data

Björn Krüger, Anna Vögele, Tobias Willig, Angela Yao, Reinhard Klein, *Member, IEEE*, and Andreas Weber

***Abstract*—We introduce a method for automated temporal segmentation of human motion data into distinct actions and compositing motion primitives based on self-similar structures in the motion sequence. We use neighborhood graphs for the partitioning and the similarity information in the graph is further exploited to cluster the motion primitives into larger entities of semantic significance. The method requires no assumptions about the motion sequences at hand and no user interaction is required for the segmentation or clustering. In addition, we introduce a feature bundling preprocessing technique to make the segmentation more robust to noise, as well as a notion of motion symmetry for more refined primitive detection. We test our method on several sensor modalities, including markered and markerless motion capture as well as on electromyograph and accelerometer recordings. The results highlight our system's capabilities for both segmentation and for analysis of the finer structures of motion data, all in a completely unsupervised manner.**

***Index Terms*—Temporal segmentation, time series clustering, human motion analysis.**

I. Introduction

HUMAN motion capture, once associated with producing special effects for films and video games, is commonly used today in diverse applications ranging from health care to consumer electronics. The ever-increasing simplicity to capture data by different sensor modalities, along with the sheer amount of existing recorded data creates a demand for motion analysis methods that are computationally efficient, yet with minimal human input. Dividing streams of motion data into perceptually meaningful segments is a precursor to almost all analysis and synthesis methods. For example, creating a statistical motion model calls for data already preprocessed into well-defined activity segments. Further segmentation of these activities into individual cycles is greatly beneficial for action recognition, especially when repetitions should be counted, or for compressing motion data. However, the quantity of captured data does not always allow for time-consuming manual segmentation. As such, unsupervised segmentation and learning of motion primitives is a topic of interest that has been addressed in the multimedia [1]–[4], computer vision [5]–[12], and computer graphics [13]–[17] communities.

We propose a segmentation method which identifies *distinct actions* within motion sequences and further decompose such actions into atomic *motion primitives*, all in an unsupervised fashion. For example, in a sequence of a person who first walks and then breaks into a run, we can separate walking from running, as well as the individual steps of the walk and run. We identify both the actions and the motion primitives by exploiting the self-similarities that exist in the motion sequences.

We pose segmentation as an efficiently solvable graph problem, as first presented in [16] for segmenting motion capture data. To further improve computational efficiency, we employ a Neighborhood Graph [18]. In addition, we propose three new contributions, making the method robust and applicable to motion data from varying sensor modalities. First, we propose a novel feature bundling technique for preprocessing motion features. The feature bundling allows for compact model representations of the motions, as well as robustness to noise, to accommodate modalities such as markerless motion capture or accelerometers. Second, we introduce a notion of motion symmetry, and exploit this as a means of refining primitive detection. Considering symmetry often leads to primitives with more physical meaning—for instance walking cycles can be split into left and right steps. Third, we propose a unique clustering method for the detected motion segments also based on self similarities which needs no assumption on the number of clusters. We show the applicability of our method on a wide variety of motion datasets, ranging from markered and markerless motion capture to accelerometer and surface electromyography (EMG) recordings.

Defining segments based on self-similarity gives our method several advantages over previous segmentation methods. First, it allows us to distinguish not only the distinct action segments, but also the transition segments between actions. Explicit treatment of transitions has not been addressed so far in previous works on unsupervised segmentation [9], [10], [19]. However, it has a direct impact on synthesis methods [19], [20]. Not handling the transitions forces synthesized sequences to include additional

Manuscript received April 21, 2016; revised September 8, 2016 and November 24, 2016; accepted November 24, 2016. Date of publication December 9, 2016; date of current version March 15, 2017. This work was supported in part by Deutsche Forschungsgemeinschaft under Research Grant KR 4309/2-1. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Dong Xu.

B. Krüger is with the Gokhale Method Institute, Stanford, CA 94305 USA (e-mail: kruegerb@cs.uni-bonn.de).

A. Vögele, T. Willig, A. Yao, R. Klein, and A. Weber are with the Department of Computer Science, University of Bonn, Bonn 53113, Germany (e-mail: voegele@cs.uni-bonn.de; twillimail@gmail.com; yao@cs.uni-bonn.de; rk@cs.uni-bonn.de; weber@cs.uni-bonn.de).

This paper has supplementary downloadable material available at http://ieeexplore.ieee.org provided by the authors. This includes a video that shows the relationship between motion sequences and the sparse self-similarity matrix, as well as some motions that have been segmented and clustered using the method described in the paper. This material is 69.7 MB in size.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TMM.2016.2635030

1520-9210 © 2016 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

Authorized licensed use limited to: FH Koeln. Downloaded on November 22,2022 at 10:34:35 UTC from IEEE Xplore. Restrictions apply.

primitives from possibly unrelated transitions, which is neither convenient nor intrinsically motivated.

The second advantage of leveraging self similarity is that the resulting motion primitives are highly consistent, with similar start and end points from sequence to sequence. Previous approaches often yield primitives which are phase-shifted from one another [10]. Statistical models built on unaligned primitives will be noisier and less representative of the actual motion. Note that our motion primitives are not limited to being full motion cycles i.e. those starting and ending in the same body pose, but can also be parts of a cycle or entirely non-cyclic in nature.

The rest of this paper is organized as follows. Section III introduces the feature bundling technique while Section IV describes the motion segmentation into actions and subsequent motion primitives. In Section V, we introduce a notion of symmetry to help refine the segmented motion primitives. In Section VI, the clustering of the motion primitives is discussed. We present segmentation results from motion sequences of varying modalities in Section VII, and conclude by discussing the achievements and limitations of our novel method as well as possible extensions in Section VIII. Source code for the algorithm is available online.

II. Related Work

Temporal segmentation is relevant in a number of different fields such as data mining [21], [22], audio and speech processing [23]–[25], and behavioural pattern recognition [26]. In statistical terms, the segmentation problem can be posed as a change point detection task [22], which has been extended to a multi-dimensional setting [26] based on the established Bayesian techniques [27]. From a signal processing point of view, kernel-based methods for change-point analysis [28], [29], Hidden Markov Models (HMMs) [23], and audio thumbnailing [24], [30], have been used. Below, we outline attempts to automatically segment human motion data.

Pose clustering: One segmentation strategy is based on clustering poses in a time series [3], [31]–[35]. Beaudoin *et al.* [31] proposed to extract motion motifs as building blocks of graphs. Chew *et al.* [32] come up with a fuzzy clustering approach to compress motion data. Gall *et al.* [33] and Kadu and Kuo [3] create temporally meaningful pose clusters associated with unlabeled actions. Bernard *et al.* [34] proposed a search and analysis system called *MotionExplorer* based on similarity features. Depending on the aggregation level, shorter or longer motion segments were found to be associated with isolated human actions. However, Bernard's main focus was on a new visual representation of motion data rather than analysis tasks as discussed here. Li *et al.* [35] introduced a cluster in a regularized subspace to tackle segmentation.

Exemplars and template models: An alternative approach for segmentation is to apply example segments or pre-computed templates and match them to test sequences. For example, Lv *et al.* [36] combines HMMs and AdaBoost to learn discriminative feature templates from labelled segments for action recognition and segmentation.

Müller *et al.* [14], [37], [38] used geometric features to learn templates for fast solving of matching problems such as annotation and retrieval [37], [38]. Adaptive segmentation [14] is a fundamental result of using geometric feature vector sequences to compare motion capture data streams at a segment rather than frame level. Another set of related approaches [39], [40] learn intrinsic regularities for segmentation and demonstrate that motion capture data can be segmented using only a limited set of example motions, even if the examples are of different actions. Template approaches work well if the templates are available; our work is targeted at cases in which the exemplars or templates are not known beforehand.

Motion synthesis: Segmentation has also been addressed in conjunction with motion synthesis. In motion concatenation [20], a *motion graph* is constructed from clips of motion capture data; new sequences are then synthesized by motion extraction on this graph. In motion parameterization [41], motion elements are retrieved from large datasets based on similarity to a query motion, and then blended according to user constraints. The novel distance relation used in [41] has become the standard for finding similar motion clips at interactive speeds.

Later works [19], [42]–[44] combine these ideas to accomplish synthesis techniques for high quality interactive applications. Min and Chai's Motion Graphs++ [19] effectively enables motion segmentation, recognition and online synthesis. All these approaches have a need for meaningful motion primitives that can be clustered to build statistical motion models or at least to allow for interpolation. Typically these are found by manually selecting some examples and then retrieving similar exemplars from a database.

Unsupervised motion segmentation: The methods most similar to ours are those which segment motions in an unsupervised way. Partitioning motion sequences into behaviour segments by a PCA-based method was proposed by Barbic *et al.* [13]. This segmentation is similar in spirit to the first step of our approach for isolating distinct actions. The quality of local PCA models is tracked temporally; new activities are defined when the old PCA model cannot capture the data variance and a new PCA model is required. This approach cannot separate activities that fit into one local model and also cannot detect individual representations. Jones *et al.* [45] combine a PCA approach with linear regression to derive a metric that can be used to segment unknown motion sequences.

The groundbreaking work of Zhou *et al.* [9], [10] uses (hierarchically) aligned cluster analysis (H)ACA to temporally group poses into motion primitives which are then assigned to different action classes. These approaches use kernel-based projections and a time alignment to compare motion primitives of varying length. An initial segmentation of uniform length is refined by the clustering approach though the final resulting segments do not vary much from the initial length. A major difference between the work of Zhou *et al.* and ours is that they do not consider transitions between distinct actions. Transition frames are assigned to previous or following action segments, thereby reducing the consistency of the primitives. More recently, Osmanlıoğlu *et al.* [46] used metric embedding in combination with hierarchically separated trees to segment video sequences in the spatial and temporal domain.

Authorized licensed use limited to: FH Koeln. Downloaded on November 22,2022 at 10:34:35 UTC from IEEE Xplore. Restrictions apply.

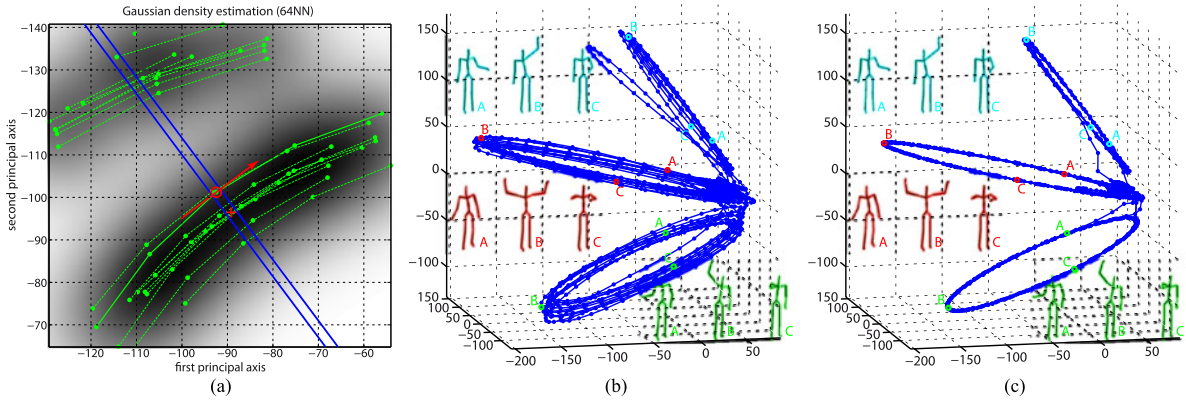


Fig. 1. (a) 2D example for *density estimation*: original data points (red circle), 64 *nearest neighbors* (green dots connected in time), the *direction of movement* (red arrow), the 1D *subspace* used for optimization (blue lines), and the resulting position (red cross). (b) 3D *PCA projection* of the feature sets of a mocap sequence (CMU 86_11). The three sections of loops correspond to repetitions of differing arm rotation movements as indicated by representative body poses. (c) *Bundled feature points* for the same motion capture sequence. Figure is best viewed in color.

III. Feature Bundling

Semantically similar or visually similar motions, even when represented in dedicated feature spaces, may still differ notably due to variations in the performance of each action or cycle within the same action. Inspired by the idea of edge bundling used in visualization [47], [48], where similar edges of a graph are visualized together for a better overview, we propose a bundling of similar features. The goal of feature bundling is to topologically align disjoint motions that belong to the same action class but exhibit differences in the feature space. As opposed to filtering, bundling deals with differences due to performance variation in recorded motions. Note that our feature bundling technique has completely different objectives and methodology from *bundle adjustments* used in 3D reconstruction and is similar in name only.

In our bundling technique we use a density estimation based optimization to adjust each point orthogonally to the direction of its trajectory in the feature space. This effectively project the features onto a smoother manifold (see Fig. 1 for an overview). Vejdemo-Johansson *et al.* [49] considered a related idea by computing a typical motion cycle of a set of similar periodic motions.

For each frame i of an input sequence of length N, a feature vector F_i where $i \in [1 \dots N]$ is computed; the specific feature depends on the sensor modality (more details in Section VII). The goal is to compute a new feature $\hat{F}_i$ that is representative of F_i, but closer to other features at the same stage of the same motion cycle. Specifically,

1) For each F_i, find k *nearest neighbors* within all other features F_j, $j \in [1 \dots N] \setminus i$.
2) Compute *direction of movement* d_i for F_i and build a D–1 dimensional *subspace* $\mathcal{S}_i$ orthogonal to d_i.
3) *Optimize* $\hat{F}_i$ using a *kernel density estimate* of F_i in $\mathcal{S}_i$ based on the k nearest neighbors.
4) Backproject $\hat{F}_i$ to the original feature space for the final representation.

For the **k**-*nn search* we use a kd-tree as per [18] to find similar frames within the motion sequence. We do not fix the search radius since distances between sample points may vary greatly. Instead, a fixed number of k nearest neighbors ensures that the model reflects the local density of samples.

The *direction of movement* d_i for frame i is given by the numerically centered five-point derivative at this frame. Constructing an orthogonal *subspace* prevents the overall data structure from collapsing. The basis of this subspace is computed via QR decomposition of a $D \times D$ matrix, where the first column vector is set to the direction of movement, while all other entries are filled with random numbers.

The local kernel *density estimation* is based on the k nearest neighbors and characterizes the data distribution of F_i with kernel function K_H, i.e. a symmetric multivariate density with bandwidth matrix H

$$K_{H^i}(x) = |H^i|^{-\frac{1}{2}} K\left(H^{i^{-\frac{1}{2}}} x\right). \tag{1}$$

Since our data is assumed to be Gaussian, we use a general approximation of the bandwidth matrix which minimizes the mean integrated squared error (MISE; refer to Scott's rule in Chapter 6 of [50]) by

$$H_{jj} = \sigma_j k^{\frac{-1}{d+4}} \tag{2}$$

where $j = 1, \dots, d$ and σ_j is the standard deviation of the j^{th} variate. We use the multivariate Gaussian kernel

$$K(x, \mu, \sigma) = e^{-\frac{1}{2}(x-\mu)'\Sigma^{-1}(x-\mu)} \tag{3}$$

with $x = (x_1, \dots, x_d)$, $\mu = (\mu_1, \dots, \mu_d)$ the vector of empirical mean values, and Σ the sample covariance matrix.

Note that the kernel density estimation is compatible with a preceding dimensionality reduction step. In fact, it is possible to reliably estimate the kernel density function without increasing the sample size k [50]. Therefore, we apply a PCA to each set of samples in advance to reduce the dimensionality while keeping 97.5% of the sample set's variance. In principle, any dimensionality reduction technique is applicable; we prefer PCA for its simplicity and speed, though other approaches have been explored extensively [13], [51], [52].

The *optimization* searches for the new position $\hat{F}_i$ and is posed as an minimization problem. The offset O_i from F_i is

Authorized licensed use limited to: FH Koeln. Downloaded on November 22,2022 at 10:34:35 UTC from IEEE Xplore. Restrictions apply.

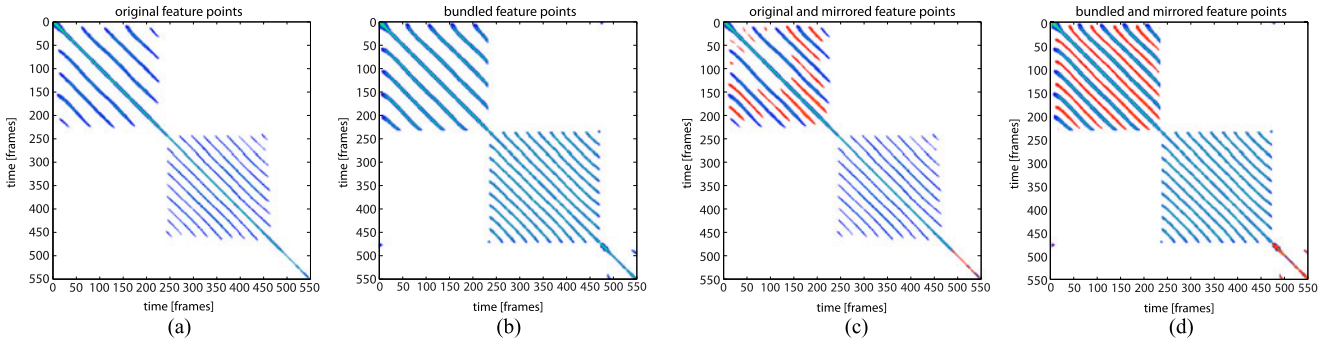


Fig. 2. Comparison of SSSMs with original and bundled features: (a) shows the SSSM of original features extracted from the input motion; (b) shows the same SSSM of the bundled features; (c) and (d) include mirrored features (highlighted in red) of the original and bundled features respectively. Note how the SSSM in (c) based on the original mirrored features is less consistent than that of the bundled features in (d).

optimized with respect to the density estimation, resulting in the final feature position $\hat{F}_i$

$$\hat{F}_i = \underset{O_i}{\operatorname{argmin}} K_{H^i}(F + O_i). \tag{4}$$

IV. Segmentation Technique

The input to our method is a multi-dimensional time series recording of a motion trial. First, the local neighbors of each frame in the sequence are found. The sequence is then partitioned into distinct temporally coherent action segments; a subsequent step partitions the actions based on recurring patterns, i.e. shorter *motion primitives*.

A. Local Neighbors

A motion sequence M is given as a time series of m points $p_1, \ldots, p_m$, each represented by a feature vector $F = (f_1, \ldots, f_N)$ of dimension N. The features are modality specific and stacked in the time dimension, yielding a vector representation $[p_{i-f_1}, p_i, p_{i+f_2}]'$ of features over a window $w = [i - f_1, i, i + f_2]$ in a higher dimension.

Within the feature space, we define a radius r to search for neighbors. Given no prior knowledge of the input data, we introduce a *generalized search radius* R which is independent of the feature set time-window size w and input data dimensionality N. R is defined as the search radius for a window size of $|w| = 1$ and dimensionality of $N = 1$; the search radius is then rescaled as $r = R\sqrt{|w| \cdot N}$.

We construct a *kd-tree* from all feature points F_i in the input stream and then search for the points located within the radius r based on the Euclidean distance d_{ij} between the features F_i of p_i and F_j of p_j. As a result of this search, we obtain a set S_i of neighbors for each data point p_i. The neighbors are specified as pairs $(j \in [1 : M], d_{ij})$ of an index j to a frame in the input motion and the distance d_{ij} between the query point and neighbor j.

The sets of neighbors for a motion trial is then converted into a *sparse self similarity matrix (SSSM)*. Self-similarity matrices are commonly used in human motion analysis for tasks ranging from retrieval [41] to multi-view action recognition [53]. A SSSM, as shown in Fig. 2(a), is generated by initializing an empty $M \times M$ matrix $\mathcal{M}$. For each frame $i \in [1 : M]$ we set the entries $\mathcal{M}_{i,j} \forall k \in S_i$ to the values of d_{ij} stored in S_i. An illustration of this connection is also given in Fig. 4(a) and 4(b). An example of the effect of feature bundling to the neighborhoods is given in Fig. 1(a). A 3D projection of input and output feature points of a motion sequence are given in Fig. 1(b) and 1(c), respectively. A comparison between a sparse self similarity matrix (SSSM) based on the original and the bundled features is given in Fig. 2(a) and 2(b) respectively. Note that we use the matrices only for visualization in this work; for efficiency purposes, computations are performed directly on the sets of neighbors or derived data structures.

B. Segmentation Into Distinct Activities

Fig. 3 shows an example of an SSSM with two cyclic activities, running and jumping, separated by an 'uncertain' period of inconclusive user annotations. The cyclic activities are characterized by structured diagonal blocks. We separate activities by searching for these characteristic blocks, using *region growing* to determine the blocks' borders.

A connected region starts as a seed in the upper left corner of the neighborhood representation matrix $\mathcal{M}_{1,1}$. Contents of the lower triangular matrix below the main diagonal are then probed using scan lines. The triangular region is gradually extended to adjacent rows as long as the number of nearest neighbors in the updated region increases. If no new neighbors are found between frame i and $i + w$ in the larger region, the current region is considered complete. The parameter w is introduced to handle noisy data; we set $w = 8$ in all our experiments. With such a stop criterion, neighbors from the main diagonal of the SSSM cannot be considered—otherwise these elements would also be counted in the region-growing and result in a large region covering the entire SSSM. We remove all main diagonal entries in the proximity corresponding to one second (in time), based on observations that cyclic behaviour in motion data does not occur at higher speeds. A new region is then started from the upper left entry of the remaining matrix $\mathcal{M}_{i,i}$, with scan lines probing the content of the lower triangular matrix between $\mathcal{M}_{i,i}$, $\mathcal{M}_{i,j}$ and $\mathcal{M}_{j,j}$, where j is the current frame.

The region growing is performed once as a *forward step*, seeding the first region at frame $\mathcal{M}_{1,1}$ of the matrix to identify the

Authorized licensed use limited to: FH Koeln. Downloaded on November 22,2022 at 10:34:35 UTC from IEEE Xplore. Restrictions apply.

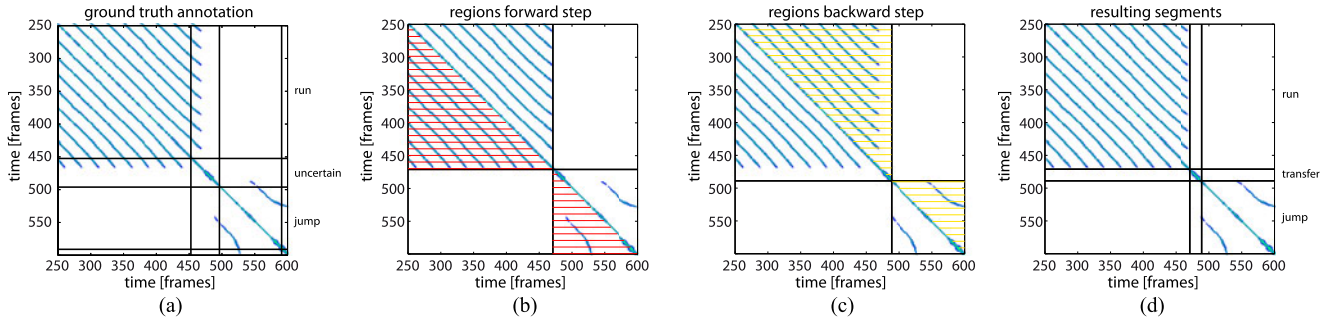


Fig. 3. Region growing for activity separation. (a) Sparse self similarity matrix with ground truth annotations; the label "uncertain" indicates an area of inconclusive user annotations. (b) Same matrix without main diagonal; also results for forward step of region growing. (c) Results backward step of region growing. (d) Final outcome of segmentation. Note that in the actual computation of the region growing steps (b) and (c) the main diagonal is removed. It is displayed in this representation for visualization purposes though.

end of repetitive patterns [see Fig. 3(b)] and once as a *backward step*, seeding at the last frame $\mathcal{M}_{n,n}$ of the input sequence first to identify the start of the actions [Fig. 3(c)]. The lower right corners of the forward region growing correspond to end frames of an action, while the upper left corners of the backwards step correspond to the start frames of an action. Areas in between are considered transitions between the repetitive parts.

For efficiency, we work on the sets S_i of neighbors, counting the number of entries in the neighborhoods between the seed frame and the current scan line index. Because we work with a symmetric matrix, this is equivalent to scanning triangular parts of the sparse similarity matrix.

Compared to the region growing approach of Vögele *et al.* [16], where neighbors were counted in a quadratic region, the method presented here is computationally more efficient. For each scan line, only one set of nearest neighbors needs to be considered, whereas in [16], all preceding neighbor sets were reconsidered for each growth step. The runtime complexity is therefore reduced from $O\left(k\frac{n(n-1)}{2}\right)$ to $O(kn)$ in our approach, where k is the maximum number of nearest neighbors and n is the number of frames of the motion trial; in the worst case the first region grows over the whole SSSM.

C. Subdividing Actions Into Motion Primitives

After segmenting the input sequence into distinct actions, we search for primitives within each action. Consider a single action, such as walking in Fig. 5; we want to find the reoccurring units, i.e. steps of the activity. Such units are responsible for the minor diagonals in the SSSM of the specific activity, with start and end frames of each unit corresponding to the start and end position of a diagonal. Rather than searching for the diagonals' starts and ends in the SSSM, we use an alternative neighborhood graph representation and simplify the problem to finding the shortest path. We note that if no reoccurring primitives are found, the action segment is considered itself a single primitive.

Alternative representation by a neighborhood graph: The neighbors of p_j in a specific activity stored in the set of neighborhoods $S = \{S_i,\ i \in 1, \ldots, n\}$ can be considered nodes on a graph G_{act}. The criteria for connecting the nodes in this graph is based on accessibility between the points and can be characterized by the concept of dynamic time warping.

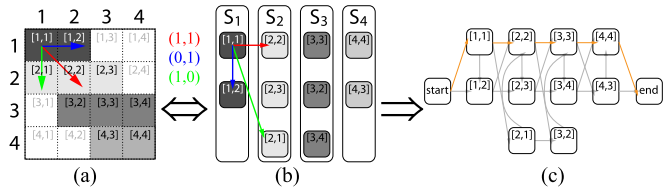


Fig. 4. Toy example illustrating the relationship between the SSSM and the neighborhood graph G_M. (a) SSSM of four consecutive points with allowed steps indicated by arrows. Red arrow: step $(1, 1)$; blue arrow: step $(0, 1)$; green arrow: step $(1, 0)$. (b) Neighborhoods of each of the four points, e. g. S_i is the neighborhood of the point i. (c) Resulting neighborhood graph.

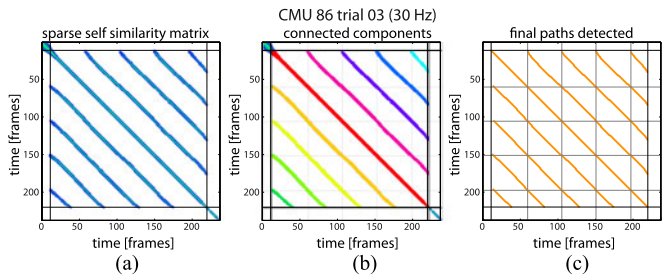


Fig. 5. Illustration of connected components in G_{act}. (a) SSSM corresponding to walking. (b) Same matrix with its connected components colour coded. (c) Optimal warping paths highlighted by orange lines.

Consider two points $p_j \in S_i$ and $p_{j^*} \in S_{i^*}$. A valid time warping step to access the point p_{j^*} from p_j is defined as a pair $(a, b) \in \{(1, 1), (0, 1), (1, 0)\}$ such that $p_{j^*} = p_{j+b}$ and $S_{i^*} = S_{i+a}$. In particular, the point p_{j^*} is always an entry further below in the neighborhood list S_{i^*} than p_j is in the list S_i, while S_{i^*} is either identical to S_i or lies to its right hand side $(S_{i^*} = S_{i+1})$. Fig. 4 shows a toy example to illustrate a possible scenario.

An important property of the graph G_{act} which we exploit lies in the following observation: each diagonal of the matrix $\mathcal{M}_{\text{act}}$ reflecting local similarities is one *connected component* of G_{act}. For the next steps, it is useful to work only with neighbors belonging to the same connected component cc for a given frame, with the resulting restricted graph denoted as G_{cc} (Refer to Fig. 5 for a visualization of connected components).

Computation of warping paths: To calculate an optimal match between two given time series A and B with restrictions,

Authorized licensed use limited to: FH Koeln. Downloaded on November 22,2022 at 10:34:35 UTC from IEEE Xplore. Restrictions apply.

dynamic time warping creates a path between these sequences. The sequences are matched non-linearly in time to optimize for a similarity measure. Technically, a warping path $\mathcal{P}_{A,B}$ of length λ between two such sequences is given as a pair of vectors (v_A, v_B) where $v_A = (a_1, \ldots, a_\lambda)$ with $a_i \in A$ meeting constraints such as $a_i \leq \nu a_{i+1}$ for all indices and $v_B = (b_1, \ldots, b_\lambda)$ with $b_i \in B$ meeting $b_i \leq \nu b_{i+1}$. In our experiments, we use $\nu = 2$. The constraints on v_A and v_B could be seen as upper and lower limit of the paths *slope*.

The sets of neighbors S_i are suitable to replace conventional dynamic time warping based on the neighborhood graph described above; more details on this is given in [18]. Since each diagonal in the SSSM translates to one connected component in the graph, searching for an optimal warping path reduces to finding a shortest path through the connected component G_{cc}. To this end, we add one additional start and end node to the graph. The start node connects to all nodes corresponding to the first set of neighbors in the component, while the end node connects to all nodes corresponding to the last set of neighbors. Now, the warping path can be found efficiently by searching the shortest path from the start to the end node. The costs of a path is the accumulated distance of the included neighbors.

We further limit the set of warping paths per activity based on their length and slope. First, paths covering less than five frames of the motion trial are discarded; such paths are found for very short but similar segments existing between longer primitives. Although these segments may be semantically meaningful, we ignore them to prevent extremely short primitives. Secondly, paths with average gradients less than 0.5 and larger than 2 are also discarded. Such paths represent mapping between motions whose speeds vary by a factor greater than two. We want to avoid such cases, e.g. when a longer standing sequence is mapped to a few poses in the middle of a walking cycle.

For each valid warping path $\mathcal{P}_i$ we have a pair (a_i, b_i) representing the starting position within the SSSM. These positions correspond to the bordering frames between motion primitives. We only check if any candidates are closer than 5 frames and if so, we consider only the one with a corresponding warping path with smaller cost.

Complexity analysis: The critical computation step for detecting primitives is building the graph representation from the sets of nearest neighbors. Creating this graph requires checking all possible connections of each neighborhood entry in $\mathcal{N}$ to other entries by a number of s possible steps. For each of the neighborhoods of each activity there is a maximum of k entries. Since the number of edges is limited by $O(k \cdot s \cdot n)$, the search for connected components is limited to the same complexity, with an overall run time of $O(k \cdot s \cdot n)$.

V. Symmetry of Motion Data

Motion data often contains intrinsic symmetries which can be exploited during analysis. We focus on mirrored motions and begin by defining the plane of symmetry. Let $X = \{x_1, \ldots, x_J\} \in \mathbb{R}^{3\times J}$ be a geometric model of a moving subject, i.e. an ordered set of joints. A motion X_M is a multi-dimensional trajectory $\mathbb{X}$ of X over time. Let P_X be a plane spanned by two perpendicular vectors which connect joints or linear combinations of joints.

For human models, the plane of symmetry is the sagittal plane, i.e. the vertical plane dividing the body into left and right. A motion is symmetric with respect to this mirror plane if, for a set of descriptive features $\mathcal{F}$, at least one pair of features $(f_i, f_j) \in \mathcal{F}$ can be switched without imposing a (significant) change on $\mathbb{X}$. The concept of the mirror plane can be transferred from humans to other models; all vertebrates are bilaterally symmetrical with two pairs of appendages such as limbs, fins or wings and the sagittal plane is also a mirror plane.

The symmetry of an action segment X_A, based on its mirrored version X'_A mirrored at the sagittal plane, may be characterized as follows:

1) X_A is highly symmetric if its primitive segmentation is exactly the same as that of X'_A.
2) X_A is exclusively phase-shift symmetric if its primitive segmentation has no cuts in common with X'_A.
3) X_A is asymmetric if the primitive segmentation of X'_A returns no cuts at all.

Naturally, a mixture of two or more situations is possible when an action contains different types of motion primitives. Therefore, it makes sense to treat each action found by the action segmentation separately. We make use of phase-shifted symmetry in order to distinguish phase-shifted primitives like single steps in walking.

VI. Clustering of Motion Primitives

Clustering the motion primitives allows us to find the frequency with which primitives occur and have an indication of the semantic and temporal relationships between different primitives. This is of great interest for both motion synthesis and motion analysis. Note that for our unique clustering algorithm the number of clusters does not need to be specified.

A cluster graph $\mathbf{G}_\mathbb{M}$ is used to store the similarity information between the motion primitives. In this graph, each primitive is represented as a node. Consider a sparse self similarity matrix associated with a motion M; the motion primitives m_q are represented by squares on the main diagonal. The goal is to search for valid warping paths between each pair of motion primitives m_i and m_j. To this end, we can build a neighborhood graph (see Section IV-C) including the neighbors in the rectangle region that is spanned when comparing m_i and m_j. We then try to find the shortest path in this submatrix from entries at the top to the bottom. If this shortest path satisfies a minimum length requirement and has a slope inside the range of valid slopes (see Section IV-C) we add an edge between the corresponding nodes in $\mathbf{G}_\mathbb{M}$. After the algorithm has gathered all similarity information, a search for the strongly connected components is performed on $\mathbf{G}_\mathbb{M}$. Each strongly connected component represents a cluster of motion primitives.

The algorithm presented above is a modified version of the algorithm of Vögele *et al.* [16]. Both approaches perform equally well for clustering, though the current approach is more efficient, since only small neighborhood graphs between each pair of

Authorized licensed use limited to: FH Koeln. Downloaded on November 22,2022 at 10:34:35 UTC from IEEE Xplore. Restrictions apply.

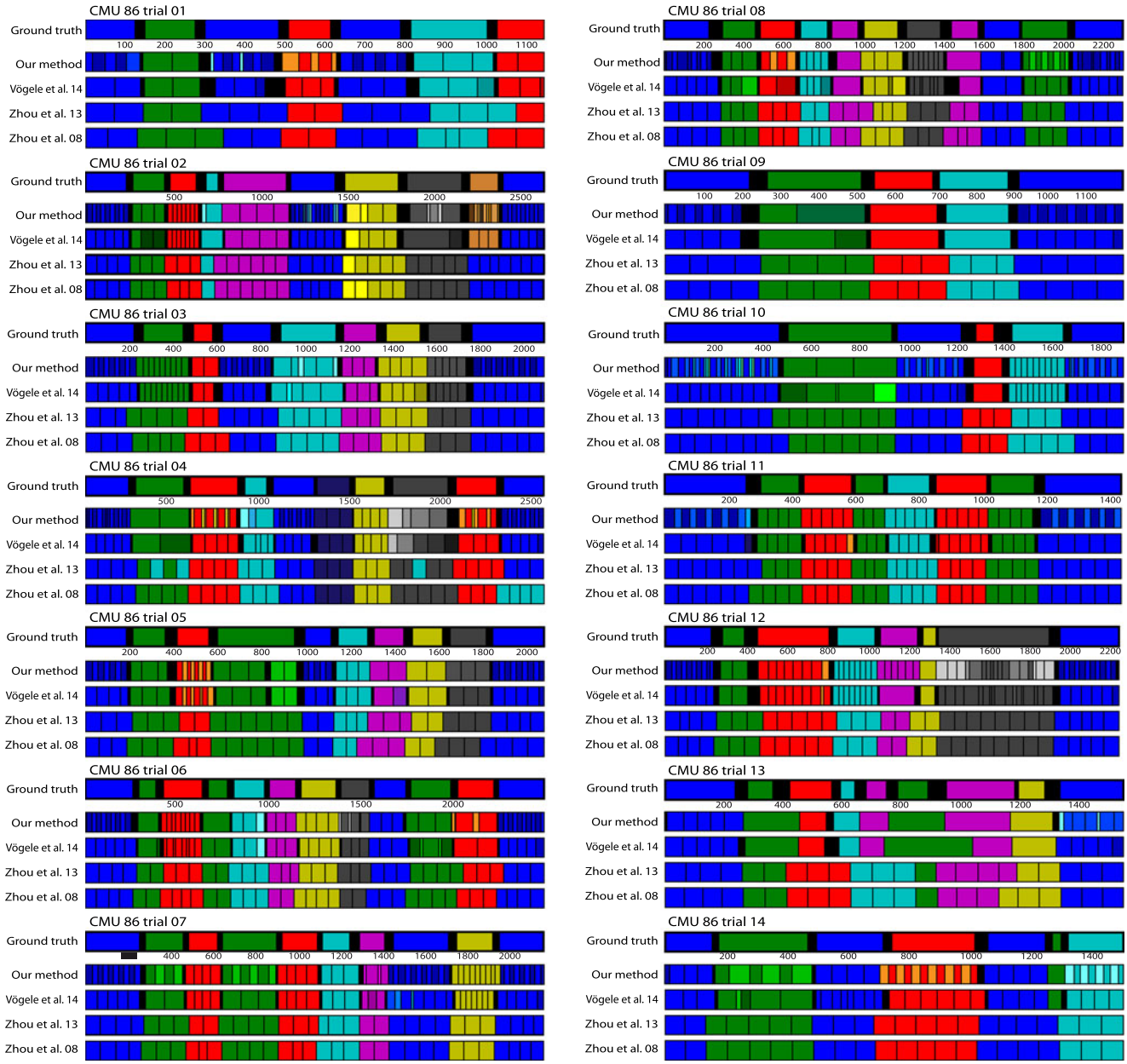


Fig. 6. Segmentation results for CMU 86 trial 01 to 14. For each trial, the first row displays the human ground truth annotations, the second row displays our results, and the third to fifth rows display competing methods [9], [10], [16]. Note the variation in length of actual motion primitives.

primitives are constructed, while in the previous work one large graph was constructed over all the neighbors of the trial.

VII. Experiments

We report on a series of experiments to show the effectiveness of our approach. First, we compare our results with previous methods on a set of motion capture data. Second, we show that meaningful results are obtained when using different sensor modalities such as accelerometers and EMG sensors. Finally, we apply our approach to Kinect skeleton data and show that our motion primitives are meaningful and consistent with of human-annotated key frames.

A. Segmenting Markered Motion Capture Data

We apply our segmentation method to the motion sequences of subject 86 of the CMU database [54], as was done by Zhou *et al.* [9], [10]. We compare our segmentations to theirs and our previous work [16] in Fig. 6. We show a number of improvements in comparison to [9], [10], the most notable being the ability to segment fine-structured motion primitives in nearly all cases. We are also able to distinguish different styles of executing a task. For example, in the case of wiping a window/black board (CMU, trial 12 of subject 86), Zhou *et al.* [10] group all the primitives together as the same type (dark grey blocks), while we are able to distinguish between back and forth wiping motions versus circular wiping movements (various shades of grey).

Authorized licensed use limited to: FH Koeln. Downloaded on November 22,2022 at 10:34:35 UTC from IEEE Xplore. Restrictions apply.

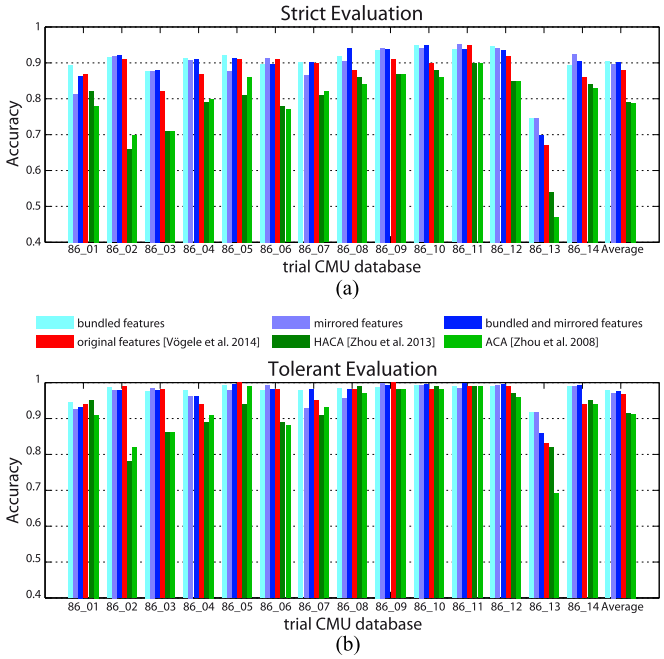


Fig. 7. Accuracy values of different segmentation methods. Light blue bars refer to our method (bundled features only), purple is our method (mirrored features only), blue is bundled and mirrored features combined, red is Vögele *et al.* [16], dark green is Zhou *et al.* (ACA) [9], and light green is Zhou *et al.* (HACA) [10] using (a) the strict evaluation, counting strictly the classes all methods detect and (b) the more tolerant evaluation which allows transitions as valid classes. For both evaluations the bundled and mirrored features give higher accuracy values in average compared to the original features and the (H)ACA based approaches. The bundled features especially have a better effect on the accuracy compared to the mirroring.

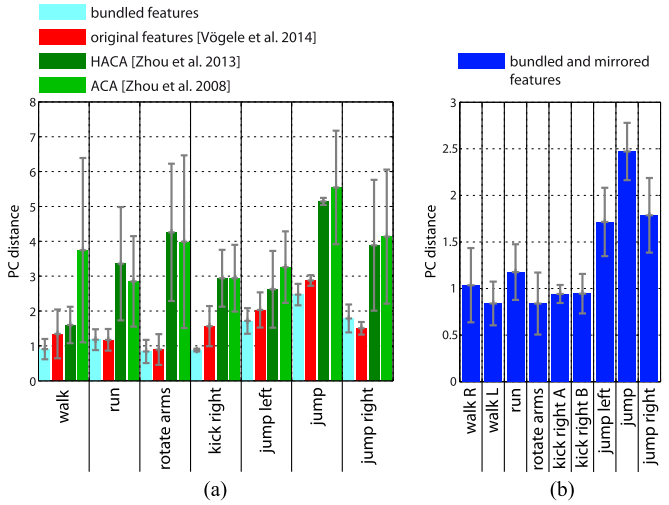


Fig. 8. (a) Evaluation of clusters produced by our method, the method of Vögele *et al.* [16], the HACA method, and the ACA method. The example at hand is trial 3 of CMU subject 86. The respective means are given by the blue color bars for our method, red for Vögele *et al.*, and (dark) light green color bars for (H)ACA, with variance shown as error bars. Note that lower distances reflect more consistent clusters. (b) Evaluation of clustered results produced by our method. Here we plot the mean D, with variances shown as error bars as in (a), but for the case where both bundling and symmetry features are included. This is one case where additional motion classes are introduced by exploiting symmetry of motion: there are two classes of steps and also two classes of "kicks." The distinctions are caused by different types of symmetry: walking is phase-shifted and kicking included one part which was symmetrical (more static) and one which was asynchronous.

Secondly, we are able to distinguish symmetric movements and separate primitives accordingly. Examples include rotation of the body in Trial 7 (variations of blue and green blocks), dribbling the ball in trial 14 (dark green, light green, red and orange blocks). Other approaches cannot distinguish between left versus right steps of the foot, nor ball handling with the left versus right hand.

Accuracy comparison: Our method produces the same action classes as [16] and nearly the same as [9], [10]. We use the same methods as [16] to evaluate the segmentation accuracy on a frame level, using both a strict and a tolerant evaluation, and present the results in Fig. 7. For a motion primitive s, the strict method checks whether all of s's frames belong to the same action class as the other primitives found from the same segment; the tolerant method eases the constraint to both the same action class as well as transition/uncertainty segments. Due to the finer division of primitives found by our method, there are different clusters representing the same motion. For example, walking consist of alternating left and right steps. For consistent evaluation with previous methods, we have assigned such symmetrical counterparts to the same class, i.e. the 'left step' cluster and the 'right step' cluster are both assigned to the walk action. We achieve significantly higher accuracy values for both types of evaluation, with an average of 90% for our method, 88% for the method of Vögele *et al.*, and 79% for (H)ACA for the strict evaluation and 99% in comparison to 97% for Vögele *et al.*, 92% for HACA and 91% for ACA for the tolerant evaluation. Applying segmentation to the label transfer problem is discussed in Vögele *et al.* [16] and based on our current values, we anticipate that applying feature bundling would yield similar results.

Intra-cluster variance for motion primitives: Dynamic time warping is an established distance measure for temporal sequences and accumulates the local (frame-wise) distances from one segment to the warped version of the other. For a given cluster C, a cumulative distance measure D, tallied over all pairs of segments s_i and s_j contained in the cluster can be defined as

$$D = \sum_{i=1, j\neq i}^{\|C\|} \left(\frac{\mathrm{DTW}_\alpha(s_i, s_j)}{\|s_i\|} \right) \tag{5}$$

where $\|s_i\|$ is the length of segment s_i and DTW_α is the DTW distance of point clouds as defined by Kovar *et al.* [20]. Note that D is particularly sensitive to outliers and will detect scattered or inconsistent clusters.

By making finer distinctions between motion primitives, we achieve lower intra-cluster variances for the clusters. Fig. 8(a) compares the clustered results by our method, the method of Vögele *et al.* [16], the HACA [10] and the ACA [9] method. Fig. 8(b) shows the same low variances when mirrored features are included. The given example is one case where additional motion classes are introduced by exploiting symmetry: there are two classes of 'steps' and also two classes of 'kicks'. The distinctions are caused by different types of symmetry. While

Authorized licensed use limited to: FH Koeln. Downloaded on November 22,2022 at 10:34:35 UTC from IEEE Xplore. Restrictions apply.

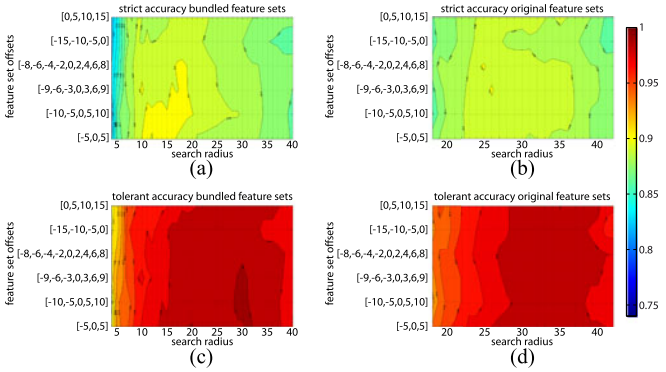


Fig. 9. Maps displaying the accuracy for various combinations of the parameters. The generalized radius R is plotted on the horizontal axis, choices of stacking offsets are plotted along the vertical axis. A stacking offset of $[-5, 0, 5]$ is a concatenation of frames at times $i-5$, i and $i+5$. (a) and (b) show the maps based on the bundled features, where (a) is the stricter and (b) is the more tolerant version. (c) and (d) show the same based on the bundled features, where (c) is the stricter and (d) is the more tolerant version.

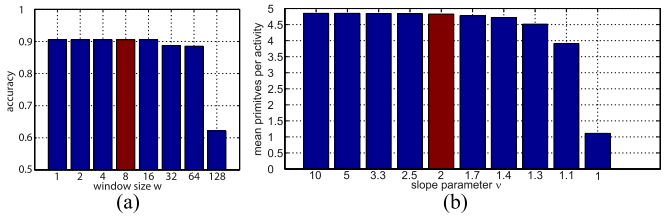


Fig. 10. (a) Accuracy (strict evaluation) for varying window sizes w of the region growing step. $w = 8$ (red bar) was chosen for our experiments. (b) Average number of motion primitives per activity based on the minimal and maximal slope of the warping paths. $\nu = 2$ (red bar) was chosen for all other experiments in this work.

walking is phase-shifted, kicking includes one symmetric part (more static) and one phase-shifted part (where the leg was up).

Our clusters group together a variety of motion primitives without transitions and primitives from other actions. In particular, our primitives reflect exactly the number of repetitions within actions. For illustration, there are five repetitions of 'rotate arms' in one sequence (see Fig. 6, Subject 86 trial 03, frames 1600-1800) and we segment this into exactly five primitives. Zhou *et al.*'s methods [9], [10] do not account for the inherent repetition and segment the action into three primitives, thereby yielding much higher DTW distances between these primitives.

Over all clusters of all trials from actor 86 we obtain an average cluster variance of 1.18 (min: 0.52, max: 2.66, std: 0.42) for mirrored and bundled features, 1.17 (min: 0.58, max: 2.58, std: 0.37) for bundled features and 1.69 (min: 0.69, max: 3.45, std: 0.54) for the original features. Compared to HACA 2.51 (min: 0.48, max: 8.78, std: 1.53) and ACA: 2.56 (min: 0.86, max: 9.42, std: 1.50).

Parameter evaluation: The most important parameters for our segmentation method are the search radius (Section IV-B) and the temporal window for feature stacking (Section IV-A). We found that our method is insensitive to either parameter and show the segmentation accuracy in Fig. 9 for various parameter settings for the strict and tolerant evaluations for both original and bundled features. All plots show that the accuracies are high for nearly all possible combinations of parameters.

Our region growing in the activity separation step stops if no new neighbors were found in a window of w frames. We tested our approach with various window sizes and computed the strict accuracy measure for evaluation. Fig. 10(a) shows that our method is very robust with respect to the window size, with accuracy dropping only when the window gets very large (128 frames).

For segmenting the motion primitives, there are the additional parameters of the allowed min. and max. value of the warping path slope ν (Section IV-C). Following the conventions of [18], [41], we set the slope to be within the window $\frac{1}{\nu}$ and ν. We evaluate ν by computing the average number of primitives found per activity on the dataset, with results shown in Fig. 10(b). If ν is smaller than 2, the number of primitives drastically decreases, while larger values for ν do not increase the number of primitives. For all our experiments, we set $\nu = 2$ to permit extreme temporal deformation between motion segments. Our experiments indicate that such warps did not occur in the CMU dataset.

Timings: A timing breakdown shows that action segmentation (including feature bundling and knn search) is, in practice, approximately linear in the number of frames. Theoretically, the worst case complexity for region growing is $O(kn)$, i.e. when the first region grows from the first to the last frame of the input motion sequence. This case was not observed in practice. The shortest path searches for primitive detection is quadratic with respect to the number of frames per activity. Finally, the clustering step also computes in linear time (in the number of motion primitives).

On an Intel Core i7 4930 K at 3.40 G Hz we were able to segment and cluster each motion sequence in the CMU examples (up to 3000 frames in length) in less than 15 seconds using our single threaded Matlab implementation. In comparison, the HACA method of Zhou *et al.* [10] has at least cubic complexity. In the experiments with short trials described in this section, both methods produce approximately the same timings.

B. Combined Motion Sensors

To demonstrate our algorithm's effectiveness on different sensor modalities, we recorded a set of electromyography (EMG) and acceleration motion data using a Delsys Trigno wireless acquisition system. EMG recordings show the electrical activity produced by skeletal muscles and are commonly analyzed in biomechanics and neurology. Acceleration recordings show the local accelerations from change in sensor velocity and are commonly analyzed in biomechanics and sports science. Nearly all 'wearables' use accelerometer readings to analyze user activity. Typically, analysis of EMG and accelerometer signals is done on sequences already segmented into motion cycles; the segmentation is almost always done manually, and frequently relies on other readings such as motion capture or video data. Our automatic segmentation technique can significantly improve the work flow in domains using EMG and accelerometer recordings.

Authorized licensed use limited to: FH Koeln. Downloaded on November 22,2022 at 10:34:35 UTC from IEEE Xplore. Restrictions apply.

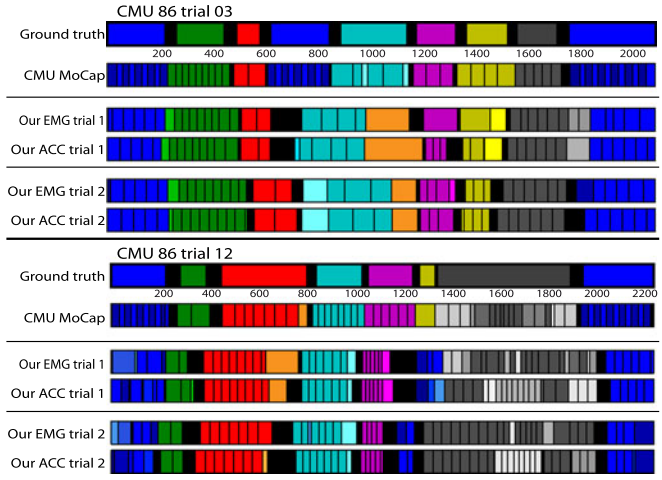


Fig. 11. Exemplary results of two trials. As can be seen from the key point visualization, this third example is also an interesting situation for the consistency measures of subject 86 of the CMU. The color codes correspond to the different motion clusters the primitives were assigned to by our method. The results are very similar between the data sets, even between the original and the re-enacted recordings.

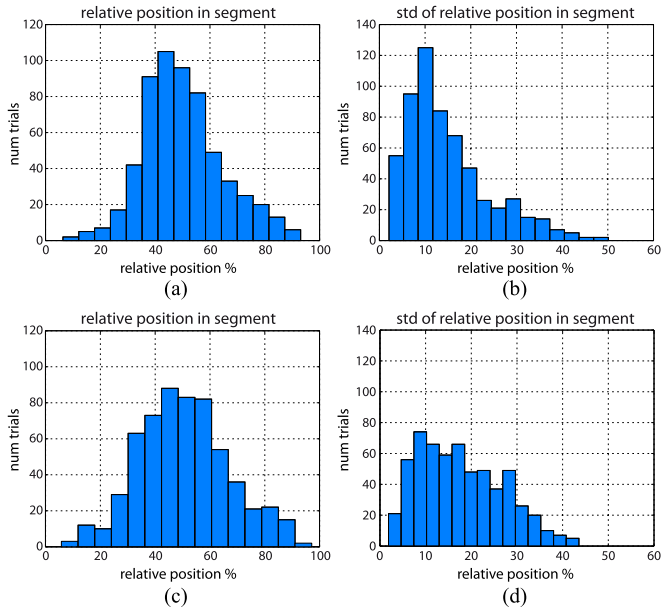


Fig. 12. Histograms show the average location of given key point annotations (left-hand side) in the motion primitive and their standard deviations (right-hand side). In the majority of cases, the location of key points is approximately at the center of the corresponding primitive with a deviation of less than 20% if feature bunding is used, as shown in (a) and (b). If the original features are given as input the average locations are spread with larger standard deviations, as shown in (c) and (d).

We take recordings from one trial subject who was asked to re-enact the sequences of subject 86 in the CMU database. Results of two repetitions of trials 3 and 12 are compared to the CMU originals (see Fig. 11). The raw EMG readings consist of data streams of 16 sensors, each documenting the activation of large skeletal muscle groups (refer to Appendix C for documentation).

All EMG recordings were pre-processed in a standard fashion: the signals were rectified, re-sampled from 2000 Hz to 30 Hz and smoothed by a 20 Hz low pass filter (see [55] for a review on EMG data processing). Acceleration recordings were downsampled from 120 Hz to 30 Hz and filtered by a binomial filter over a window of 16 frames.

The resulting segments and primitives (see Fig. 11) show that our approach works on EMG and acceleration data as successfully as clean motion capture, despite the former two being much noisier inputs. The EMG segments are very similar to the acceleration segments representing the same motion; in most actions, the same number of primitives were found across the two modalities. We hypothesize that the slight differences in timing are due to inherent differences in the signal captured by the two modalities.

Clustering of the motion primitives is comparable between the two modalities for most activities. The largest differences are in 'wiping a window' (grey blocks in trial 12). Here we were unable to distinguish between back and forth wiping versus circular wiping in the EMG, while the accelerations gave clear motion primitive clusters. The EMG data does not reflect this difference since the muscle activation on the main arm extensors and flexors do not change as clearly as the accelerometer readings.

C. Kinect Action Data

Processing noisier markerless motion capture systems can be a challenge, but our method can reliably handle such data when the feature bundling step is included. We segment the MSRC-12 Kinect Gesture Data Set from Fothergill *et al.* [56], consisting of a number of action sequences which were originally recorded for action recognition. The data set consists of 594 sequences in total from 17 actors. The trials range from 1000-2000 frames recorded at 30 frames per second. The data are available as 35 joint angles of 3 scalars over a length of n frames. Examples of different segmentation results can be found in Fig. 13.

Impact of feature bundling: We show the original input data streams and our segmentation results with and without feature bundling. The given key points [57] are annotations of the gestures at a specified key frame and are indicated by red lines in all subplots. Feature bundling allows us to segment a regular pattern of primitives that coincide with the time series [see Fig. 13(a) and 13(b)]. Fig. 13(b) shows more variation in the lengths of the primitives and originates from a break in the input motion (first half of the trial, dark blue bar) and speed variations (later half of the trial) which can clearly be seen in the plot of the data stream. Fig. 13(c) is particularly interesting because the primitive have a much finer structure. The motion primitives occur as ones recurrent groups of smaller parts (light green/green) alternating with a longer primitive (blue) and are well aligned with the key points.

Without feature bundling, the segmented motion primitives are far less regular. In Fig. 13(a), three repetitions were broken up into two individual parts (yellow and turquoise). In Fig. 13(b), the longer breaks (brown) between the repetitions were found but not all repetitions (dark blue) could be cleanly separated. Finally, in Fig. 13(c), a similar pattern was found both with

Authorized licensed use limited to: FH Koeln. Downloaded on November 22,2022 at 10:34:35 UTC from IEEE Xplore. Restrictions apply.

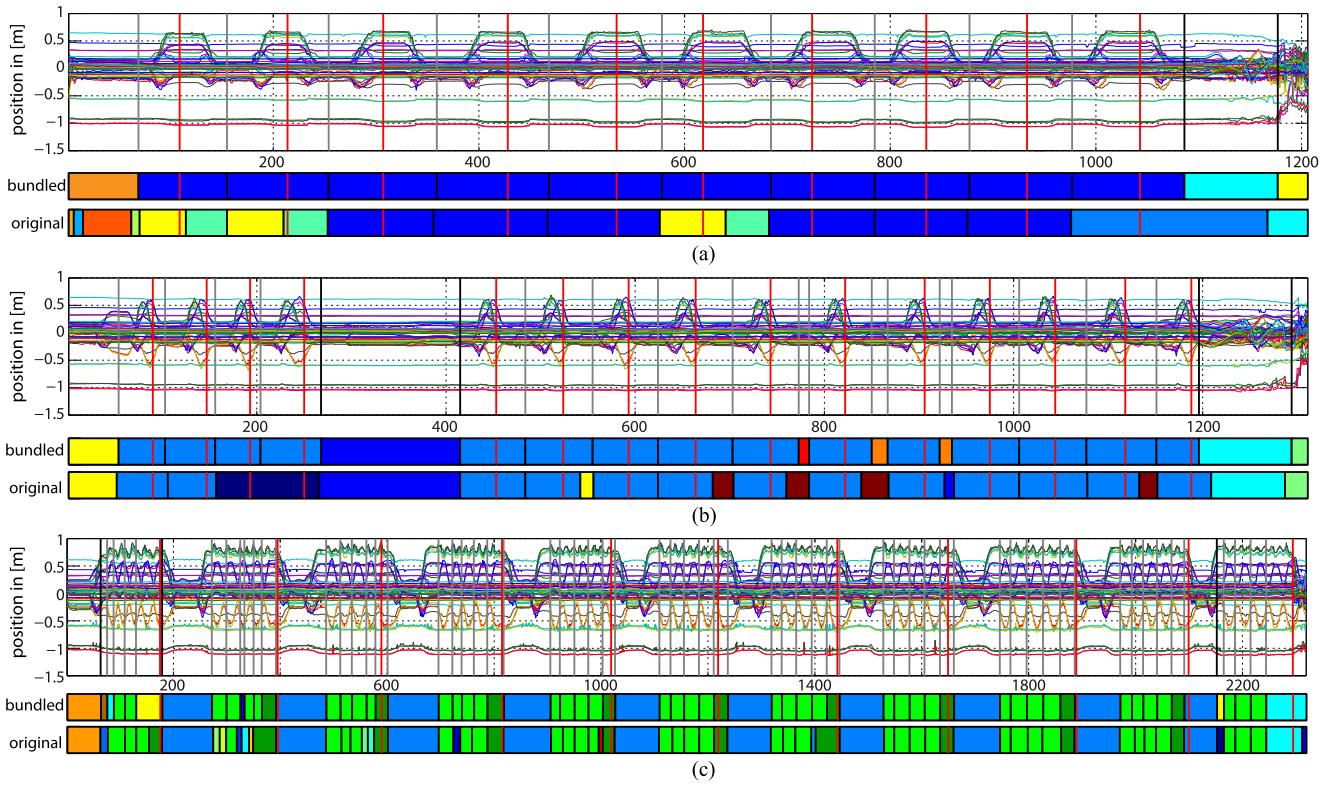


Fig. 13. Three different results produced on Kinect recordings. The input time series are plotted for an overview of the general structures of the underlying motions, showing motions with (a) regular primitives segments [P1_2_g09_s08], (b) a sequence with a longer break and also some speed variation [P1_2_g05_s08], and (c) many quick repetitions of primitives, i. e., waving both hands (see green bars) [P3_2_g11_s29]. The very fine primitive segmentation is desirable for motion understanding, but may be punished by the consistency measure based on key point locations since the key point annotation now falls at the end or beginning of the fine primitives.

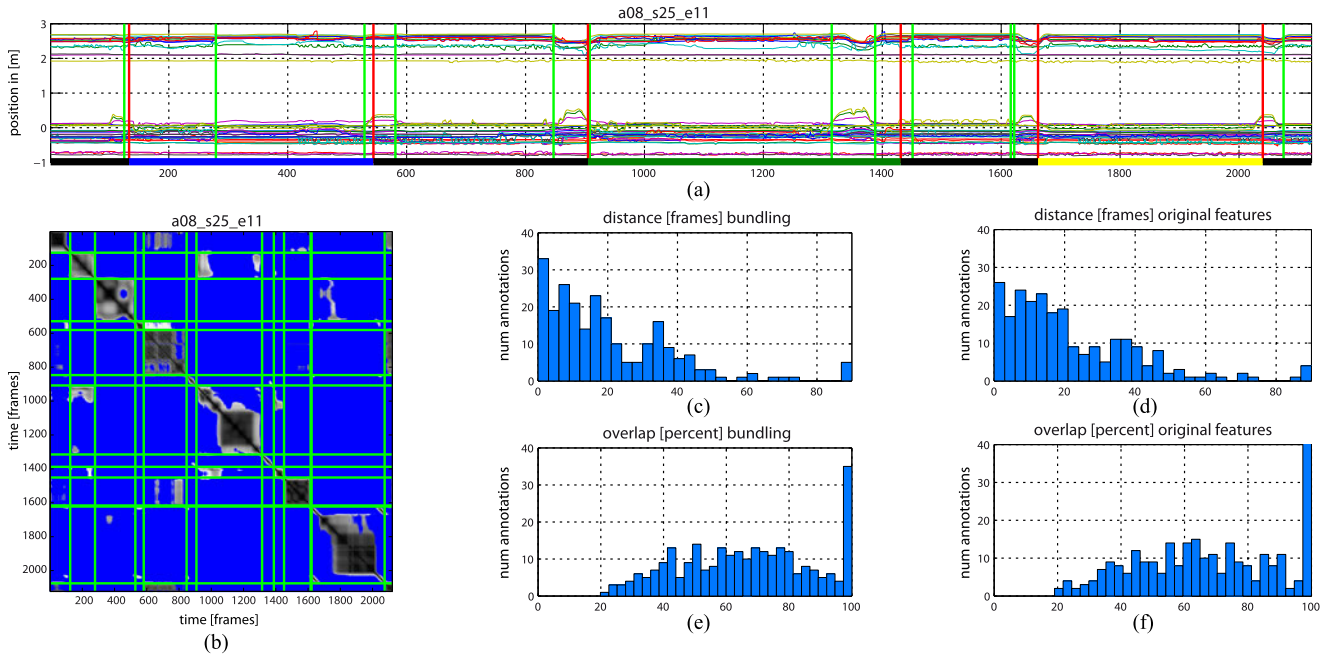


Fig. 14. Results produced on Kinect recordings from the MSR 3D Action Dataset. (a) Segmentation results (green lines) and ground truth annotations (red lines) for trial a08_s25_e11. The bars at the bottom indicate the ground truth labels: no action (black), drinking (blue), eating (green), and reading book (yellow). (b) The corresponding SSSM with segmentation results as green lines. (c) Histograms for the distance between annotations and computed cuts on bundled and (d) original features. (e) Histograms showing the overlap between annotated segments and primitives identified by our method on bundled and (f) original features.

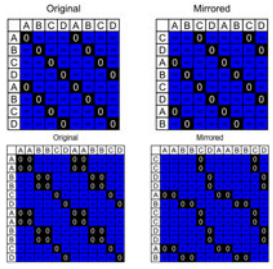
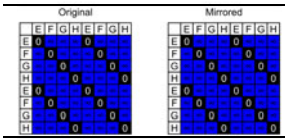
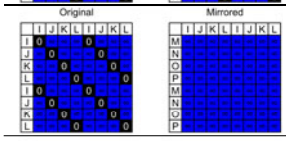
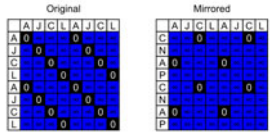
Authorized licensed use limited to: FH Koeln. Downloaded on November 22,2022 at 10:34:35 UTC from IEEE Xplore. Restrictions apply.

and without feature bundling, but without bundling, the shorter primitives are split into smaller irregular parts that are assigned to different clusters.

Consistency evaluation: To see how well our identified motion primitives correspond to the annotated key point, we measured the difference between the start frame of each motion primitives and the key point and scale this value proportionally to the length of the primitive. We plot the distribution and standard deviation in Fig. 12. The histograms show that with feature bundling, the key-point annotation as judged by the human annotator occurs consistently at a relative position of 40-60% in the segmented motion primitive [see Fig. 12(a)], with a relatively low standard deviation of 15% around the mean [see Fig. 12(b)]. Without feature bundling [see Fig. 12(c) and (d)] the mean values are more spread with higher standard deviations.

While our segmentations are highly consistent with the annotated key points, there are still a number of exceptions which contribute to the distribution of deviations. One example is shown in Fig. 13(c) where a sequence of smaller motion primitives occurs arranged in the same order. According to our goal, to identify repetitive motion primitives, the fine segmentation is the desired result. However, this is non-ideal for the consistency measure, since it considers the relative position within the motion primitive. With very fine primitives, the key points are at the end of the last small cluster (green) or at the beginning of the subsequent larger cluster (blue). Grouping together smaller primitives and re-clustering could alleviate this problem.

TABLE I
SCHEMATIC SELF SIMILARITY MATRICES FOR VARIOUS TYPES OF SYMMETRIES

	Two cases of phase-shifted symmetries. The first matrix shows that the original and mirrored features are disjoint, i.e. the diagonal structures in the SSSM are at different locations. The second is a similar situation where the original motion shows some speed variation, leading to a more diverse structure in both matrices. The mirrored features match the originals in some sense but the variation is seen in the slopes of the diagonals.
	Example of a symmetric motion: The diagonal structures in the SSSM are at exactly the same locations, leading to no additional cuts.
	Example of an asynchronous motion: The diagonal structures in the original SSSM have no matches in the mirrored version, again leading to no additional cuts.
	Example of a motion with different symmetries, with local matches for the original diagonals in the mirrored version. This means that the motion has phase-shifted symmetrical phases interrupted by other phases which are asynchronous.

D. Noncyclic Kinect Data

We also test on the more challenging MSR 3D Online Action Dataset [58]. This data set contains noisy, mostly non-cyclic actions such as drinking, eating, using laptop, reading cellphone, etc. recorded with a Kinect. We apply our method on the subset S4, which contains 36 sequences of long trials (1000-3500 frames recorded at 30 fps) from 11 subjects and is intended for continuous action recognition. The data are available as 20 3D joint positions. An example segmentation result can be found in Fig. 14(a) along with the corresponding SSSM b). Since the annotations are given as intervals in this data set, we evaluate according to the absolute distance in frames from the start and end points of the annotation with respect to the next found cut. This measure alone would favor over-segmentation; thus we compensate with an additional measure of the overlap of an annotation with the largest segment found by our method. Using bundled features, we obtain a mean distance, between the start and end points of the annotations and our primitives, of 21.2 frames with a standard deviation of 19.6 frames. The overlap is 66.56% (mean) and 21.16% (std). Without bundling, we obtain 22.5 (mean) and 19.7 (std) frames and 67.2% (mean) and 22.3% (std) overlap. Fig. 14(c)–14(f) shows the histograms for these values. These sequences contain relatively static poses and as such, the effect of feature bundling is not as strong as in the more dynamic sequences where there are more variations in pose. We were able to determine that many actions in this data set can be split into three segments: a dynamic part in the beginning (e.g. raising cup to mouth), a static part in the middle (drinking) and another dynamic part in the end (lowering cup). This corresponds to the rough annotations in this dataset where complex actions are annotated as a block, while our approach returns finer motion segments.

E. Applications

Our segmentation method facilitates solutions to many common problems in motion analysis and motion synthesis. For instance, the semantic annotations given for primitives or clusters of primitives could be used to transfer annotation labels to unlabeled data. Consider a database $S_{\mathrm{u}} = [\mathcal{U}_1, \ldots, \mathcal{U}_n]$ of n unlabelled motion primitives obtained by our proposed segmentation method and a set $S_{\mathrm{l}} = [\mathcal{L}_1, \ldots, \mathcal{L}_j]$ of j labelled motion primitives. For all labelled primitives $\mathcal{L}_i, i \in [1, \ldots, j]$ a knowledge base can be built by computing a feature-space representation of S_{l}. A nearest neighbor search for all unknown motion primitives in S_{u} can be performed then creating a similarity graph from this information. Based on this, optimal warping paths between any of the $\mathcal{L}$s and $\mathcal{U}$s can be found by application of a subsequence DTW. If there is a valid correspondence between the query and a labelled primitive, then the label associated with the primitive with the lowest warping path cost can be transferred. We achieve very high precision with this approach (1.0 for motion classes jump, kick, punch, run, squat, and stretch and 0.98 for walk) when using trials 86_01, 86_02, and 86_03 as knowledge base and transfer labels to all motions of subject 86 of the CMU motion capture database.

Authorized licensed use limited to: FH Koeln. Downloaded on November 22,2022 at 10:34:35 UTC from IEEE Xplore. Restrictions apply.

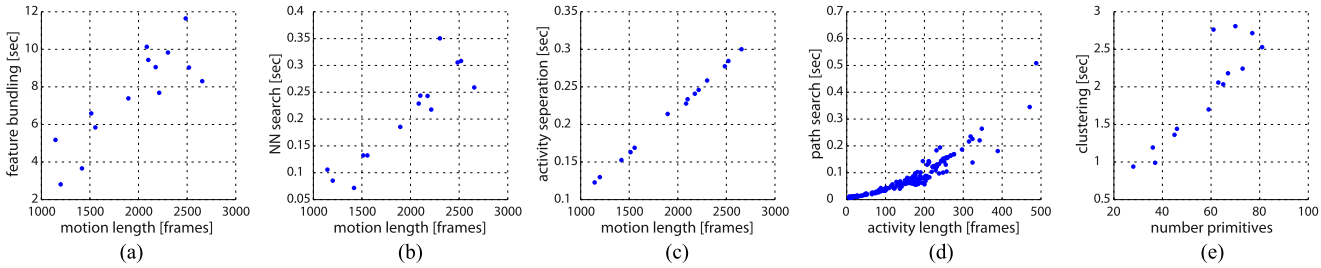


Fig. 15. Scatter plots for timings of the steps of our method: (a) feature bundling, (b) knn search, (c) region growing (scan lines), (d) path searches, and (e) clustering. Note that the complexity of the first three steps (a)–(c) is linear in the number of frames, whereas path searches (d) are quadratic in the lengths of the activities. In practice, the clustering (e) also computes linear in the number of motion primitives.

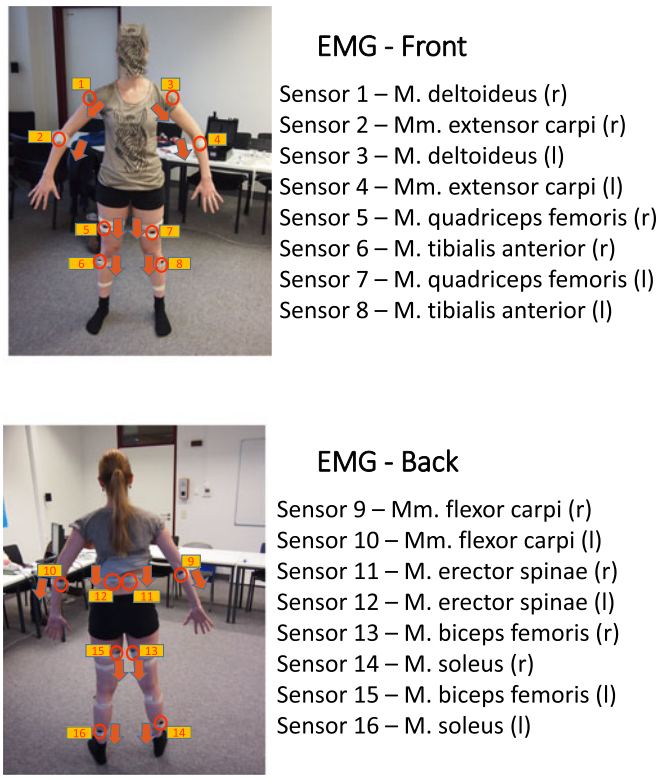


Fig. 16. Documentation of EMG sensor placement on a human subject seen from the front and back, respectively. Orange arrows point to direction of y-axis.

As another application, precomputed motion primitives can also be used for motion synthesis with classical motion graph techniques or as input for statistical models as used in [19], [59].

VIII. Conclusion

We have presented a segmentation method that can separate cyclic activities and their transitions for a number of data modalities. Our approach tackles the segmentation problem on a general level in terms of the choice of crucial parameters, e.g. the search radius and the feature offsets for stacking. Our feature bundling is a novel contribution and proves to be especially helpful for processing noisy data modalities such as EMG, accelerometer and Kinect motion capture. We have used a five-point derivation to estimate the direction of movement in the bundling, but when faced with severe noise, one will need more robust methods. This will further reduce variance in the feature space, with few implications, as long as one does not try to synthesize new sequences from the feature space.

So far, we have only shown our segmentation method on sequences taken in constrained settings. We anticipate that it is also applicable to sequences taken "in the wild", given the right features and similarity measures. This remains an open topic for future work. Challenges also remain to be seen once the segmentation is even further generalized, for example to spatial segmentation problems such as mesh animation sequences [60], [61].

Since our method is based on self-similarities, the limits of finding primitives are reached when input sequences contain only non-repetitive activities (e.g. one step, jump, turn). However, the assumption that most human activities are of repetitive nature is valid for motion capture data within existing datasets such as CMU [54]. If the motion is cyclic, so are the measured local accelerations. For EMG, however, we could not identify substantial changes between the individual repetitions in the muscle activation patterns. Here changes may occur from fatigue effects in longer motion trials, though this was not observed in our experimentation.

Currently, all tested data modalities have been processed individually; a natural next step is to jointly process several data types in a multi-modal setting. Secondly, exploring smooth embedding approaches [62] for feature bundling is a promising direction for future work. Even though a PCA works for our examples, small artifacts are nonetheless visible, which may not be the case in other non-linear (but more computationally expensive) embeddings.

We offer source code for our feature bundling, segmentation and clustering approach online.[1] We believe that it will be of use for many different research applications and hope that it will encourage others in the community to work on generalized segmentation.

Appendix A
Symmetry Types

Schematic representations of self-similarity matrices for each of the possible cases, discussed in Section V are given in Table I.

[1][Online]. Available: http://cg.cs.uni-bonn.de/en/projects/gemmquad/motionsegmentation/

Authorized licensed use limited to: FH Koeln. Downloaded on November 22,2022 at 10:34:35 UTC from IEEE Xplore. Restrictions apply.

For symmetrical motions, there is no difference in the diagonal structure in the SSSM from the original and mirrored features. This leads to no additional cuts when symmetry is exploited for the segmentation. Phase-shifted symmetry can occur with and without speed variation. Without any speed variations, a more diverse structure in the self-similarity matrices occurs in both the original and mirrored features. For the original features, more block-shaped parts may appear on each diagonal, while in the mirrored setting, all diagonals aggregate to a more curvy pattern. In the asynchronous case, the mirrored self-similarity matrix shows no diagonal structures at all, adding to no additional cuts. When symmetries are mixed, the SSSMs are characterized locally according to the corresponding symmetry type.

Appendix B
Details on Timings

Fig. 15 shows a timing breakdown for the CMU examples. The first segmentation step [part (c)] including feature bundling [part (a)] and knn search [part (b)] activity detection are, in practice, approximately linear in the number of frames [Fig. 15(a)–15(c)]. In theory, the worst case complexity for region growing is $O(kn)$, i.e. when the first region grows from the first to the last frame of the input motion sequence. This case was not observed in practice. The shortest path searches for primitive detection [part (d)] is quadratic with respect to the number of frames per activity. Finally, the clustering step [part (e)] also computes in linear time (in the number of motion primitives).

On an Intel Core i7 4930 K at 3.40 GHz we were able to segment and cluster each motion sequence (up to 3000 frames in length) in less than 15 seconds using our single threaded Matlab implementation.

Appendix C
Data Source Documentation

We recorded our EMG and acceleration data using a combined wireless Delsys Trigno system. The sensor setup consisted of 16 sensors placed on larger muscle groups of the trial subject (see Fig. 16 for a list of locations and a visualization of the hardware attachment). Both sensor attachment and recordings were supervised by an accredited expert.

A selection of trials performed originally by subject 86 of the CMU database was re-enacted by another human subject as true to original as possible. This was done to compare the performance of our segmentation method on other data modalities while maintaining control of the activities represented by the data streams. There are some minor deviations from the original scripts due to different geometry of the location (e.g. in one of our trials, originally 86_12, the subject had to walk some additional steps after sweeping the floor and before reaching the white board). The total length of the trials differs from the original slightly due to similar reasons. Nevertheless, the sequences contain the same activity classes as the original sequences, and have mostly the same number of repetitions and are therefore suitable for a qualitative comparison.

Acknowledgment

The authors would like to thank K. Welle (Bonn University Hospital, Clinic for Orthopaedics and Trauma Surgery) for supporting with the EMG measurements.

References

[1] C. Li, S. Q. Zheng, and B. Prabhakaran, "Segmentation and recognition of motion streams by similarity search," *ACM Trans. Multimedia Comput., Commun., Appl.*, vol. 3, no. 3, Aug. 2007, Art. no. 16.
[2] Y. M. Chen, I. V. Bajic, and P. Saeedi, "Moving region segmentation from compressed video using global motion estimation and Markov random fields," *IEEE Trans. Multimedia*, vol. 13, no. 3, pp. 421–431, Jun. 2011.
[3] H. Kadu and C. C. J. Kuo, "Automatic human MoCap data classification," *IEEE Trans. Multimedia*, vol. 16, no. 8, pp. 2191–2202, Dec. 2014.
[4] D. S. Alexiadis and P. Daras, "Quaternionic signal processing techniques for automatic evaluation of dance performances from MoCap data," *IEEE Trans. Multimedia*, vol. 16, no. 5, pp. 1391–1406, Aug. 2014.
[5] Y. Rui and P. Anandan, "Segmenting visual actions based on spatio-temporal motion patterns," in *Proc. IEEE Conf., Comput. Vis. Pattern Recog.,* Jun. 2000, vol. 1, pp. 111–118.
[6] L. Zelnik-Manor and M. Irani, "Statistical analysis of dynamic actions," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 9, pp. 1530–1535, Sep. 2006.
[7] D. D. Vecchio, R. M. Murray, and P. Perona, "Decomposition of human motion into dynamics-based primitives with application to drawing tasks," *Automatica*, vol. 39, no. 12, pp. 2085–2098, 2003.
[8] C. Lu and N. J. Ferrier, "Repetitive motion analysis: Segmentation and event classification," *IEEE Trans. Pattern Anal. Mach. Intell.,* vol. 26, no. 2, pp. 258–263, Feb. 2004.
[9] F. Zhou, F. D. la Torre, and J. K. Hodgins, "Aligned cluster analysis for temporal segmentation of human motion," in *Proc. 8th IEEE Int. Conf. Autom. Face Gestures Recog.*, Sep. 2008, pp. 1–7.
[10] F. Zhou, F. D. la Torre, and J. K. Hodgins, "Hierarchical aligned cluster analysis for temporal clustering of human motion," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 3, pp. 582–596, Mar. 2013.
[11] M. Hoai, Z.-Z. Lan, and F. De la Torre, "Joint segmentation and classification of human actions in video," in *Proc. 2011 IEEE Conf. Comput. Vis. Pattern Recog.,* Jun. 2011, pp. 3265–3272.
[12] R. Araujo and M. Kamel, "Semi-supervised kernel-based temporal clustering," in *Proc. 2014 13th Int. Conf. Mach. Learn. Appl.*, Dec. 2014, pp. 123–128.
[13] J. Barbič *et al.*, "Segmenting motion capture data into distinct behaviors," in *Proc. Graph. Interface 2004*, May 2004, pp. 185–194.
[14] M. Müller, T. Röder, and M. Clausen, "Efficient content-based retrieval of motion capture data," *ACM Trans. Graph.*, vol. 24, no. 3, pp. 677–685, Jul. 2005.
[15] G. Liu and L. McMillan, "Segment-based human motion compression," in *Proc. ACM SIGGRAPH/Eurograph. Symp. Comput. Animation*, 2006, pp. 127–135.
[16] A. Vögele, B. Krüger, and R. Klein, "Efficient unsupervised temporal segmentation of human motion," in *Proc. ACM SIGGRAPH/ Eurograph. Symp. Comput. Animation*, 2014, no. 10, pp. 167–176.
[17] J. Hou, L.-P. Chau, Y. He, and N. Magnenat-Thalmann, "Human motion capture data tailored transform coding," *IEEE Trans. Vis. Comput. Graph.*, vol. 21, no. 7, pp. 848–859, Jul. 2015.
[18] B. Krüger, J. Tautges, A. Weber, and A. Zinke, "Fast local and global similarity searches in large motion capture databases," *Proc. ACM SIGGRAPH/Eurograph. Symp. Comput. Animation*, 2010, pp. 1–10.
[19] J. Min and J. Chai, "Motion graphs++: A compact generative model for semantic motion analysis and synthesis," *ACM Trans. Graph.*, vol. 31, no. 6, pp. 153:1–153:12, Nov. 2012.
[20] L. Kovar, M. Gleicher, and F. Pighin, "Motion graphs," *ACM Trans. Graph.*, vol. 21, no. 3, pp. 473–482, Jul. 2002.
[21] E. Keogh, S. Chu, D. Hart, and M. Pazzani, "An online algorithm for segmenting time series," in *Proc. IEEE Int. Conf. Data Mining*, Nov.–Dec. 2001, pp. 289–296.
[22] P. Fearnhead, "Exact and efficient Bayesian inference for multiple change-point problems," *Statist. Comput.*, vol. 16, no. 2, pp. 203–213, 2006.
[23] M. Ostendorf, V. Digalakis, and O. Kimball, "From HMM's to segment models: A unified view of stochastic modeling for speech recognition," *IEEE Trans. Speech Audio Process.*, vol. 4, no. 5, pp. 360–378, Sep. 1996.

Authorized licensed use limited to: FH Koeln. Downloaded on November 22,2022 at 10:34:35 UTC from IEEE Xplore. Restrictions apply.

[24] M. Müller, P. Grosche, and F. Wiering, "Robust segmentation and annotation of folk song recordings," in *Proc. 10th Int. Soc. Music Inf. Retrieval Conf.*, Oct. 2009, pp. 735–740.
[25] T. Prätzlich and M. Müller, "Frame-level audio segmentation for abridged musical works," in *Proc. 15th Int. Conf. Music Inf. Retrieval*, 2014, pp. 307–312.
[26] X. Xuan and K. Murphy, "Modeling changing dependency structure in multivariate time series," in *Proc. 24th Int. Conf. Mach. Learn.*, 2007, pp. 1055–1062.
[27] A. S. Willsky, E. B. Sudderth, M. I. Jordan, and E. B. Fox, "Sharing features among dynamical systems with beta processes," in *Proc. Adv. Neural Inf. Process. Syst. 22*, 2009, pp. 549–557.
[28] Z. Harchaoui, E. Moulines, and F. R. Bach, "Kernel change-point analysis," in *Proc. Adv. Neural Inf. Proc. Syst. 21*, 2009, pp. 609–616.
[29] F. Desobry, M. Davy, and C. Doncarli, "An online kernel change detection algorithm," *IEEE Trans. Signal Process.*, vol. 53, no. 8, pp. 2961–2974, Aug. 2005.
[30] M. Müller and P. Grosche, "Automated segmentation of folk song field recordings," in *Proc. ITG Conf. Speech Commun.*, 2012, pp. 1–4.
[31] P. Beaudoin, S. Coros, M. van de Panne, and P. Poulin, "Motion-motif graphs," in *Proc. ACM SIGGRAPH/Eurograph. Symp. Comput. Animation*, 2008, pp. 117–126.
[32] B. S. Chew, L. P. Chau, and K. H. Yap, "A fuzzy clustering algorithm for virtual character animation representation," *IEEE Trans. Multimedia*, vol. 13, no. 1, pp. 40–49, Feb. 2011.
[33] A. López-Mendez, J. Gall, J. R. Casas, and L. J. V. Gool, "Metric learning from poses for temporal clustering of human motion," in *Proc. Brit. Mach. Vis. Conf.*, 2012, pp. 1–12.
[34] J. Bernard *et al.*, "Motionexplorer: Exploratory search in human motion capture data based on hierarchical aggregation," *IEEE Trans. Vis. Comput. Graph.*, vol. 19, no. 12, pp. 2257–2266, Dec. 2013.
[35] S. Li, K. Li, and Y. Fu, "Temporal subspace clustering for human motion segmentation," in *Proc. IEEE Int. Conf. Comput. Vis.*, Dec. 2015, pp. 4453–4461.
[36] F. Lv and R. Nevatia, "Recognition and segmentation of 3-D human action using HMM and multi-class adaboost," in *Proc. 9th Eur. Conf. Comput. Vis.*, 2006, vol. 3954, pp. 359–372.
[37] M. Müller and T. Röder, "Motion templates for automatic classification and retrieval of motion capture data," *Proc. ACM SIGGRAPH/ Eurograph. Symp. Comput. Animation*, 2006, pp. 137–146.
[38] M. Müller, A. Baak, and H.-P. Seidel, "Efficient and robust annotation of motion capture data," *Proc. ACM SIGGRAPH/Eurograph. Symp. Comput. Animation*, 2009, pp. 17–26.
[39] R. Lan and H. Sun, "Automated human motion segmentation via motion regularities," *Vis. Comput.*, vol. 31, no. 1, pp. 35–53, 2015.
[40] S. Salamah, L. Zhang, and G. Brunnett, "Hierarchical method for segmentation by classification of motion capture data," in *Virtual Realities* (ser. Lecture Notes Comput. Sci.). Berlin, Germany: Springer-Verlag, 2015, vol. 8844, pp. 169–186.
[41] L. Kovar and M. Gleicher, "Automated extraction and parameterization of motions in large data sets," *ACM Trans. Graph.*, vol. 23, no. 3, pp. 559–568, Aug. 2004.
[42] R. Heck and M. Gleicher, "Parametric motion graphs," in *Proc. Symp. Interactive 3D Graph. Games*, 2007, pp. 129–136.
[43] D. Okwechime, E. J. Ong, and R. Bowden, "Mimic: Multimodal interactive motion controller," *IEEE Trans. Multimedia*, vol. 13, no. 2, pp. 255–265, Apr. 2011.
[44] N. C. Tang, C. T. Hsu, M. F. Weng, T. Y. Lin, and H. Y. M. Liao, "Example-based human motion extrapolation and motion repairing using contour manifold," *IEEE Trans. Multimedia*, vol. 16, no. 1, pp. 47–59, Jan. 2014.
[45] S. Jones and L. Shao, "Linear regression motion analysis for unsupervised temporal segmentation of human actions," in *Proc. IEEE Winter Conf. Appl. Comput. Vis.*, Mar. 2014, pp. 816–822.
[46] Y. Osmanlıoğlu, S. Dickinson, and A. Shokoufandeh, *Unsupervised Motion Segmentation Using Metric Embedding of Features*. Cham, Switzerland: Springer-Verlag, 2015, pp. 133–145.
[47] D. Holten, "Hierarchical edge bundles: Visualization of adjacency relations in hierarchical data," *IEEE Trans. Vis. Comput. Graphics*, vol. 12, no. 5, pp. 741–748, Sep. 2006.
[48] J. Bottger, A. Schafer, G. Lohmann, A. Villringer, and D. S. Margulies, "Three-dimensional mean-shift edge bundling for the visualization of functional connectivity in the brain," *IEEE Trans. Vis. Comput. Graphics*, vol. 20, no. 3, pp. 471–480, Mar. 2014.
[49] M. Vejdemo-Johansson, F. T. Pokorny, P. Skraba, and D. Kragic, "Cohomological learning of periodic motions," *Applicable Algebra Eng., Commun. Comput.*, vol. 26, pp. 5–26, 2015.
[50] D. W. Scott, *Multivariate density estimation: Theory, practice, and visualization* (Wiley Ser. Prob. Math. Statistics: Appl. Prob. Stat. Sec.). New York, NY, USA: Wiley, 1992..
[51] O. C. Jenkins and M. J. Matarić, "A spatio-temporal extension to isomap nonlinear dimension reduction," in *Proc. 21st Int. Conf. Mach. Learn.*, 2004, pp. 441–448.
[52] G. Guerra-Filho and Y. Aloimonos, "A language for human action," *Computer*, vol. 40, no. 5, pp. 42–51, May 2007.
[53] I. Junejo, E. Dexter, I. Laptev, and P. Perez, "View-independent action recognition from temporal self-similarities," *IEEE Trans. Pattern Anal. Mach. Intelli.,* vol. 33, no. 1, pp. 172–185, Jan. 2011.
[54] CMU, "Carnegie Mellon University Graphics Lab: Motion Capture Database," accessed on Apr. 21, 2016. [Online]. Available: http://mocap.cs.cmu.edu
[55] R. H. Chowdhury *et al.*, "Surface electromyography signal processing and classification techniques," *Sensors*, vol. 13, no. 9, pp. 12431–12466, 2013.
[56] S. Fothergill, H. M. Mentis, P. Kohli, and S. Nowozin, "Instructing people for training gestural interactive systems," in *Proc. ACM Annu. Conf. Human Factors Comput. Syst.*, 2012, pp. 1737–1746.
[57] S. Nowozin and J. Shotton, "Action points: A representation for low-latency online human action recognition," Microsoft Res. Cambridge, Cambridge, U.K., Tech. Rep. MSR-TR-2012-68, Jul. 2012.
[58] G. Yu, Z. Liu, and J. Yuan, "Discriminative orderlet mining for real-time recognition of human-object interaction," in *Proc. Asian Conf. Comput. Vis.*, 2015, vol. 9007, pp. 50–65.
[59] B. Krüger, J. Tautges, M. Müller, and A. Weber, "Multi-mode tensor representation of motion data," *J. Virtual Reality Broadcast.*, vol. 5, no. 5, pp. 1–13, Jul. 2008.
[60] M. Sattler, R. Sarlette, and R. Klein, "Simple and efficient compression of animation sequences," in *Proc. ACM SIGGRAPH/Eurograph. Symp. Comput. Animation*, 2005, no. 9, pp. 209–217.
[61] A. Vögele, M. Hermann, B. Krüger, and R. Klein, "Interactive steering of mesh animations," in *Proc. 11th ACM SIGGRAPH/Eurograph. conf. Comput. Animation*, 2012, no. 6, pp. 53–58.
[62] A. Yao, J. Gall, L. V. Gool, and R. Urtasun, "Learning probabilistic non-linear latent variable models for tracking complex activities," in *Proc. Adv. Neural Inf. Process. Syst. 24*, 2011, pp. 1359–1367.

Björn Krüger received the M.S. (Dipl.-Inform.) and Ph.D. (Dr. rer. nat.) degrees in computer science from the University of Bonn, Bonn, Germany, in 2006 and 2012, respectively.

From 2012 to 2015, he was a Postdoctoral Fellow with the University of Bonn. Since 2015, he has been with the Gokhale Method Institute, Stanford, CA, USA, as a Senior Researcher. His research interests include computer animation, computer graphics, machine learning, and motion capture.

Anna Vögele received the M.S. (Dipl.-Math.) degree in mathematics and the Ph.D. (Dr. rer. nat.) degree in computer science from the University of Bonn, Bonn, Germany, in 2011 and 2016, respectively.

She is currently a Research Assistant with the Department of Computer Science, University of Bonn. Her research interests include computer graphics and animation as well as machine learning.

Authorized licensed use limited to: FH Koeln. Downloaded on November 22,2022 at 10:34:35 UTC from IEEE Xplore. Restrictions apply.

Tobias Willig received the B.A.Sc. degree in computer science from the Bonn-Rhein-Sieg University of Applied Sciences, Sankt Augustin, Germany, in 2011, and the M.Sc. degree in computer science from the University of Bonn, Bonn, Germany, in 2015.

Angela Yao received the B.A.Sc. degree in engineering science from the University of Toronto, Toronto, ON, Canada, in 2006, and the Ph.D. degree in information technology and electrical engineering from ETH Zurich, Zurich, Switzerland, in 2012.

She is currently an Assistant Professor with the Institute of Computer Science, University of Bonn, Bonn, Germany. Her research interests include computer vision and machine learning, with special focus on human pose estimation and action recognition.

Reinhard Klein (M'00) received the M.S. (Dipl.-Math.) degree in mathematics and the Ph.D. degree in computer science from the University of Tübingen, Tübingen, Germany, in 1989 and 1995, respectively.

In 1999, he received an appointment as Lecturer (Habilitation) in computer science also from the University of Tübingen, with a thesis in computer graphics. In September 1999, he became an Associate Professor with the University of Darmstadt, Darmstadt, Germany, and the Head of the Animation and Image Communication Research Group with the Fraunhofer Institute for Computer Graphics, Darmstadt, Germany. Since October 2000, he has been a Professor with the University of Bonn, Bonn, Germany, and the Director of the Institute of Computer Science II, University of Bonn.

Andreas Weber received the M.S. (Dipl.-Math.) degree in mathematics and the Ph.D. (Dr. rer. nat.) degree in computer science from the University of Tübingen, Tübingen, Germany, in 1990 and 1993, respectively.

From 1995 to 1997, he was working under the scholarship from Deutsche Forschungsgemeinschaft as a Postdoctoral Fellow with the Computer Science Department, Cornell University, Ithaca, NY, USA. From 1997 to 1999, he was a member of the Symbolic Computation Group, University of Tübingen. From 1999 to 2001, he was a member of the Animation and Image Communication Research Group at the Fraunhofer Institute for Computer Graphics, Darmstadt, Germany.

Authorized licensed use limited to: FH Koeln. Downloaded on November 22,2022 at 10:34:35 UTC from IEEE Xplore. Restrictions apply.