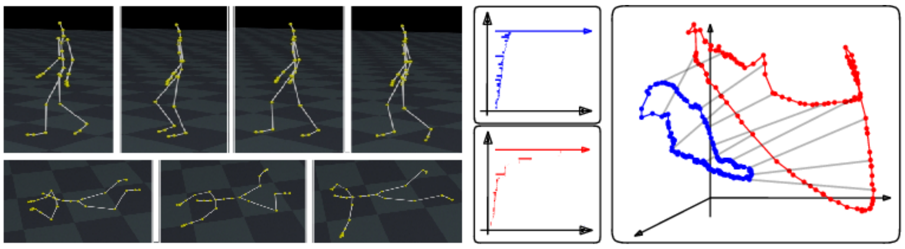
AAECC manuscript No.
(will be inserted by the editor)

Cohomological Learning of Periodic Motion



Mikael Vejdemo-Johansson · Florian T. Pokorny · Primoz Skraba · Danica Kragic

3 January 2014

Abstract This work develops a novel framework which can automatically *detect, parameterize and interpolate* periodic motion patterns obtained from a motion capture sequence. Using our framework, periodic motions such as walking and running gaits or any motion sequence with periodic structure such as cleaning, dancing etc. can be detected automatically and without manual marking of the period start and end points. Our approach constructs an intrinsic parameterization of the motion and is computationally fast. Using this parameterization, we are able generate prototypical periodic motions. Additionally, we are able to interpolate between various motions, yielding a rich class of 'mixed' periodic actions. Our approach is based on ideas from applied algebraic topology. In particular, we apply a novel persistent cohomology based method for the first time in a graphics application which enables us to recover circular coordinates of motions. We also develop a suitable notion of homotopy which can be used to interpolate between periodic motion patterns. Our framework is directly applicable to the construction of *walk cycles* for animating character motions with motion graphs or state machine driven animation engines and processed our examples in approximately one minute or at a rate of about 10 frames per second.

Mikael Vejdemo-Johansson · Primoz Skraba
AI Laboratory, Jožef Stefan institute, Slovenia
E-mail: primoz.skraba@ijs.si

Mikael Vejdemo-Johansson · Florian T. Pokorny · Danica Kragic
Computer Vision and Active Perception Lab, KTH Royal Institute of Technology, Sweden
E-mail: mvj@csc.kth.se, fpokorny@csc.kth.se, danik@csc.kth.se

1 Introduction

Periodic motion patterns are abundant both in animal and human behaviour. The human gait in particular is a classical example of a periodic motion which has been studied intensely in recent decades. Motion capture databases such as [1] provide a rich source of motion patterns which can be used to animate human characters in movies and computer games. For example, [2] use this type of data to construct a controller for a simulated human character, allowing the character to walk based on recorded motion capture sequences.

Despite the recent availability of motion capture equipment, it remains impossible to pre-record every variation of a motion for these applications. One approach to generate new motions is hence to interpolate between recorded movements.

For non-periodic motions, various interpolation schemes have been developed to adapt the motion to the physical constraints posed by the environment [3], or to vary the perceived 'style' of the motion. In robotics, pre-recorded human motions have also successfully been used in a 'learning from demonstration' setting to transfer motion patterns from a human demonstrator to a robotic platform [4]. Previous work on periodic motion generates prototypical motions from motion capture data by determining period start and end frames either manually or by means of a low-dimensional search procedure. This currently involves a fair amount of parameter tuning and manual intervention. An example of this approach is the recently released Mecanim toolbox in Unity 4 [5].

We propose a framework for dealing with periodic motions with enables automatic *detection, parameterization and interpolation* of periodic motion patterns. Our contributions[1] can be summarized as follows:

a) *Detection:* Our framework detects periodic motion patterns using persistent cohomology. Furthermore, our framework can be applied even when several different periodic motions are present in the input.
b) *Parameterization:* We compute a circle-valued coordinate map parameterizing high dimensional motion data and allowing us to determine an intrinsic description of periodic motions.
c) *Interpolation:* Our approach is capable of continuously interpolating between various periodic motions while maintaining periodicity.
d) *Computational efficiency:* Our approach is computationally efficient since no difficult optimization procedures are required. This enables us to process motions at a speed of 11 frames per second on a current laptop.

Our work relies on the efficient computation of persistent cohomology barcodes and representative cocycles to produce quantifiably reliable periodic features for detection, parameterization, and interpolation. By using a topological approach, we are able to find natural descriptions of motion sequences while working directly in a high-dimensional configuration space. This provides several advantages over more geometric techniques. In particular, we require fewer parameters to achieve similar results and gain increased robustness towards the choice of metric.

[1] Our contributions are also outlined in a video summary, accessible at `http://www.youtube.com/watch?v=NGQ-M2gdibQ`.

2 Related Work

Gait analysis and motion generation have received extensive attention in a number of different fields including graphics, computer vision, robotics and biomedical research. Since the amount of previous work is too large to give a complete account of here, we shall focus primarily on related work from graphics and, to a lesser extent, from computer vision and robotics. In this paper, we do not examine the problem of motion capture – rather, we assume as input articulated skeletal structures with corresponding joint angles that may have come from animation, captured video or other sensing equipment. Previous work on the characterization and generation of gaits fall into two broad categories: optimization-based and motion graph based – as well as numerous hybrids.

The optimization-based approaches use variational methods based on physical constraints to perform space-time optimization [6,7,8,9]. Due to the non-convexity of the optimization problem, these methods are used mostly in order to generate small alterations of existing motions. An alternate class of methods is based on motion graphs [10]. These methods generally assume that a relatively dense sample of example motions is available and find paths through these samples to generate new types of motions [11,12,13]. The problem of a large search space of possible paths remains and it is again difficult to determine a good path by optimization. Most of the methods on interpolation concentrate on a blending of given motions [14,15]. This often results in complex procedures involving time-warping and sequence alignment. More recently, there has been work towards interpolating and clustering motions based on more general techniques [16,17].

There has been significant work on characterizing cyclic behaviour in gaits in the robotics community from the view of control theory. For animation, there has been less work [18,19]. In particular, [19] applies a conceptually similar approach to the work presented here using the inherent circular nature of periodic motions for characterization. However, their method is also fundamentally different since it uses fitting techniques [20] rather than persistent cohomology.

A topological analysis has been applied to gaits before, in [21] and [22]. The work [21] applies a topological signature based on a persistence diagram to the problem of identifying a person using gait data. While we also use the persistence diagram in our work, the approach we develop is fundamentally different since it extracts more detailed information: we use a topological parameterization of the motion, rather than an abstract signature such as the persistence diagram. In the latter work, persistent homology was used to separate gaits into steps, i.e. periods of time between which the feet establish contact with the ground. Persistent homology was used to search for stable choices of thresholds for certain constraints. We use a similar step, however, we obtain a more complete parameterization of the gaits (as well as being agnostic to the particular sensors used for capturing the gaits).

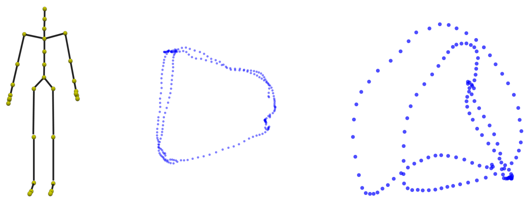


Fig. 1 The left figure displays the skeleton used in our experiments. The middle and right figure show PCA projections for the motions 69:01: WALK FORWARD (middle) and for 143:05: HOPPING (right). While the walking motion appears close to periodic in the PCA projection, the hopping motion represents a more challenging real world example.

3 Background

3.1 Representing periodic motion

Fundamentally, a motion of an object can be thought of as a trajectory in the object's configuration space. For humanoid figures, we model the configuration space $\mathcal{C}$ as in [1] by a 'root joint' which is placed at the hip in the skeleton – with translations along x, y, and z and Euler angle rotations around x, y and z; together with 56 additional joint angles of the various attached joints. Throughout the paper, we use the skeleton in Figure 1. Motions are then given as a sequence of points in $\mathcal{C}$, forming a piecewise linear curve. Even relatively simple motions can form very geometrically complex curves $\gamma : [0,1] \to \mathcal{C}$ in this space. Periodic motions, including the special case of an idealized perfectly periodic gait, determine a *closed* curve through the configuration space which – from a topological point of view – can be considered as a circle if it does not self-intersect. This is formalized in a result in [23] which implies that any periodic motion is a closed *simple* curve – *i.e.* without self-intersections and topologically a circle – if it is embedded in sufficiently high dimensions. In the remainder of this paper, we will refer to motions by their designation in the CMU-MOCAP database [1], a pair of numbers denoting (Actor:Motion).

3.2 Topological and geometrical background

At the heart of our method are techniques from applied topology and in particular *persistent cohomology*. In this section, we review the relevant definitions and concepts. Wherever possible, we will point readers to additional references with a more thorough discussion of the background material. The classical algebraic topology reference [24] goes into detail about cohomology and [25] is an excellent reference for applied topology and persistence. Let K be a simplicial complex – a collection of simplices such that for each simplex $\sigma \in K$, all the faces of σ are also in K and the intersection of any two simplices is either empty or a common face. A *filtration* is a sequence of increasing subcomplexes of K. Readers familiar with *persistent homology* will recognize this as the primary object of study. Persistence theory tells us that the p-th homology of a filtration has a decomposition into a p-th *barcode* or equivalently a p-th *persistence diagram*. For each p, we have a collection of bars (x_i, y_i), which intuitively represent the birth and death times of homological features. This has been widely used in applications and has been the object of independent study.

Rather that persistent homology, we use the dual construction - *persistent cohomology*. Before giving the formal definition, we note that the p-th persistent cohomology also studies a filtration of spaces and produces a p-th barcode/persistence diagram. In fact, in the case of simplicial complexes, it will produce an *identical* barcode.

Here, we highlight the main differences between homology and cohomology. Recall that, to compute homology, we study the *boundary operator*, ∂_k. For a simplex $\sigma = [v_0, \ldots, v_k]$ with vertices $v_0, \ldots, v_k$, the boundary operator defined by

$$\partial_k(\sigma) = \sum_{i=0}^{k} (-1)^i [v_0, \ldots, \hat{v}_i, \ldots, v_k],$$

where $\hat{v}_i$ indicates that v_i is omitted. We consider the coefficient field $\mathbb{Z}/p$ for some prime p. The boundary operator is then extended to a linear map $\partial_k : C_k \to C_{k-1}$ between the $\mathbb{Z}/p$-vector spaces C_k and C_{k-1} of formal linear combinations of k-simplices and $k-1$-simplices in the simplicial complex K respectively. The definition of k-dimensional homology is then given by $H_k = \operatorname{Ker} \partial_k / \operatorname{Img} \partial_{k+1}$. Cohomology is defined analogously, but rather than using the boundary operator, it applies the dual *coboundary operator* δ_k.

The coboundary operator $\delta_k : \operatorname{Hom}(C_k, \mathbb{Z}/p) \to \operatorname{Hom}(C_{k+1}, \mathbb{Z}/p)$ is abstractly defined by $\delta_k(\phi)(c) = \phi(\partial_{k+1}(c))$, for $\phi \in \operatorname{Hom}(C_k, \mathbb{Z}/p)$ and $c \in C_{k+1}$, and where $\operatorname{Hom}(C_k, \mathbb{Z}/p)$ denotes the dual vector space of $\mathbb{Z}/p$-linear maps from C_k to $\mathbb{Z}/p$. However, identifying $\operatorname{Hom}(C_k, \mathbb{Z}/p)$ and C_k, we can think of δ_k as mapping a k-simplex to a linear combination of its cofaces. So just as the boundary of an edge is a linear combination of the two end vertices, its coboundary is a linear combination of incident triangles. Likewise, the coboundary of a vertex is a linear combination of all its incident edges. Over field coefficients, the coboundary operator can then be defined as the transpose of the boundary operator: $\delta_k = \partial_{k+1}^T$. Persistent cohomology is defined analogously to persistent homology, but in terms of the coboundary operator – the definition of k-dimensional cohomology is $H^k = \operatorname{Ker} \delta_k / \operatorname{Img} \delta_{k-1}$.

We now recall some topological constructions which we require before explaining why we work with cohomology rather than homology for our application. Our input is a motion which is represented as a trajectory in a configuration space. In practice, the motion is described by a sequence of poses where each pose corresponds to a point in the configuration space, which leads to a piecewise linear (PL) curve. To construct a simplicial complex, we compute the *Vietoris-Rips* filtration on the set of points, i.e. input poses.

Given a set of points and a metric, the Vietoris-Rips complex with filtration parameter α constructs the graph which connects points (x, y) with an edge if and only if $d(x, y) \leq \alpha$. Higher dimensional simplices are inserted for all cliques in the graph (e.g. triangles for all 3-cliques, tetrahedra for all 4-cliques, etc.). This filtration is obtained by increasing α from 0 to $+\infty$, although in practice we do not construct the full filtration. This construction comes with provable approximation guarantees relating it to the topological space defined by the union of balls of radius α around the data-points and is relatively straightforward to compute even in high dimensional spaces (such as in the case of human motion capture data). There are a number of potential choices for metrics. We primarily use the Euclidean metric between points or a modified Euclidean metric where points in adjacent time steps are first connected (i.e. adjacent points are effectively at distance 0). This modified metric then incorporates the temporal information in the sampled motion.

Details on the algorithm for persistent cohomology and the construction we use can be found in [26]. Even though the resulting barcodes are equal, homology and cohomology have fundamentally different representatives – homology is represented by structures inside the shape we measure, while cohomology is represented by maps from the shape we measure. These maps are of fundamental importance since they will be thought of as coordinate functions in our application. Finally, this requires that the choice of coefficient field is different from $\mathbb{Z}/2$. For this application, we use $\mathbb{Z}/p$ for some small prime $p > 11$.

4 Processing Pipeline

We process motions using the pipeline outlined in Figure 2. The input to our approach is given by a sequence of points describing a piecewise linear (PL) motion trajectory in $\mathcal{C}$ and our approach can be split into three main stages: *preprocessing, parameterization and output.*

To illustrate examples, we use a 2-dimensional principal component analysis (PCA) as in Figure 1. This is for visualization and illustration purposes only: unless explicitly stated otherwise, we carry out all computation in a high-dimensional space with 62 dimensions[2].

4.1 Preprocessing

A typical motion, for example a walk, will have a spiral-like trajectory. Most of the coordinates will be almost periodic, reflecting the periodic nature of the walk, while the translation coordinates describing the absolute hip position destroy this periodicity. One approach would be to remove the translation component, collapsing the spiral onto a circle. Here, we will instead replace the position and orientation coordinates with first order differences (approximating the time derivative) allowing us to more closely preserve the periodic structure in our experiments.

4.2 Parameterization

The pipeline is based on the cohomology tools developed by [26]. For a set of vectors $\mathbb{X} \subset \mathcal{C}$ describing measurement points representing a piecewise linear close to periodic trajectory, we can compute a coordinatization map $\mathbb{X} \to \mathbb{S}^1$, assigning to each data-point in $\mathbb{X}$ a coordinate on the circle $\mathbb{S}^1$. The right choice of embedding of the motion data points $\mathbb{X}$ (for example using a delay embedding) will give mostly

[2] Recall that our human motion input data lies in a 62 dimensional configuration space $\mathcal{C}$.

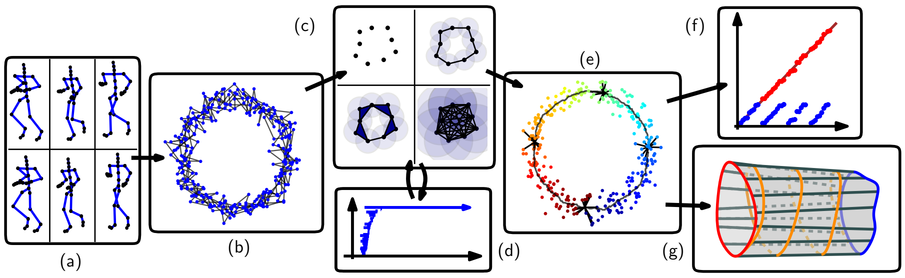


Fig. 2 We display our topological processing pipeline for periodic motions: The input motion capture data (a) with a near-periodic trajectory (b) in a high-dimensional configuration space is visualized (in 2D) in the first two steps. We then construct a Vietoris-Rips filtration (c) and compute its persistent cohomology (d). We then produce circle-valued coordinate functions (e) and use these functions for several applications including describing the motion itself (f) and its periodicity, as well as to automatically align motions for blending or interpolation (g).

periodic motions the topological type of a circle [23]. A coordinatization map from the cohomology computation above will then capture the geometry and topology of the resulting trajectory.

Given an input motion, we treat the segment start and end-points of the piecewise-linear trajectory as a point cloud $\mathbb{X}$ and discard the temporal trajectory information. Next, we compute the Vietoris-Rips complex filtration $K_\alpha = V_\alpha(\mathbb{X})$ up to a maximal threshold and determine the first barcode, corresponding to the first persistent (co-)homology groups as in [26]. Each bar corresponds to a coclass representing a possible choice of circle-valued coordinate induced by the data as well as an associated filtration interval. As in persistent homology, the larger bars correspond to more prominent features which are stable under noise.

If we expect to see only one periodic motion, we identify only the most persistent coclass. Likewise, if we expect to see k motions, we would choose the k most persistent coclasses in the barcode. We found that, for our motion capture data, the motions were furthermore well-sampled enough so that adjacent points in the time series/trajectory were also always connected in the Vietoris-Rips filtration at the filtration values we determined by investigating the barcode in the above manner.

In practice, we often see a gap between the persistent classes corresponding to periodic motions and other classes. Currently, we choose the number of classes we select manually, however, we believe that ultimately this could be chosen automatically.

For simplicity, we assume here that there exists a single periodic motion in each input trajectory. Once the desired coclass is determined, a sufficiently large filtration parameter is chosen such that the coclass has been born but has not died yet. We find that better results are obtained with with relatively high filtration values within this range.

To compute a parameterization of the circle in the form of circular coordinates $\phi : \mathbb{X} \to \mathbb{S}^1$, we must choose a persistent cohomology class $[z]$, represented by a 1-cocycle z, and a scale parameter α such that the coclass is not trivial in $H^1(K_\alpha)$. The coclass is then lifted to $\mathbb{R}$ coefficients [26] and any other representative of the same coclass is then some $z + \delta w$ for $w : \mathbb{X} \to \mathbb{R}$. As explained by [26], minimizing the L_2 norm of the representative produces a smoothest possible coclass and simultaneously the coordinate function as $w \pmod{1.0}$. The optimization problem is given as

$$\arg\min_{w} \|z + \delta w\|_2.$$

The result of this computation is a coordinate value in the interval $[0, 1)$ for each point in the dataset $\mathbb{X}$. The spacing of these coordinate values, however, is induced by the geometry of the point cloud, not the original frame rate of the input time series. It is worth noting the structure of the coordinate value histogram, as we show in Figure 3: the circular coordinate picks out a geometric description of the closed curve – but a bipedal gait does not flow with even speed throughout: the pose is largely unchanged for parts of the step, and changes rapidly for other parts of the motion. This is reflected in the bimodal distribution of the period – resting on the left foot and resting on the right foot correspond to one peak each, while the transition between the rest states is more hurried, and corresponds to the two valleys.

4.3 High-quality coordinate

For motion samples that contain a single period, our work with cohomology and circle-valued coordinates has demonstrated that the best results in terms of a smooth and even coordinatization are achieved when the topological type measured by the persistence diagrams is that of a circle. This has two concrete consequences. On the one hand, the computational parameter for the persistent cohomology computation has to be chosen large enough to find a circle-type signature: one single connected component (as measured by 0-persistence) and one single tunnel/loop (as measured by 1-persistence). On the other hand, the embedding of the input data we work with must actually display this sort of signature. In particular, the trajectory should not have self-intersections (returning to a previous pose before a full period is done), and each repetition of the motion should be close to the previous repetition (for instance moving the head from one side to another will decrease or destroy the quality of the topological signal).

For cases when several periodic motions are detected in the same motion sample, we found that the best coordinates were obtained near the end of their respective lifetimes; a high filtration appeared to improve the coordinate quality, and the persistence lifetime of a coordinate helped to judge its quality. With this map in hand we showcase the following applications:

1. Construction of a “typical” parametrized periodic motion from the input data.
2. Interpolation and transitioning between periodic motions.
3. Determination of different types of periodic sub-behaviour from an input motion.

4.4 Producing a typical trajectory

We now describe how to generate typical trajectories from the circular parametrization ϕ of the point cloud $\mathbb{X}$ describing the motion data. For the example we introduced in Figure 1, the corresponding coordinate ϕ is described by the color mapping in Figure 3. To produce a typical periodic trajectory, we proceed as follows:

1. Since the circular coordinates of the points for each frame of a motion are not necessarily uniformly distributed, we build an approximation of the circular coordinate density using a Gaussian Mixture Model and sample N new points from the reconstructed density. By increasing N, the resolution of the synthesized motion can be increased as desired. An example of such a probability distribution and its fit is provided in Figure 3.
2. For each sampled circular coordinate $\theta \in \mathbb{S}^1$ in order, determine the k points $x_1, \ldots, x_k \in \mathcal{C}$ whose circular coordinates $\phi(x_1), \ldots, \phi(x_k) \in \mathbb{S}^1$ form k-nearest neighbours of θ. The parameter k should be higher than the number of periods in the dataset, but not high enough that the points x_i stretch out too much along the trajectory. In Figure 4, we illustrate how picking the 10 nearest neighbours of a sampled coordinate value provided a small cluster of points in $\mathcal{C}$.
3. For the points $x_1, \ldots, x_k \in \mathcal{C}$, we compute the centroid $c \in \mathcal{C}$ This centroid then yields the next point in the periodic trajectory. In Figure 4, we illustrate this step and depict the resulting centroid curve acquired by applying this approach for 500 sampled θ values.

The centroid here can be replaced by other means of averaging points – Gaussian weighting produces similar results. We estimate the probability distribution on the circle by using Gaussian mixture models, but other methods can also be considered. Finally, the curve described by the computed centroids is resampled to correspond to the frame rate of the original data before the results are saved for further use.

The process is somewhat sensitive to the quality of the coordinate function used. Looking at Figure 1, we observe that the dataset in the middle is reasonably simple to handle – the several repetitions of the motion are close enough so that the topological analysis picks out a thick tube from the time series. The right hand plot however indicates a potential problem: averaging points across the two separated parts of trajectories like these tended to produce jittery output. Taking k nearest neighbours of a coordinate value produces points on both curve parts, and the two curves pull the constructed averaged periodic trajectory back and forth.

4.5 Interpolation

To interpolate between a set of input motions $\mathbb{X}_1, \ldots, \mathbb{X}_n$, which are approximately periodic, we first extract a typical continuous parametrization for each of them using the procedure just outlined. This results in piecewise linear periodic trajectories $x_i : \mathbb{S}^1 \to \mathcal{C}$ with coordinates on the circle $\mathbb{S}^1$ given by $[0, 1)$, and for each $i \in \{1, \ldots, n\}$. We furthermore assume that each x_i has been resampled and discretized using the same number of piecewise linear segments. To interpolate between the input motions, we consider convex combinations. That is, given weights $w_i \in \mathbb{R}_{\geq 0}$ such that $\sum_{i=1}^n w_i = 1$, we define the interpolated periodic motion $y : \mathbb{S}^1 \to \mathcal{C}$ by

$$y(\theta) = \sum_{i=1}^{n} w_i x_i(\theta).$$

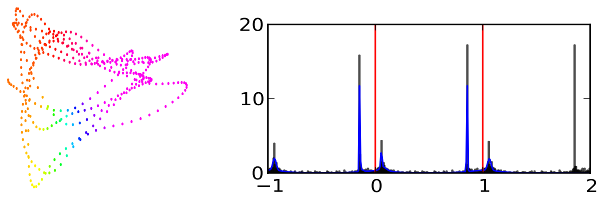


Fig. 3 The left figure displays a circular coordinate for the motion 143:28. On the right, we show a Gaussian mixture model fitted to three copies of the coordinate values on $[-1,0]$, $[0,1]$ and $[1,2]$. The distribution is uneven, reflecting the fast and slow segments of the motion.

We note that this may generate impossible motions with self-intersecting skeleta. In the examples we used, this did not occur, but it could in more complicated motions such as dancing. Self-intersections could however be detected using forward kinematics. In future work, we will investigate how to locally modify the resulting trajectories into feasible non-intersecting motions.

Note that the interpolation scheme can be applied regardless of the number of input motions. Furthermore, to generate a transition between two motions, we vary the weighting parameter from 0 to 1, generating a homotopy between the two topological circles parametrizing each of the motions. At the end of this process, our approach hence produce as output a number of loopable parametrized motions from the input data which can be interpolated to obtain novel periodic behaviours.

4.6 Picking out different motions.

When computing the barcode for the first persistent cohomology groups, zero or more large bars may exist. We distinguish two cases: when a single interval of dominant length exists and when there are several such long intervals at a given filtration range.

In the simple case, when there is a clear single periodic motion in the data, we can expect to see barcode diagrams like in Figure 4 with a single long bar. For this case, the methods described above worked well in our experimental evaluation and we were able to extract topological information from the motion capture data and generate typical periodic trajectories. Most of the motion sequences we have analyzed in Figure 7 are of this type. For some complex motions in [1], the raw motion capture file however contains more than one motion type in sequence.

These complex motion sequences can be recognized by the behaviour of the barcode: a sequence of disjoint periodic motions in a single motion capture tended to produce a barcode as in Figure 8 or as in Figure 9, where there are several long intervals in the barcode. Each of these intervals corresponds to a different circle-valued coordinate: a different way to map the entire sequence to a circle, emphasizing some part of the motion sequence. As can be seen in Figure 8, the coordinates in a sequence of periodic motions create signatures that can be used to identify sufficiently similar repetitions of a motion. By restricting the analysis to the *excitation regions* – connected parts of the timeseries exhibiting most variation in circular coordinate – we are able to extract a periodic portion of the motion for further analysis.

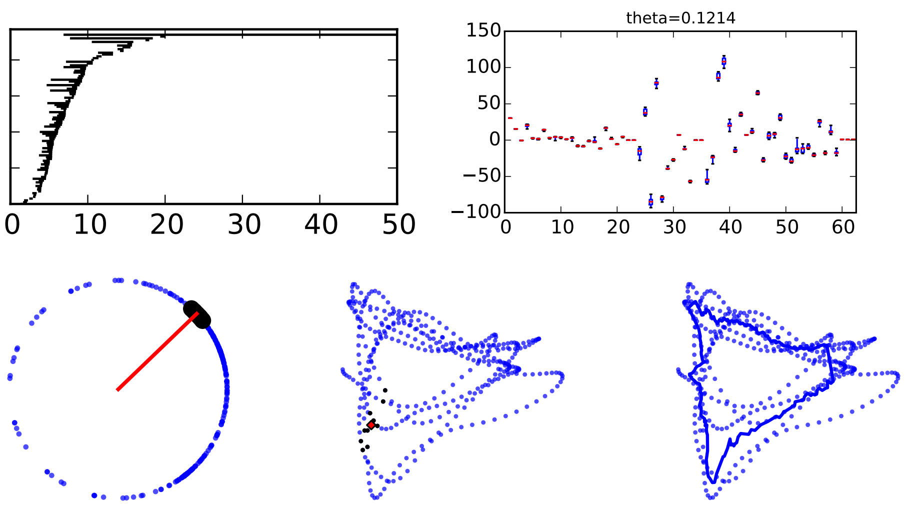


Fig. 4 The top left figure displays the first barcode diagram illustrating available circle-valued coordinate candidates for the motion 143:28. For any parameter value over 20 and up to the maximal parameter of 50, a single bar remains. In the top right figure, we consider the 62 dimensional coordinates for the 10 points nearest in circular coordinate to $\theta = 0.1214 \in [0, 1)$. We display the means of these 10 points in red and also show upper/lower quartiles (blue box) as well as the full range of the coordinates (as black bars). The bottom left plot displays the neighbours in the θ domain on the circle, while the middle plot shows a PCA projection of the data points with the nearest points (black) and the centroid (red). The bottom right figure displays the PCA projection of the extracted typical trajectory as a solid line.

5 Implementation

In this section, we describe additional details of the implementation of our approach.

5.1 PCA Projections

A problem that can make the recovery of a clear topological signal challenging in a high dimensional space such as the 62 dimensional human pose configuration space is the presence of many changes in individual joint angles between periods and trajectory parts. The actor may for example have straightened her spine while walking, or turned her head somewhat. In these cases, the topological signal might appear clearer when the dimensionality of the data is first reduced using a PCA projection. In the high-dimensional space, on the other hand, the filtration parameter required to merge all the individual periodic trajectory parts may be too large and destroy the periodic structure.

Consider, for instance, the motion 142:15, whose 2 dimensional PCA projection is depicted in Figure 5. The motion has four iterations of the same pattern, but one of the four is different in some aspects, producing the curve that, in the full-dimensional configuration space, is highly separated from the remaining iterations (see the lower left part of the leftmost picture in Figure 5). When extracting a prototypical periodic trajectory from this data, the separated curve parts pull the solution trajectory towards

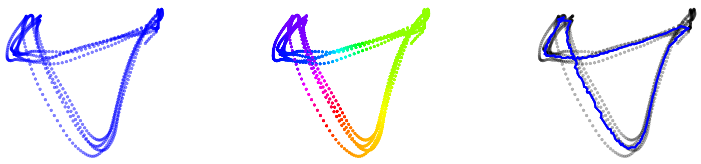


Fig. 5 We display a projection of Motion 142:15. The quality of the circular coordinate for this point-cloud was improved by a 3D PCA projection before the typical periodic trajectory was extracted. One iteration of the trajectory is situated somewhat apart from the others and gives rise to a separate coclass up to high values of the filtration parameter. Projecting the point-cloud to 3 dimensions reduces the distances between data points and improves the topological signal. The middle figure displays the resulting circular coordinate values of a 2D projection of the data in color, while the figure on the right shows a projection of the resulting constructed periodic motion as a solid line.

this outlier. The separated curve segment furthermore produces a coordinate function of its own and decreases the quality of the other topological coordinate. By projecting the trajectory data instead onto 3 dimensions using PCA coordinates, the distance between this stray curve and the rest of the data decreases. It is then possible to produce the topological coordinate we can see (in color) in the 2D projection in the middle plot and the resulting typical trajectory whose 2D projection is shown in the right plot.

Since the PCA projection is linear and therefore continuous, nearby points remain close in the projection. The projection may however map distant points close to each other. Some care must hence be taken to ensure that self-intersections are not introduced, which in practice meant not to reduce the dimension too much. The 2-dimensional illustration in Figure 5, for instance, has several self-intersections of the trajectory, but a projection to 3 dimensions turned out to be sufficient to maintain the circular structure of this trajectory. Projecting to a very low dimensional space may however result in the underlying curve having a small topological feature size. Therefore, care has to be taken when adjusting the PCA dimension.

5.2 Extracting and parameterizing periodic trajectories

Since the parameterization of our topological circular coordinates is not directly related to the speed of the motion, a direct parameterization in terms of these would introduce an arbitrary time-warp and lead to an unnatural looking motion. Instead, we consider the density of the circular coordinate values of the motion capture trajectory and sample new coordinate points from this density. To use a simple and standard Gaussian mixture density estimation, we replicate three copies of the coordinate values – the two extra copies are translated by ± 1, producing a density as depicted in Figure 3. Fitting a density estimator to these three periods, we arrive at a density whose middle part is close to periodic. The sampling only uses the central region $[0,1)$ to produce points. We sample at a much higher rate than the original input motion and downsample at the end to maintain the speed of the motion. More sophisticated periodic density estimations could have been employed, but we found that the Gaussian mixture estimation approach provided results of sufficient quality.

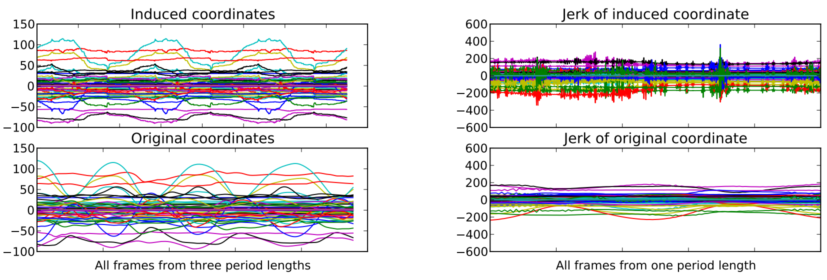


Fig. 6 We display motion 143:28. On the left, we show the 62 original and induced coordinates for 3 periods. On the right, we display the jerk of the induced and original coordinates for 1 period.

Once the circular coordinate samples $S \subset [0, 1)$ are produced, we assign a pose to each sample $\theta \in S$ by finding find the k points in the original input with the closest circular coordinate value and take the centroid c of these poses. The centroid pose c is then assigned to the coordinate value θ. In the top right part of Figure 4, we consider an example of k-nearest neighbours for a given circular coordinate value. We see that the neighbours have tightly distributed coordinate values in each dimension. For coordinates that take on values in $[0, 360]$ (degrees), very few have a total spread of more than 5 degrees.

6 Experimental evaluation

We tested our proposed method on motion captures from [1]. We selected 24 motion capture sequences from the database and applied our entire processing pipeline. Our selections, computational settings, and results are shown in Figure 7. For 8 of the motions, results were improved by using PCA projections. Our experiments used Dionysus from [27] for the cohomology computations and SciPy [28] for pre- and post-processing.

6.1 Parameter estimation evaluation

Even though our approach does not include many parameters, there are still some choices to be made: the filtration parameter for the Vietoris-Rips complex, methods for probability density estimations, etc. which we shall discuss now. For the sample selections, we picked 2–5 repetitions of a motion, and 90–900 frames, with most motions spanning 200–500 frames. We used an initial Vietoris-Rips filtration parameter of 50.00, increasing or lowering it as indicated by the persistence diagram. For example, if multiple connected components persisted, the Vietoris-Rips filtration parameter had to be increased – see HOPPING (143:05) in Figure 7. Alternatively, if the most persistent cohomology class died before 50, a smaller value was chosen - e.g. WALK (142:15) in Figure 7. A complete list of our choices can be found in Figure 7.

To estimate the probability density of coordinate values on the circle, we used a Gaussian mixture model from scikit-learn [29], with 30 components running 100

	Capture file	Capture frames		PCA	Cohomology				Post	
Motion type	ID	Start	Stop	dim's	ϵ born	ϵ end	# simplices	Total (s)	proc. (s)	fps
playground	01 : 09	0	4238	–	8.98	50.00	1 946 901	64.21	–	–
swordplay	02 : 07	0	2247	–	23.04	50.00	8 285 462	108.86	–	–
dance steps	05 : 13	0	1091	–	40.92	50.00	446 979	7.04	–	–
punching	13 : 29	0	1200	–	19.36	30.00	817 463	11.04	–	–
walk forward	69 : 01	200	450	–	9.95	50.00	290 566	7.07	18.71	9.70
walk sideways	69 : 42	100	250	–	16.77	28.00	30 292	0.53	12.09	11.89
careful run	77 : 10	225	325	–	37.34	59.90	6 485	0.11	12.07	8.21
regular walk	104 : 19	225	550	–	8.18	50.00	326 580	7.09	18.90	12.50
breast stroke swim	125 : 01	1300	1900	–	14.78	100.00	1 606 061	43.77	50.84	6.34
butterfly stroke swim	125 : 05	1500	2200	–	45.67	72.00	1 180 629	30.61	44.70	9.29
arrogant walk	142 : 04	250	650	3	8.06	40.00	374 216	2.96	24.14	14.76
depressed walk	142 : 05	300	1000	3	1.58	24.00	2 693 262	27.01	75.42	6.83
aggressive walk	142 : 06	300	600	3	8.32	40.00	121 898	1.21	18.49	15.23
walk with hand-rail	142 : 07	450	1000	2	1.22	14.00	2 153 124	21.60	61.84	6.59
walk	142 : 08	300	600	–	15.80	50.00	183 922	3.76	15.87	15.28
military walk	142 : 11	300	800	–	9.60	39.00	193 445	4.51	14.83	25.85
limping walk	142 : 12	310	710	2	3.12	30.00	995 299	8.43	36.68	8.87
walk	142 : 15	300	1200	3	2.11	20.00	4 212 216	44.06	122.37	5.41
jogging	143 : 01	0	96	–	24.83	50.00	3 334	0.07	11.41	8.36
hopping	143 : 05	100	240	–	58.38	105.00	32 014	0.53	11.83	11.33
stepping over	143 : 15	450	587	–	34.51	85.00	27 740	0.49	11.94	11.02
stepping over	143 : 16	52	186	–	52.07	110.00	35 962	0.78	12.09	10.41
stairs	143 : 17	394	603	–	34.06	50.00	25 471	0.43	12.29	24.37
sitting, standing	143 : 18	28	763	–	16.28	50.00	1 810 614	51.99	61.73	6.46
punching	143 : 23	100	500	4	2.66	30.00	844 627	12.72	–	–
using a broom	143 : 28	40	441	–	7.02	50.00	188 371	1.58	20.35	18.29
walk	143 : 32	30	270	–	14.06	50.00	128 308	2.41	14.10	14.54
sideway walk	143 : 40	1	425	–	16.31	40.00	548 524	13.48	25.30	10.93
sideway hopping	143 : 41	80	380	2	3.06	20.00	473 866	3.81	25.59	10.20

Fig. 7 We display the chosen motion capture examples with computation times (in seconds). Each motion has a description and a CMU dataset id (*Actor:Motion*); frames which were used; PCA dimension (if used); Vietoris-Rips parameter of relevant coclass birth; maximal Vietoris-Rips parameter used; complex size; total cohomology computation time and finally total post-processing time.
The motions 1:9, 2:7, 5:13, 13:29 and 143:23 were computed to illustrate the use of circular coordinates as detection tools. Thus, the post-processing pipeline was not applied on these motions, and we report no timings for these.

expectation maximization iterations and using a minimum covariance of 10^{-5}. For the centroid curve construction, we computed the 10 nearest neighbours of each of 500 sampled coordinate values. We computed centroids as coordinate-wise arithmetic means.

6.2 Running time

The timings were performed on a MacBook Air with a 2GHz Intel Core i7 processor and 8GB RAM. As can be seen in Figure 7, the entire pipeline runs within a few minutes – from 11.48 seconds for motion 143:01 ranging up to 166.63 seconds for motion 142:15. We account for the time spent in two different categories: speeding up the computation of cohomology is an active research field, and while we use state of the art software for this, there are approaches that speed up the computation which we have not used: witness complexes [30] and, for the PCA projected coordinates, alpha complexes [31,32].

As for the post-processing, which in many cases far outweighs the cohomology computation in computational effort, we note that we have written our entire code-base in Python using SciPy. Our source-code has not been optimized beyond some attempts to use the SciPy support for vectorized arithmetic. A re-implementation of the post-processing steps with attention paid to optimization and speed would certainly improve the running times.

As implemented, the pipeline however runs swiftly enough that it can be used directly with motion capture data – although not quite in real time.

6.3 Evaluating the results

To evaluate our results, we animated the motions using a humanoid skeleton, the mocapPlayer [33] for quick evaluations, and Maya [34] with a motion capture rigging as described by [1].

In addition, we inspected PCA scatter plots, plots of the extracted centroid curve and plots of coordinate values and computed jerk ($\dddot{x}$) to judge the smoothness of the motions. We computed an estimate of the jerk using a 4th order forward finite difference coefficient formula for third derivatives as given by [35].

The resulting coordinates for motion 143:28, are shown in Figure 6. We would like to point out that, while the linearly interpolated reconstructed coordinates are more jittery, most of the visible trends follow the periodic motion in the same speed as the original coordinates. In Figure 6, we also compare the jerk ($\dddot{x}$, the time derivative of the acceleration) of the two motions, and we note that the magnitude of the jerk in our generated periodic trajectory is similar to the magnitude of the jerk in the original data.

For comparison, we also implemented the techniques from [19] which produced similar results only when applied to pre-cut periodic segments. This is unsurprising as the key difference between our work and [19] is how the parameterization is found. The technique used in [19] for parameterization has some significant drawbacks. First, the motion segment must be periodic and this period must be known. In the experiments, we found that an incorrect period length, or data which contained non-periodic parts produced poor results with the approach of [19]. To determine a periodic motion for use with the methods of [19], manual cutting is hence required. For example, in many of the motions, the actor begins by standing for several seconds before starting to walk. Furthermore, using several periods of a motion as input to [19] also produced poor results in our tests.

One interesting advantage of the techniques used in [19] is a consequence of only using one example period from the motion capture to perform the interpolation. If there are large variations between the periods of the gaits, this prevents interpolations from producing unnatural poses. We expect that an incorporation of the techniques from [19] and other work in this area could further improve results. In particular, better interpolation and a reduction of foot skate would likely result in more natural motions. However, our approach does currently already produce comparable results with fewer parameters and is more robust to the choice of values for these parameters.

6.4 Multiple motions

We consider the example of different punching motions of 143:23 from [1] in Figure 8. Using a subsequence consisting of 400 frames and computing circular coordinates of those (using a Euclidean metric where each dimension was normalized by variance), we find 5 highly persistent classes – though the sequence only contains 3 different types of punches. The indicated red coordinate excitations correspond to an uppercut punch with the right hand, while the other two excitations correspond to a left uppercut and a jab. The various circular coordinate excitations can differentiate between the two types of left punches also – even though this relationship is not a straightforward correspondence.

Another example is produced by concatenating several motions into a single motion file. We combined motions 104:19 (REGULAR WALK), 143:01 (JOGGING) and 143:05 (HOPPING) into a combined sequence and computed coordinates for the entire sequence. In Figure 9, we can see the resulting coordinates at two different scales. We found that the lower scale captured the slower walking and running motions well, but missed the faster jumping motion at the end, while the larger scale missed the slow motions and detects the fast motion instead.

More examples from the database [1] have been included. In Figure 10, we see a sequence of playground activities. The initial walking motion, and one sequence where the actor dangles his feet off a ledge, are detected by spikes in the computed coordinates. In Figure 11, a sequence of sword slashes have been recorded. At the scale used, we pick out coordinates for a subsequence of mid-height slashing motions. Finally, in Figure 12, we see a recording of a ballet dancer moving through a few steps and two pirouettes. The chosen filtration parameter picks out the preparatory posture adjustment and the first pirouette.

In our experiments, all motions have been analyzed using the full dimensional human skeleton configuration space. By weighting the components, or even excluding some dimensions, results could potentially be improved further.

7 Discussion and Future Work

This paper introduced a new framework for extracting a periodic motion parametrizations from real world motion capture data, resulting in closed curves in a high-dimensional configuration space which describe a periodic motion in a format that can be used for procedural animation. By using topological information at its core, our method can be applied in high-dimensional spaces and executes the complete processing pipeline in a matter of a few minutes. The result is a natural and intrinsic parameterization of motions.

We have presented several applications using our approach: identifying periodic motions, generating typical periodic motion cycles and automatic alignment and interpolation between motions. The parameterization identifies periodic parts of the motion and gives us a natural way to find a typical periodic motion parametrization as well as the ability to estimate the period. With access to the found intrinsic parameterization, aligning motions then becomes a 1 dimensional search problem.

In this paper we have concentrated on our general approach and the information we can extract with it. We believe this raises a number of interesting questions:

Motion synthesis: Our approach to synthesizing typical motions once a topological circular parametrization is found is currently somewhat simple. It is orthog-

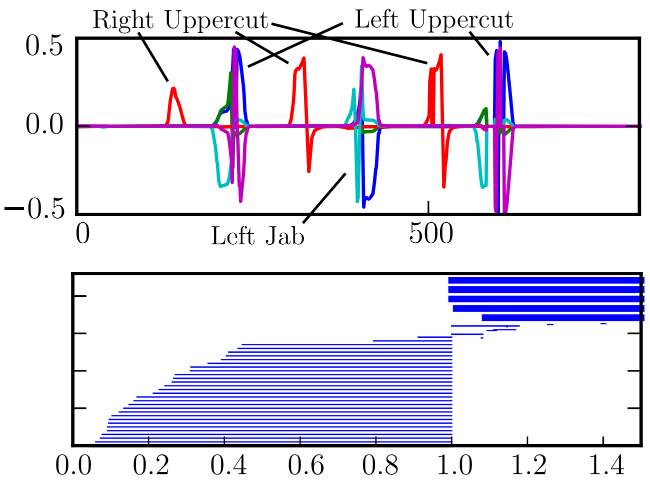


Fig. 8 We consider a sequence of punches producing several periodic motions. In the barcode (using Euclidean distance normalized by variance), we see 5 persistent cohomology classes (highlighted in the diagram) for 3 punches. The right uppercut is assigned 1 coordinate, while the left uppercuts and jab are assigned the remaining 4. However, the signatures of the jab and uppercut are significantly different.

onal to most of the existing work in animating gaits. Visually, the results could be further improved by using optimization-based post-processing taking physical constraints into account. This is especially the case for walking motions, where rather than integrating the differences, we could detect contact points with the ground to modify the resulting motion accordingly [36]. We currently do not take into account possible forbidden configurations, such as arms passing through each other. This is however a well studied problem in robotics and it may be possible to locally modify the trajectory to a physically realizable path while respecting the overall topological structures.

Boosting the Topological Signal: We experimented with primarily Euclidean metrics (normalized and un-normalized). There are numerous other metrics including weighting joint angles by the mass or inertia of the elements that the joint is driving [37]. As well as different metrics, complexes other than the Vietoris-Rips may also provide a finer distinction between motions. This could especially improve the performance for the case of multiple motions. Multiple coordinates corresponding to one motion indicate that we have chosen the coclass poorly, i.e. it might be possible to determine a better coordinate by considering linear combinations of coclasses. Another approach we would like to consider in future work is to learn a distance metric based on labelled input examples.

Classifying Motions: Persistence barcodes have been used successfully as shape signatures for clustering and shape comparison. We believe that our approach could also provide an interesting signature for comparing and clustering databases of motions. From the multiple motion experiments, we see that similar motions (e.g. running and walking) occur at the same filtration scale, or have similar coordinates (left

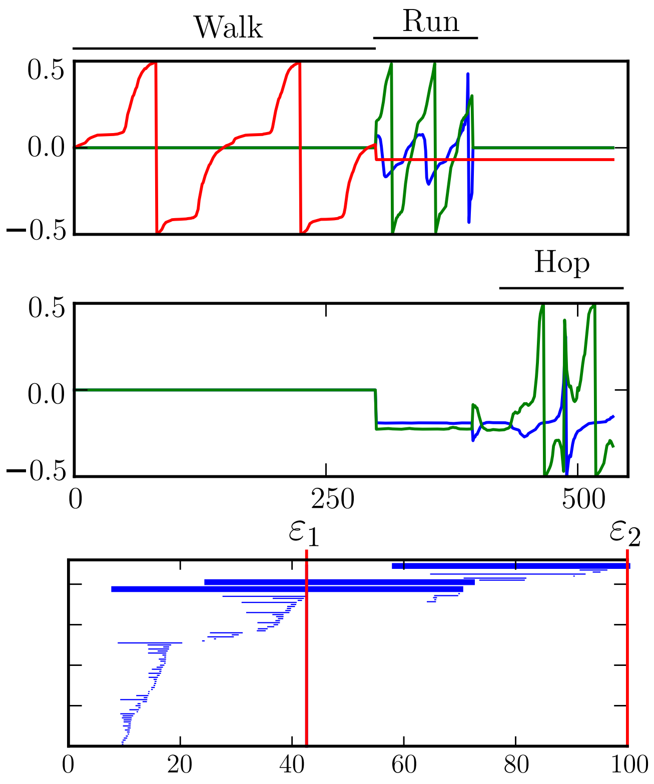


Fig. 9 We consider three concatenated motions: a walk, a run and hopping. The barcode detects the motions at different scales. Two of the motions are detected at filtration value $\epsilon_1 = 42.6$ (top figure) and one at $\epsilon_2 = 100$ (middle figure). The hop and run are described by two coordinates each. The relevant classes are highlighted in the first barcode diagram in the bottom figure.

versus right punch). We could therefore consider defining metrics between motions by examining how these coordinates change over time.

In summary, we have presented a periodic motion processing pipeline based on novel topological techniques for motion synthesis and classification that can be directly applied in high-dimensional configuration spaces. Our method has few parameters and does not rely on heavily preprocessing input, such as manually picking precise start and end frames. Our approach can detect and parametrize typical periodic motions, and it can extract periodic motion segments from real world motion sequences with minimal user intervention.

References

1. CMU Graphics Lab, "CMU graphics lab motion capture database." `http://mocap.cs.cmu.edu/`, accessed November 2012, 2012.

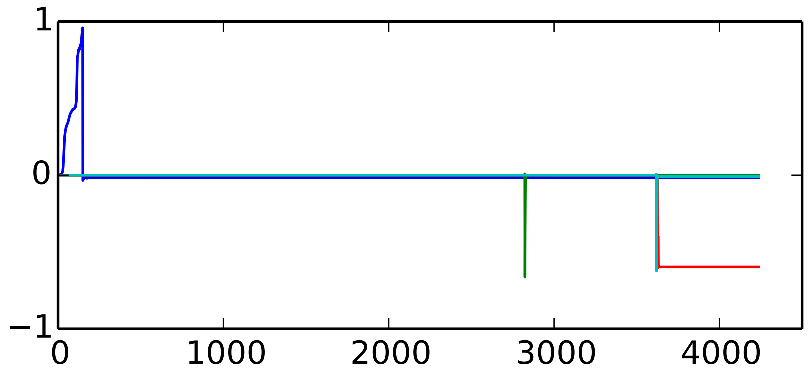


Fig. 10 We consider motion 1:9 (PLAYGROUND ACTIVITIES) with four detected periodic motion indicator functions. The motion consists of walking on plane ground (frames 0–293, 1961–2136); climbing a ladder (294–617, 2137–2745); sitting with dangling feet (618–1483, 2745–3880) and jumping down (1484–1960)

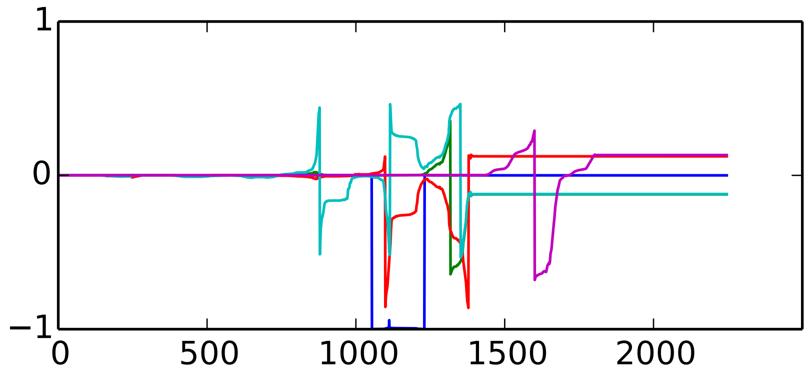


Fig. 11 We consider motion 2:7 (SWORDPLAY) with four detected periodic motion indicator functions. The motion consists of 11 different sword slashes in sequence, transitioning at frames 280, 515, 750, 1020, 1235, 1445, 1500, 1680, 1815, 1945, and 2025.

2. U. Muico, Y. Lee, J. Popović, and Z. Popović, "Contact-aware nonlinear control of dynamic characters," in *SIGGRAPH*, pp. 81:1–81:9, 2009.
3. K. Yin, S. Coros, P. Beaudoin, and M. van de Panne, "Continuation methods for adapting simulated skills," in *SIGGRAPH*, pp. 81:1–81:7, 2008.
4. B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robot. Auton. Syst.*, vol. 57, pp. 469–483, May 2009.
5. Unity Technologies, "Unity 4." Commercial Software, 2013.
6. A. Witkin and M. Kass, "Spacetime constraints," in *SIGGRAPH*, pp. 159–168, 1988.
7. C. Rose, B. Guenter, B. Bodenheimer, and M. F. Cohen, "Efficient generation of motion transitions using spacetime constraints," in *SIGGRAPH*, pp. 147–154, 1996.
8. A. C. Fang and N. S. Pollard, "Efficient synthesis of physically valid human motion," *ACM Trans. Graph.*, vol. 22, pp. 417–426, July 2003.
9. C. K. Liu, A. Hertzmann, and Z. Popović, "Composition of complex optimal multi-character motions," in *SCA '06*, pp. 215–222, 2006.
10. L. Kovar, M. Gleicher, and F. Pighin, "Motion graphs," in *SIGGRAPH*, pp. 473–482, 2002.

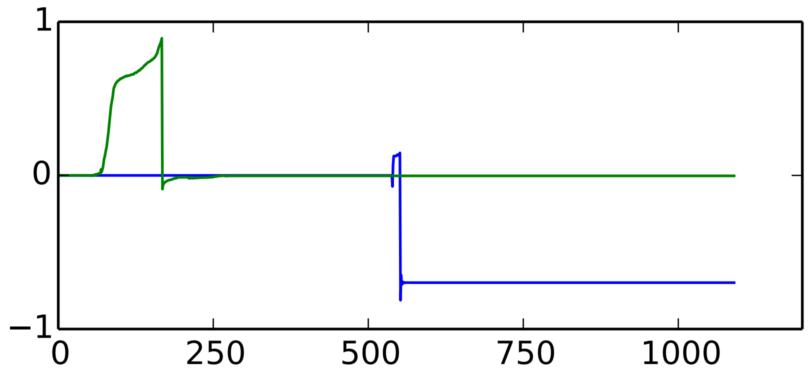


Fig. 12 We consider motion 5:13 (DANCING) with two detected periodic motion indicator functions. The motion goes through an initial balancing (frames 0–275); a skip step (276–495); two pirouettes (495–700 and 701–840) and a final pose (841–1095). The computed coordinates pick out the balancing step and one of the pirouettes.

11. O. Arikan and D. A. Forsyth, "Interactive motion generation from examples," *ACM Trans. Graph.*, vol. 21, pp. 483–490, July 2002.
12. L. Kovar and M. Gleicher, "Automated extraction and parameterization of motions in large data sets," in *SIGGRAPH*, pp. 559–568, 2004.
13. A. Safonova and J. K. Hodgins, "Construction and optimal search of interpolated motion graphs," *ACM Trans. Graph.*, vol. 26, July 2007.
14. S. Guo and J. Robergé, "A high-level control mechanism for human locomotion based on parametric frame space interpolation," in *Proceedings of the Eurographics workshop on Computer animation and simulation '96*, pp. 95–107, 1996.
15. D. J. Wiley and J. K. Hahn, "Interpolation synthesis for articulated figure motion," in *VRAIS '97*, 1997.
16. P. Beaudoin, S. Coros, M. van de Panne, and P. Poulin, "Motion-motif graphs," in *SCA '08*, pp. 117–126, 2008.
17. M. Gleicher, H. J. Shin, L. Kovar, and A. Jepsen, "Snap-together motion: assembling run-time animations," in *SIGGRAPH*, pp. 52:1–52:9, 2008.
18. K. Pullen and C. Bregler, "Animating by multi-level sampling," in *Computer Animation*, pp. 36 –42, 2000.
19. T. Mukai, "Motion rings for interactive gait synthesis," in *I3D '11*, pp. 125–132, 2011.
20. M. Peternel and A. Leonardis, "Visual learning and recognition of a probabilistic spatio-temporal model of cyclic human locomotion," in *ICPR*, vol. 4, pp. 146 — 149, aug. 2004.
21. J. Lamar-Len, E. Garca-Reyes, and R. Gonzalez-Diaz, "Human gait identification using persistent homology," in *Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*, vol. 7441, pp. 244–251, 2012.
22. R. Vasudevan, A. Ames, and R. Bajcsy, "Persistent homology for automatic determination of human-data based cost of bipedal walking," *Nonlinear Analysis: Hybrid Systems*, vol. 7, no. 1, pp. 101 – 115, 2013. IFAC World Congress 2011.
23. F. Takens, "Detecting strange attractors in turbulence," *Dynamical systems and turbulence, Warwick 1980*, pp. 366—381, 1981.
24. A. Hatcher, *Algebraic topology*. Cambridge University Press, Cambridge, 2002.
25. H. Edelsbrunner and J. Harer, *Computational Topology: an Introduction*. AMS Press, New York, 2009.
26. D. Morozov, V. de Silva, and M. Vejdemo-Johansson, "Persistent cohomology and circular coordinates," *Discrete and Computational Geometry*, vol. 45, no. 4, pp. 737—759, 2011.
27. D. Morozov, "Dionysus." `http://www.mrzv.org/software/dionysus/`, accessed November 2012, 2011.

28. E. Jones, T. Oliphant, P. Peterson, *et al.*, "SciPy: Open source scientific tools for Python." `http://www.scipy.org/`, 2001–.
29. F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine Learning in Python ," *J.M.L.R.*, vol. 12, pp. 2825–2830, 2011.
30. V. de Silva and G. Carlsson, "Topological estimation using witness complexes," in *Symposium on Point-Based Graphics*, 2004.
31. H. Edelsbrunner, D. Kirkpatrick, and R. Seidel, "On the shape of a set of points in the plane," *IEEE Transactions on Information Theory*, vol. 29, pp. 551 — 559, July 1983.
32. H. Edelsbrunner and E. P. Mücke, "Three-dimensional alpha shapes," in *Proceedings of the VVS*, pp. 75—82, 1992.
33. J. Barbic and Y. Zhao, "mocapPlayer." `http://graphics.cs.cmu.edu/software/mocapPlayer.zip`, accessed November 2012, 2012.
34. Autodesk, "Maya." Commercial software, 2013.
35. B. Fornberg, "Generation of finite difference formulas on arbitrarily spaced grids," *Mathematics of Computation*, vol. 51, no. 184, pp. 699–706, 1988.
36. J. Craig, *Introduction to robotics: mechanics and control*. Prentice Hall, Englewood Cliffs, 2004.
37. J. Lee, J. Chai, P. S. Reitsma, J. K. Hodgins, and N. S. Pollard, "Interactive control of avatars animated with human motion data," in *ACM Transactions on Graphics (TOG)*, vol. 21, p. 491500, 2002.