Eurographics/ ACM SIGGRAPH Symposium on Computer Animation (2011)
A. Bargteil and M. van de Panne (Editors)

Real-Time Classification of Dance Gestures from Skeleton Animation

Michalis Raptis[1], Darko Kirovski[2], Hugues Hoppe[2]

[1]University of California, Los Angeles
[2]Microsoft Research

Abstract

We present a real-time gesture classification system for skeletal wireframe motion. Its key components include an angular representation of the skeleton designed for recognition robustness under noisy input, a cascaded correlation-based classifier for multivariate time-series data, and a distance metric based on dynamic time-warping to evaluate the difference in motion between an acquired gesture and an oracle for the matching gesture. While the first and last tools are generic in nature and could be applied to any gesture-matching scenario, the classifier is conceived based on the assumption that the input motion adheres to a known, canonical time-base: a musical beat. On a benchmark comprising 28 gesture classes, hundreds of gesture instances recorded using the XBOX Kinect platform and performed by dozens of subjects for each gesture class, our classifier has an average accuracy of 96.9%*, for approximately 4-second skeletal motion recordings. This accuracy is remarkable given the input noise from the real-time depth sensor.*

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Animation

1. Introduction

Real-time depth sensing systems have recently sparked a revolution in the videogame industry by embracing the human body as a controller. The XBOX Kinect system, an example of such a natural user interface, parses a depth-map stream at 30 frames per second to estimate in real-time the positions of 16 predefined points that constitute a wireframe skeleton of a moving user. Subsequent algorithmic processing can then attempt to understand the user's motion in order to interactively control gameplay.

Dancing is a quintessential form of human motion and expression; hence, it deserves special focus within such a computing platform [KPS03, HGP04]. Compared to existing experiences such as Harmonix' Dance Central [Har] where a player, vis-à-vis, repeats the motion posed by an animated character in the videogame, our aim is to expand substantially the interaction that the dancer has with the avatar animation and control, by allowing him/her to dance at any time any of the pre-choreographed gestures that are modeled in system's database. To address this objective, our system learns a statistical model that captures the nuances of a pre-determined set of gesture classes, and then uses the model to classify the input skeletal motion. The system is comprised of several modules as illustrated in Fig. 1.

A novel **angular skeleton representation** is a key to overall system performance. It is used to map the skeleton motion data to a smaller set of features (each a scalar time series) that can be robustly estimated from the noisy input and yet retains the salient aspects of the motion. The aim is to reduce the overall entropy of the signal, remove dependence on camera position, and avoid unstable parameter configurations such as near gimbal lock. The approach is to fit the full torso with a single frame of reference, and to use this frame to parameterize the orientation estimates of both the first- and second-degree limb joints.

A **cascaded correlation-based max-likelihood multivariate classifier** takes into account the fact that dancing adheres to a canonical time-base, a musical beat, to simplify the template matching process. During training, our classifier builds a statistical model for each gesture class based upon both an oracle and a database of gesture instances performed by a group of subjects with a wide range of dancing

skills. At runtime, the classifier correlates the multivariate input buffer with the prototype gesture model for each class and constructs a per-class log-likelihood score. Then, it uses the scores to rank all classes and performs rounds of logistic regression tests among the top classes to identify the winning match.

A **space-time contract-expand distance metric** is the final step in our approach; it uses dynamic time-warping with exponential scaling of time-space to achieve robust comparison of the input gesture with the matched oracle. The difficulty in this task stems from the noise present in the skeletal data and the fact that humans exhibit a wide spectrum of ability to replicate a specific motion.

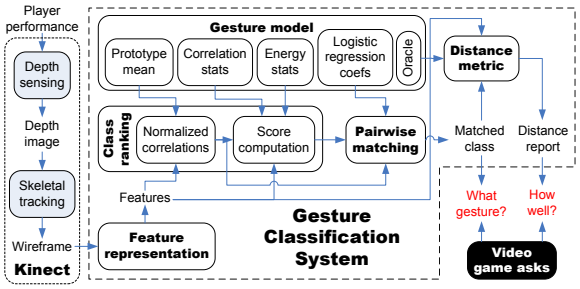

Figure 1: *Flowchart of the gesture classification system.*

We test our system on a large corpus of data acquired using the XBOX Kinect platform. It comprises 28 choreographed gesture classes, and hundreds of gesture instances performed by dozens of people for each gesture class. Our classifier attains a classification accuracy of 96.9% on average for approximately 4-second skeletal recordings. The computational complexity of the detector is linearly proportional to the number of classes in the system. For a 4-second input buffer, it consumes only 2 and 25 milliseconds on a state-of-the-art PC and a single XBOX core respectively (and the present implementation does not yet use a vectorized FFT implementation). Classification accuracy appears to scale well to a larger database. We also observe strong robustness to various types of noise including mislabeling in the training data.

Finally, we summarize our contributions by stressing that the proposed angular skeleton representation and distance metric are applicable to other gesture recognition tasks.

2. Related Work

Controlling and directing a virtual character is an important problem in computer graphics movies and computer games. Several different approaches have been proposed for the creation of realistic avatar motions that satisfy user constraints [BH00]. [AF02, KGP02] creates a graph structure to effectively search for human motions that satisfy low-level user constraints, e.g. a particular key-pose in a particular time instance. [LCR*02] organizes a motion capture database into a graph for efficient search, and generates motion using three different interfaces: a list of choices, a sketch-based interface and a video performance. The vision-based control matches silhouette features of the user with the features obtained by rendering the motion capture database at each video frame. The recent system of [IWZL09] takes an input feed from a motion capture system and translates the human intention into an action in the virtual world. Their proposed action recognition method is based on the reconstruction error of body poses in a low-dimensional space. This limits the classification ability to motions that differ significantly, such as walking versus climbing. Real-time depth-sensing systems enable us to do human motion analysis in a rather unconstrained environment. In turn, this lets us go beyond simple avatar control, and enables the use of semantic information in the interface.

Existing approaches to human action and gesture recognition can be coarsely grouped into two classes. The first uses 2D image sequences, e.g. [BD01, SLC04], while the other is based on 3D point trajectories, e.g. [CB95, KPZ*04, BSP*04, MR06]. Methods using 2D image sequences mainly focus on extracting robust and informative image features that capture the spatiotemporal characteristics of the performed action. [BD01] propose the use of global spatiotemporal templates based on the human contour to identify similar motions. Recently, bag-of-words representations of the videos [SLC04] that discard much of the spatial structure of the data have proved effective in the classification of simple human actions, e.g. “walking”, “running”, “boxing”. The use of 3D information provides many benefits over the pure image-based approaches. The data does not suffer from sensitivity to nuisance variability such as vantage point, scale, and illumination changes. Moreover, the ability to track parts of the human body robustly for the full temporal range of the human action enables the analysis of the temporal dynamics of the moving body, which encode significant discriminative information [Joh73].

Campbell and Bobick [CB95] represent the 3D joint trajectories as a collection of 2D curves in phase spaces, and performs action recognition based on the distance to each of these curves at each time instance. This does not take into account the dynamics of the activity. In [LNL05], actions are represented by spatiotemporal motion templates and multiscale template matching is performed to deal with possible temporal scale changes. However, the proposed motion templates do not account for phase alignment of the human actions. Such alignment is a crucial component of our algorithm. Moreover, the classification algorithm presented in [LNL05] does not consider the joint distribution of the motion features. The latter is important for identifying complex dancing moves. The demand for motion retrieval from large motion capture datasets for the purpose of motion synthesis has led to the adoption of efficient time series

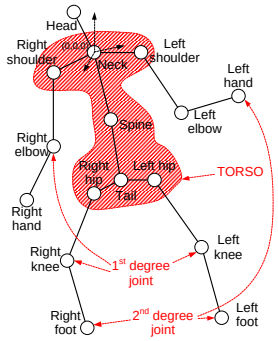

Figure 2: *Skeletal representation.*

techniques. [FRM94] introduces a subsequence-matching method that maps motion sequences into a small set of multi-dimensional rectangles in feature spaces. [KPZ*04] presents a low-complexity technique for similarity search that takes into account possible intrinsic temporal variations, based on a lower bound [Keo02] of dynamic time warping. [FF05] develops a weighted principal component analysis (PCA) based representation for human poses, mapping each human motion to a low-dimensional parametric curve, enabling efficient retrieval. Similarly, [LZWM05] introduces a indexing method that identifies a subset of discriminative feature using PCA. The distance function of [OFH08] shows the discriminative power of features based only on the kinetic energy of the motion signals. Accordingly, our recognition algorithm decomposes signals into correlation and energy features. [MRC05, MR06] transform motions into time-varying geometric feature relationships to achieve low dimensionality and robustness in the matching process. Their approach is scalable and efficient for noise-free motion capture data. Nevertheless, it is a nontrivial problem to define geometric (semantic) relationships that are discriminative and robust for highly complex human motions, e.g. dancing.

3. Skeletal Representation

The current version of the skeletal tracking algorithm (STA) [SFC*11] in the Kinect platform identifies a wireframe skeleton at a rate of 30fps for each player silhouette recognized in the sensed datastream. As illustrated in Fig. 2, the skeleton is represented by 16 points with the root of the cartesian 3D coordinate system positioned in the neck and oriented in alignment with the sensor. Each point is marked as "tracked" or "inferred" – the latter typically denoting the estimate of an occluded body part. Inferred joints are commonly plagued with substantial noise.

Our goal is to infer from this skeleton a set of features that enable effective recognition of dance gestures. These features do not require the completeness or precision necessary for re-rendering [KOF05]. One can understand the intuition behind the importance of feature construction in machine learning by noticing that the original signal may,

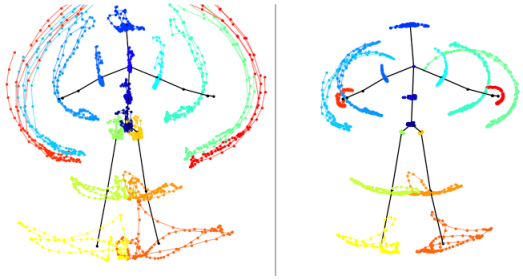

Figure 3: *Comparison of absolute (left) and relative (right) motion patterns of joints during the dance gesture* `Clap-SlideNStomp`*, as rendered in the 3D coordinate frame of the sensor.*

and typically is likely to experience loss of entropy when dimensionality-reduced to a feature vector. Once information has been lost, no classifier can recover it. Conversely, if the features are presented in a format that follows some distribution well-addressed by a group of classifiers, all will perform relatively equally well. Thus, we seek a specialized set of objectives:

- **Redundancy reduction** – express wireframe joints relative to their parent nodes, like in traditional joint-angle representations, as heuristically this should eliminate the redundancy of the resulting multivariate time-series (see Fig. 3).
- **Robustness** – overcome data errors, which are more challenging in real-time depth sensors compared to modern motion capture systems due to two factors: first, there exists strong additive noise intrinsic to the sensing system that propagates through the STA into the resulting skeleton and second, occluded parts of the skeleton are inferred and thus prone to substantial errors.
- **Invariance to sensor orientation** – maximize the invariance of the skeletal representation with respect to camera position.
- **Signal continuity and stability** – orient the coordinate axes used to compute relative positions so as to minimize the probability of signal discontinuities, e.g., gimbal lock. This objective is particularly important as we proceed to use normalized correlation for gesture detection.
- **Dimensionality reduction** – reduce the dimensionality of the search space for classification while retaining the character of the motion; compared to representations that focus on animation or motion capture, our interest lies predominantly in computing features that may not be perfectly invertible, but offer simplicity, intuitive resulting time-series, and performance.

3.1. Torso PCA frame

We observe that the points of the human torso (defined by 7 skeletal nodes as illustrated in Fig. 2) rarely exhibit strong independent motion. Thus, the torso can be treated as a ver-

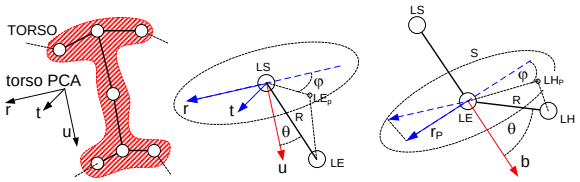

Figure 4: *Illustration of torso PCA frame, and its use in defining spherical coordinate systems for first- and second-degree joints.*

tically elongated rigid body. Yet, due to the strong noise patterns in the depth sensing system, we observe that individual torso points, in particular shoulders and hips, may exhibit unrealistic motion that we would like to limit rather than propagate by relative representation. Consequently, we aim to treat the torso as a rigid body with all of its points contributing to the estimate of its position, then use this estimate to represent the remainder of the human skeleton in relative manner.

We compute the principal components for the torso points, i.e., a 3D orthonormal basis as a result of applying PCA to the 7-by-3 torso matrix. The first principal component $\mathbf{u}$ is always aligned with the longer dimension of the torso, and we can canonically orient it (top-down) because in most dancing it is not anticipated that the player's torso will stand upside-down relative to the sensor. In contrast, for the second principal component $\mathbf{r}$, aligned with the line that connects the shoulders, the orientation is not so easily inferred, and here we must rely on the "left-right" skeleton orientation inferred by the STA. Finally, the last axis of the orthonormal basis is computed as a cross product of the first two principal components, i.e., $\mathbf{t} = \mathbf{u} \times \mathbf{r}$. We call the resulting basis $\{\mathbf{u}, \mathbf{r}, \mathbf{t}\}$ the *torso frame*.

The torso frame is well aligned with our previously stated objectives. It is an exceptionally robust and reliable foundation for a coordinate system based upon the orientation of the human body. Although it is dependent upon camera position, points represented within a coordinate system that is derived from the torso frame may be fully invariant to the sensor. It reduces 7 3D trajectories of the original problem specification to a new set of signals whose aim is to describe only the 3D orientation of the resulting orthonormal basis. We detail in Section 3.4 a set of simple features that intuitively and robustly describe torso's motion. Finally, we add a brief note that it might be possible to compute the torso frame more accurately from the underlying depth-map silhouette. Our speculation is that the computational overhead of such an approach does not offer a favorable trade-off with respect to the ensuing minor improvement in recognition performance.

3.2. First-Degree Joints

We denote all joints adjacent to the torso as first-degree joints – these include elbows, knees, and the head. We represent these points relative to the adjacent joint in the torso in a coordinate system derived from the torso frame. Consider the illustration in Fig. 4(center), where *LE* – the left elbow, is represented relative to *LS* – the left shoulder. First, we translate the torso frame, $\{\mathbf{u}, \mathbf{r}, \mathbf{t}\}$, to *LS* and construct a *spherical coordinate system* such that the origin is centered at *LS*, the zenith axis is $\mathbf{u}$, and the azimuth axis is $\mathbf{r}$. Then, *LE*'s position is described by

- its **radius** R – the distance of *LE* from the origin,
- its **inclination** θ – the angle between $\mathbf{u}$ and $\overrightarrow{(LS, LE)}$, and
- its **azimuth** φ – the angle between $\mathbf{r}$ and $\overrightarrow{(LS, LE_p)}$ where LE_p is the projection of *LE* onto the plane whose normal is $\mathbf{u}$.

Since the length of the humerus bone is normalized and constant, we ignore R for any further consideration. Thus, using this representation model, each first-degree joint is represented with two angles $\{\theta, \varphi\}$.

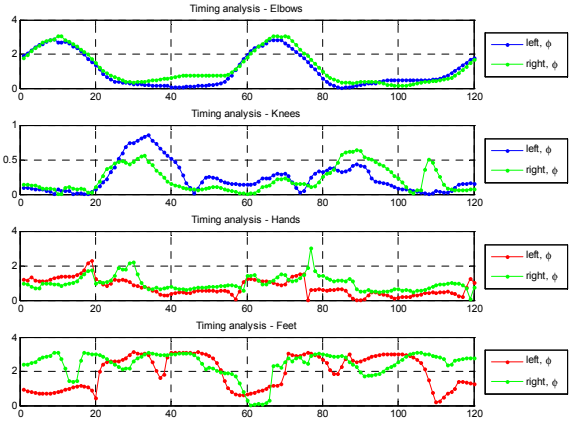

Figure 5: *Subset of feature vectors for an instance of a subject dancing* `ClapSlideNStomp`.

3.3. Second-Degree Joints

We denote as second-degree joints the tips of the wireframe extremities; these include the hands and the feet. The most descriptive vector associated with a second-degree joint is the bone that connects the adjacent first-degree joint and its adjacent torso joint. For example, as illustrated in Fig. 4(right), a vector $\mathbf{b}$ protruding out of the humerus bone is a great candidate for the zenith direction of a spherical coordinate system with an origin in the left elbow, *LE*. Let's denote the joint of the left hand as *LH*. Then, *LH*'s position is described by:

- its **radius** R – the distance of *LH* distance from the origin,
- its **inclination** θ – the angle between $\mathbf{b}$ and $\overrightarrow{(LE, LH)}$, and
- its **azimuth** φ – the angle between $\mathbf{r}_P$, the projection of $\mathbf{r}$ onto the plane S whose normal is $\mathbf{b}$, and $\overrightarrow{(LE, LH_p)}$ where LH_P is the projection of *LH* onto S.

Since the length of the forearm bone is normalized and constant, we ignore R. Thus, our model represents each second-degree joint using two angles $\{\theta, \varphi\}$. The consequences are equivalent to those of first-degree joints with one notable difference. While the inclination θ for second-degree joints is an exceptionally robust descriptor, their azimuth is not. Because the origin of the spherical coordinate system is not part of the rigid body that defines the torso frame, the orientation of $\mathbf{r}$ is dependent upon the torso's orientation and introduces noise into φ. In our experiments we have confirmed that this effect is rather mild and does not pose a challenge for the remaining steps of the classifier.

We identify an interesting problem: $\mathbf{b}$ and $\mathbf{r}$ could be oriented in such way that $\mathbf{b} \cdot \mathbf{r} = 1$, thus making the projection r_P a point. While this is unlikely to occur, any small angle between $\mathbf{b}$ and $\mathbf{r}$ is likely to pose increased levels of noise due to the instability of r_P. Although this problem could be resolved in several ways, we observed in thorough experimentation that the case $\mathbf{b} \cdot \mathbf{r} \approx 1$ seldom occurs when $\mathbf{r}$ is chosen as an azimuth reference. Note that instead of $\mathbf{r}$ we could have used $\mathbf{u}$ or $\mathbf{t}$ or any linear combination of these vectors with a wide range of impact on final performance. We selected $\mathbf{r}$ for one simple reason – its selection attenuated the issue sufficiently.

3.4. Feature Set

The novel angular wireframe model is represented by eight pairs of angles $\{\theta, \varphi\}$ for each set of the four first-degree[†] and four second-degree joints, as well as the rotation matrix of the torso frame with respect to the camera's coordinate frame.

To parameterize the rotation matrix, we use Tait-Bryan angles (i.e., yaw, pitch and roll), which is simple and intuitive. Although Euler angles are prone to gimbal lock, this problem can be avoided using quaternions, but this approach yields rather unintuitive time-series data. In practice, because the STA does not support tracking of a player who is spinning, Tait-Bryan angles can be oriented so as to rarely introduce the feared gimbal lock. This is rather important for the normalized correlation scheme in our classifier.

We denote the set of feature time-series obtained from skeletal motion as $\mathbf{f} = \{f_i(t), i = 1 \dots 19\}$ and emphasize the fact that we reduced the complexity of our input from a collection of 16 3D curves to a set of 19 1D vectors, a substantial simplification of the input specification with infrequent, negligible loss of information. Consequently, these features are geared for classification because they represent motion in relative manner that facilitates aligned, *one-dimensional* comparison.

[†] In our current implementation, we ignored the head point and thus relied on just four first-degree points.

Fig. 5 illustrates a feature set obtained by recording a subject dancing the `ClapSlideNStomp` gesture. One can observe that the torso is not too active during this gesture, the arms make two large up-down swings, each swing is preceded by a stomp, and that the dancer is moving sideways during the gesture.

In summary, our features convert 3D point trajectories to a set of time series that are robust to various types of noise common for skeletal tracking, experience exceptionally low loss of information, and are well positioned to generate "proper" distributions as input to classification tools.

4. Cascaded Classifier

The computed features of an acquired gesture are used as input to a classifier whose objective is to accurately identify which gesture class is best matched against the input. The flow of information during classification is illustrated in Fig. 1.

4.1. System Assumptions and Input/Output

A key assumption about the input to our classifier is that dancing adheres to a beat pattern – a fact rooted in the core skill and experience behind dancing. This allows us to ignore actual time and resample the input time-series so that within, say 8 beats, we create 120 frames of skeletal motion (a rate close to 30fps). This makes our classifier invariant to the pace of the beat in different musical pieces and eliminates the need to unwarp and synchronize different instances of players dancing the same gesture. Clearly, within our gaming scenario, it is assumed that each beat of music played during the game is labeled. Beat detection algorithms such as [TC02] could be used in this setting too.

We also assume that the player is allowed to dance only a limited, well-defined, and known set $\mathbb{M}$ of K moves that span over 8 beats. This way we avoid on-line learning scenarios that could be detrimental to overall error rates. Incoming frames with skeletal data are stored in a FIFO buffer. Prior to classification, the contents are resampled at a rate of 120 frames per 8 beats. The classifier finds the best matched class in $\mathbb{M}$ and finally, responds with a report that outlines how well the player danced the matched gesture.

4.2. Gesture Model

In training, we build a model of each choreographed gesture by relying on:

- a **training set**, $F_T = \{\mathbf{f}_j, j = 1 \dots L\}$ – a collection of L recordings of subjects dancing this gesture; subjects of various skill participated in the recordings, each one typically producing a handful of recordings per gesture.
- an **oracle**, $\mathbf{f}_o$ – a recording of a gesture performed by a professional dancer. This recording is considered the definition of the gesture; a single or handful of recordings

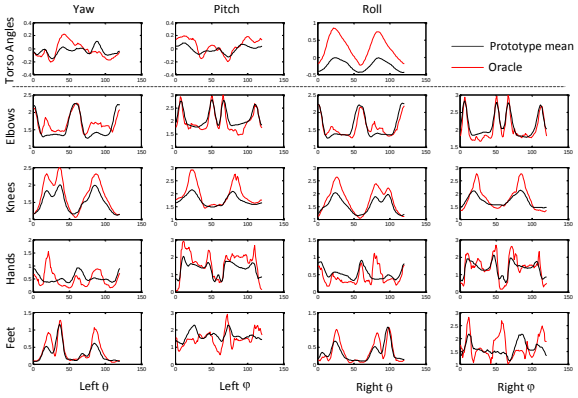

Figure 6: *Feature vectors for the derived prototype mean and the oracle that correspond to the* `ClapSlideNStomp` *gesture.*

is considered mainly because professional dancers usually repeat a specific gesture so accurately that most of the variation in the recordings stems from sensor noise.

In order to produce the expected average trajectory of a dancer for each individual feature, denoted as a **prototype mean**, we first align the training data with respect to the oracle by computing a circular normalized cross-correlation between $\mathbf{f}_o$ and each individual $\mathbf{f}_j$. Circular normalized cross-correlation c of two vectors u and v is computed as:

$$c(u,v) \equiv u \star v \equiv \frac{(u(-t)-\overline{u}) * (v(t)-\overline{v})}{||u-\overline{u}||_2 \; ||v-\overline{v}||_2}, \qquad (1)$$

where $\overline{u}$ denotes the mean of u. Circular cross-correlation of two vectors u and v can be computed as $\mathcal{F}^{-1}[\mathcal{F}(u) \cdot \mathcal{F}(R(v))]$, where $R()$ denotes reflecting the time-series vector and $\mathcal{F}$ is the discrete Fourier transform. Normalized cross-correlation is computed for each feature. In order to account for the synch of the entire body, we sum the cross-correlation vectors for all features into a single vector $\hat{c}_{j,o} = \sum_i c^i_{j,o}$. The phase offset of the two vectors equals:

$$\tau_j = \arg\max_t \hat{c}_{j,o}(t), \qquad (2)$$

thus we phase-shift all features in $\mathbf{f}_j$ for $-\tau_j$ samples in order to align the $\mathbf{f}_j$ recording with $\mathbf{f}_o$.

We define a *prototype mean* for a specific feature as $f_{m,i} = \frac{1}{L}\sum_{j=1}^{L} f_{j,i}(-\tau_j)$ and denote the gesture prototype as $\mathbf{f}_m$. The relation of $\mathbf{f}_m$ and $\mathbf{f}_o$ is that $\mathbf{f}_m$ represents the motion of an *average subject* dancing the gesture, while $\mathbf{f}_o$ is that of the expert. Typically they are similar in shape but the prototype mean is often attenuated in amplitude because skilled dancers usually emphasize movement for overall appeal. As an example, Fig. 6 illustrates $\mathbf{f}_m$ and $\mathbf{f}_o$ per feature for the `ClapSlideNStomp` gesture.

Next, we assemble a model that captures the in- and out-of-class correlation statistics. For each recording j in F_T and feature i, we compute $c^i_{j,m} = f_{m,i} \star f_{j,i}$, $\tau'_j = \arg\max_t \sum_i c^i_{j,m}(t)$ and assemble for each feature i a histogram of correlation values across $\{c^i_{j,m}(\tau'_j), j = 1 \ldots L\}$. Since L is typically small, we apply a simple kernel density estimation (KDE) filter which smoothes the histogram using a gaussian kernel [Bis06]. We store the histogram curve for a specific feature i as a lookup table, $p_i(c)$, where $-1 \le c \le 1$ is the correlation argument. For a particular feature, the lookup table thus returns the likelihood that given a correlation of the prototype mean and the input, the input gesture belongs to this specific class. Similarly, we can collect statistics on out-of-class correlations and create a corresponding lookup table $q_i(c)$. These two tables can now be combined to a scoring function for a specific correlation value. In our case, after much experimentation, we converged on: $h_i(c) = 2\log(p_i(c)) - \log(q_i(c))$. Here, we must consider the fact that skilled dancers, i.e., dancers who produce high correlations against prototype means, are typically infrequent in F_T, which may result in low $p_i(c)$ for high c. In that case, their scores are essentially penalized for their dances being "too good". To correct this anomaly, prior to applying the KDE filter, we adjust the histogram counts for high correlations.

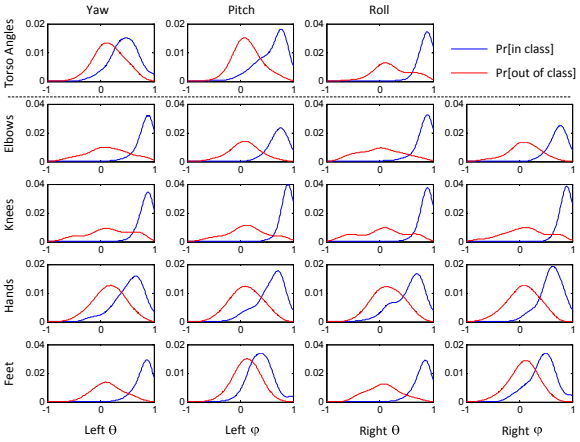

Figure 7: *Histograms p_i and q_i obtained over the training set of recordings for the* `ClapSlideNStomp` *gesture.*

As an illustration, in Fig. 7 we present the $p_i(c)$ and $q_i(c)$ curves for the `ClapSlideNStomp` gesture. One can observe which features contribute more to the "uniqueness" of this gesture based upon their Kullback-Leibler -divergence.

Normalized cross-correlation as a detection technique is effective in matching shapes [WTK87], but inefficient in matching their amplitude. Rather than using Euclidean distance or correlation without normalization, we opted for an additional distance metric, the average signal energy, as a complement to normalized correlation. Thus, for each feature f_i of an in-class gesture instance, we compute its energy-level relative to the oracle's as: $\alpha_i = ||f_{o,i}|| - ||f_i||$,

then build a histogram $e_i^+(\alpha), -4\pi^2 \leq \alpha \leq 4\pi^2$ over the energy-levels of all instances in F_T, and finally apply a KDE filter. Similar to the correlation histogram $h_i(c)$, we compute the same statistic for out-of-class instances, $e_i^-(\alpha)$, combine them as $e_i(\alpha) = 2\log(e_i^+(\alpha)) - \log(e_i^-(\alpha))$, and finally compensate $e_i(\alpha)$ for the fact that skilled dancers, who are not common in our benchmark, may have wider range of motion and thus, increased energy level of their recordings. The latter adjustment is performed by increasing the histogram counts of $e_i^+(\alpha)$ for cases of low α.

Thus, for a specific gesture and feature i our model encompasses a 3-tuple $\{f_{m,i}, h_i, e_i\}$ that consists of the prototype mean $f_{m,i}$, the correlation histogram $h_i(c)$, and the energy-level histogram $e_i(\alpha)$.

4.3. Class Ranking

The previous subsection outlined the training procedure for our classifier; in this subsection we describe the real-time classification of user motion. The input to our classifier is a stream of skeletal wireframes that we convert to feature sets. Let $\mathbf{x} = \{x_i, i = 1 \ldots 19\}$ denote the input stream of 19 features, each N samples long. For each gesture model $\mathbf{g} = \{\{f_{m,i}, h_i, e_i\}, i = 1 \ldots 19\}$ in our database we compute its score using the following steps.

- **normalized cross-correlation** – we first cross-correlate each input feature, x_i, with its corresponding prototype mean, $f_{m,i}$. This is the most computationally demanding operation of the classifier because radix-2 FFTs of length N are computed in $\mathcal{O}(N\log(N))$ operations.
- **max-likelihood score** – then we look up the corresponding histogram scores and sum them across all features:

$$s = \frac{1}{19}\sum_{i=1}^{19}\left[h_i(c_i) + e_i(\alpha_i)\right]. \tag{3}$$

- **phase-offset** – finally, we identify the phase shift τ of the input relative to the prototype mean as:

$$\tau = \arg\max_t s(t). \tag{4}$$

The phase shifts are distinct for each class.

The **classification score** for each gesture class k in the database is $s_k(\tau_k)$. We use these scores to rank all classes, with the best match having the highest score.

4.4. Pairwise Matching

The ranking classifier can still be improved because some classes are often similar in motion to the point where their prototype means across all features are equivalent except for one. One can view all instances of two gesture classes as a collection of points in a locality of a large $2(19+19)$-dimensional space. Due to acquisition noise and the variety of ways in which humans can play a certain gesture, two classes whose prototype means are nearly identical (across all but very few features) may have intersecting volumes if, for example, a multidimensional sphere is used to contain and detect all points of a specific class. Since the disambiguation of the two classes is more nuanced and selectively dependent upon features, there exists need to better distinguish neighboring classes using an advanced, pairwise matching tool.

Weighting of likelihoods in Eqn. 3 is one way to improve the classification agility. It is important to stress that the "optimal" weights need to be recomputed and are likely distinct for each pairwise comparison of gesture matches. Thus, we opt to compute these weights using standard logistic regression [Bis06] and deploy the trained coefficients at classification as follows:

logistic regression – for the two top-tiered classes with highest $s(\tau)$ scores, say indexed k_1 and k_2, we perform a binary classification by computing:

$$\begin{aligned} &Pr(C = k_1 \mid \mathbf{x}) = 1/\left(1 + e^{-\gamma}\right), \\ &\gamma = \sum_{i=1}^{19} w_{h,i}^{(k_1,k_2)} h_{k_1,i}(c_{k_1,i}) + w_{e,i}^{(k_1,k_2)} e_{k_1,i}(\alpha_{k_1,i}) + \\ &\qquad \sum_{i=1}^{19} w_{h,i}^{(k_2,k_1)} h_{k_2,i}(c_{k_2,i}) + w_{e,i}^{(k_2,k_1)} e_{k_2,i}(\alpha_{k_2,i})), \end{aligned} \tag{5}$$

where all weights have been trained using logistic regression. In case $Pr(C = k_1 \mid \mathbf{x}) \geq 0.5$, class k_1 would be denoted as the best match, otherwise k_2. The process of pairwise matching the "winner class" with the next "runner-up class" could be repeated recursively, although the likelihood that a class deep on the $s(\tau)$-list "wins" rapidly declines. Thus, in our experiments we use only a 3-deep sequence of pairwise class-comparisons via logistic regression.

Therefore, we augment our gesture model $\{f_{m,i}, h_i, e_i\}$ with another data field, the coefficient matrix for logistic regression $W = \{\{w_{h,i}^{(k_r,k_q)}, w_{e,i}^{(k_r,k_q)}, w_{h,i}^{(k_q,k_r)}, w_{e,i}^{(k_q,k_r)}\} \mid i = 1 \ldots 19, r = 1 \ldots K, q = 1 \ldots K\}$, where K is the number of gesture classes. Since the size of assets required for classification is proportional to $\mathcal{O}(K^2)$, for large K the size of the classification database would grow prohibitively. Here we note that for most gesture classes the differences among them are large enough that the scoring function in Eqn. 3 is sufficient to disambiguate them; in training we identify "similar" gestures and train weighting matrices only for these sparse pairs. Of course, the density of required pairs in the complete $K \times K$ matrix depends on the similarity of the gesture motion classes.

Finally, we point to some interesting features of our classifier:

- The length of the input buffer does not have to equal the length of the class prototypes. Thus, shorter input sequences can be matched using the same algorithm. Only the normalization parameters of the cross correlation need to be adapted.

- Our algorithm returns as a side-effect the phase shift with respect to the prototype of the matched class. This information is useful to synchronize the user's dancing pattern with the gaming platform.
- Errors reported by our system may often be benign, in particular for short input buffers. One characteristic of our classifier is that it returns in "near optimal" manner the best-matched class within the entire gesture database, as well as phase-shift within its prototype mean. Therefore, in scenarios where an avatar renders the player's motion, errors may pass unnoticed due to short-spanned cross-class similarities.

Moreover, the choice of the logistic regression with L2 regularization was made due to its simple training and evaluation procedures. Note that it is intimately related to linear SVMs and Ordinary Least Squares and on our dataset, many versions of these classifiers are expected to perform similarly.

4.5. Distance Metric

Once our classification method identifies the best matched class, the remaining question is how "well" the player performed this gesture compared to the oracle. Comparison with respect to the prototype mean (including the score obtained by correlation with it) is misleading as it outlines how well the player performed versus the average rather than the expert dancer. On the other hand, besides having a single scoring number, we favor a report that outlines how "well" the game player danced per joint. To resolve this problem, it is desired to obtain motion recordings labeled for artistic appeal, and to learn a regression model on this dataset that replicates the human expert. Even then, it is arguable how consistent human labeling is. To avoid the semantic nature of grading body motion, we measure the discrepancy between the relative motion of the current actor and the expert.

In our approach, we first globally align the feature sequence of the player using the phase-shift provided by the classification method. Subsequently, dynamic time warping is used to measure the discrepancy of the two signals considering the possible local misalignments. To overcome the outliers due to the noise we prefer a robust cost at the computation of dynamic time warping, defined as:

$$d(x,y) = \begin{cases} 1 - e^{-(x-y)^4/\sigma^2}, & \text{if } |x-y| < \delta \\ 1 - e^{-\delta^4/\sigma^2}, & \text{otherwise} \end{cases} \quad (6)$$

where σ is a parameter that controls the amount of deviation from the expert's performance allowed and δ is a threshold minimizing the effect of outliers.

The proposed metric is parameterized to adjust to different motion accuracy standards along space and time by tightening and relaxing σ and δ. In our experiments, it behaved as a rather solid detector when computed against all oracles; still, its computational complexity was prohibitive for exploring per-class applications.

5. Experimental Results

Benchmark. The proposed technology has been tested against a large benchmark of 28 different gesture classes with a total of 11361 recordings. The cardinality of the training set is outlined in Table 2 with a majority of classes having more than 200 instances performed by at least a dozen and as high as 50 different subjects. All recordings were produced using Kinect's STA at 30fps, to a variety of music clips and under different environmental settings. We linearly interpolated and normalized all recordings to 120 samples per 8 beats.

For each class, we also have an oracle recording; the skill level of all other participating subjects varied greatly. The complexity of the dance movements present in our dataset makes it appealing as the backbone of an interface for a videogame. In the same time, it is a challenging dataset for a classification algorithm, since many moves share "primitive" actions. Therefore, the temporal dynamics and the motion of small portion of the body are the only discriminative characteristics to achieve satisfactory classification performance.

Moreover, due to the sheer amount of data, the recordings were not labeled accurately, with gestures such as `TouchOut` and `StepOut` having a mislabeling rate of approximately 10%. In that respect, we ask the reader to treat the obtained experimental results as likely lower bounds on the results that could be achieved in the absence of mislabeled data.

Performance. We evaluate the proposed scheme on our dataset using 4-fold cross validation and report the average performance in Table 2. At each of the folds, the dataset was partitioned by randomly selecting dancers-subjects to approximately 70% training and 30% testing. Note that correlation of gestures coming from a single dancer is much higher than against any other dancer unless the ones considered are professionals. The classification performance for the entire 8 beat sequences ($N = 120$ samples) is on average 96.9%. The lowest performances observed are 81.1% and 89.4% for the `TouchOut` and `StepOut` gestures, respectively. As mentioned above, the main reason for those performances is the mislabeling occurred during their annotation. Also note here that, the classification accuracy for dance moves that contains body poses, which are extremely challenging for the Kinect's STA to identify (e.g., `KneeBend`) is relatively lower due to the lower signal-to-noise ratio. Moreover, we evaluate the performance of the different components of our algorithm. Specifically, the classification performance is computed in the case of (a) using only the normalized cross-correlation features ($h_i(c_i)$, Eqn. 3), (b) using only the energy features ($e_i(a_i)$, Eqn. 3), (c) using only the first stage of the cascade classifier (class ranking, Sect. 4.3)

and (d) using both two stages of classifier (logistic regression, Sect. 4.4). The results are summarized in Table 1.

Table 1: *Performance comparison of different components of the proposed method.*

Features	Class Ranking	Logistic Regression
Only Energy: $e_i(a_i)$	33.3%	57.6%
Only Correlation: $h_i(c_i)$	71.85%	89.9%
Energy + Correlation	95%	**96.9%**

We also tested our algorithm for shorter input sequences 2 and 4 beats, corresponding to 30 and 60 samples respectively. For the training of our algorithm we randomly sampled sub sequences of length 2 and 4 beats from the full 8 beat sequences of our dataset. All the sub sequences where zero-padded to 8 beats length. The correlation distributions and the coefficients of the logistic regression were learned commonly for all the 2 and 4 beats subsequences, as described in Sect. 4. Interesting, the decrease of performance for classifying 4 beats sub sequences is on average only 0.8% compared to the 8 beats classifier performance. The ability of our scheme to accurately identify the human behavior among 28 highly complex dancing moves requiring only 2 seconds buffer (assuming that 1 music beat is 0.5 seconds) is of great importance for the usability of the propose interface. For the sub sequence of 2 beats length the average performance is 86.5%, actions that are composed by the same elementary moves are confused, e.g. `ClapSlideNStomp` and `WhassupWalk`.

Additionally, we developed a detector that considers shorter input buffers but with a known phase shift with respect to the prototype mean (Table 2). In this case, a search window of 11 frames was considered to identify the best max-likelihood score. The superior performance for the 4 beat sub sequences verifies the importance of the synchronization among the players with the prototype mean. Assuming an approximate knowledge of the phase alignment, the search for the best τ can be restricted to a smaller range making the correlation statistics more discriminative.

Finally, we compare our algorithm with two efficient baseline methods for classification and retrieval of motion capture data [OFH08, SH08]. Onuma et al. [OFH08] proposed logarithmic scaled kinetic energy features and used a 1-nearest neighbor classifier based on their Euclidean distance. In order to implement the proposed method, we transformed the 3-D positions of skeletal joints of our dataset to Euler angle representation. The proposed distance function has been proven to be effective for classification of motion capture data performing "coarse" motions, e.g. "walking", "running", "boxing". However, its average classification accuracy on our dataset is relatively low 13.76%, since the dance gestures of our dataset have subtle differences on their energy profiles. [SH08] developed a performance animation system that uses accelerometer measurements as input. Using the Haar wavelet transform on the acceleration signals they manage to obtain a real-time motion retrieval algorithm. The average classification accuracy obtained using the latter algorithm is 63.32%.

Table 2: *Experimental results. L is the number of gesture recordings per class. The columns under 4 × "70/30 split" illustrate the average classification accuracy using 4-fold cross validation. The columns under "Synched" show the average classification accuracy assuming an approximate knowledge of the phase alignment.*

Class	L	4 × "70/30 split"			Synched	
		8 beat	4 beat	2 beat	4 beat	2 beat
`2Step`	714	98.2%	94.5%	91.0%	99.8%	96.2%
`ArmyWalk`	181	94.7%	92.9%	82.0%	99.9%	82.8%
`BodyRoll`	334	99.2%	95.0%	92.0%	95.5%	94.6%
`Bounce`	200	95.5%	97.5%	90.0%	98.4%	92.9%
`ClapSlideNStomp`	241	99.9%	99.7%	76.0%	100.0%	91.6%
`FunkyOldMan`	391	99.8%	98.3%	91.0%	99.1%	95.0%
`Hip`	259	99.2%	98.3%	93.9%	99.6%	94.2%
`KickSnap`	286	99.2%	96.3%	93.2%	100.0%	99.3%
`KneeBend`	186	95.0%	82.5%	78.0%	95.5%	86.3%
`LockinNew`	231	95.8%	86.8%	81.0%	100.0%	98.8%
`MJHips`	324	100.0%	95.0%	90.7%	100.0%	98.9%
`OpenClose`	494	98.2%	98.3%	84.0%	99.4%	98.4%
`PointSwim`	228	97.9%	97.7%	94.1%	99.6%	92.3%
`Roar`	379	95.9%	94.5%	81.3%	99.5%	97.6%
`RockOn`	448	99.8%	99.6%	91.1%	99.6%	94.0%
`SaluteStep`	188	100%	100%	94.7%	99.8%	96.9%
`SideStep`	626	96.6%	90.0%	87.2%	99.2%	97.0%
`SideStepHand`	417	97.1%	97.5%	89.5%	97.7%	94.0%
`Skater`	257	96.9%	97.1%	74.4%	95.2%	83.2%
`SlideNStomp`	257	99.7%	99.0%	85.8%	100.0%	94.9%
`Step80`	492	95.8%	91.6%	84.1%	98.4%	95.4%
`StepOut`	847	89.4%	84.6%	95.4%	98.1%	92.1%
`StepTouch`	293	96.3%	92.9%	79.7%	97.7%	93.1%
`StepTouchSnap`	348	97.8%	97.0%	89.1%	99.7%	98.2%
`TheWave`	1318	99.2%	94.6%	91.9%	97.1%	89.1%
`TopRock`	425	97.1%	93.5%	92.1%	97.9%	95.5%
`TouchOut`	787	81.5%	81.1%	71.1%	97.2%	90.7%
`WhassupWalk`	496	97.7%	95.1%	76.4%	98.5%	82.3%
Average	—	**96.9%**	94.2%	86.5%	98.6%	93.4%

6. Conclusion

The emergence of affordable real-time depth sensors opens new avenues for interactivity. Compared to accurate optical or mechanical marker-based motion capture systems, depth sensors offer balance in usability and cost. The consequence is substantial increase in noise, in particular when we are faced with the fact that the world population is the user. The noise stems from the sensor, player's physique, clothing, and kinetic IQ; the latter probably causing the most variability in particular in complicated motion patterns.

We have shown that in the context of dancing, a classifier can be designed and trained to recognize from among several dozen gestures in real-time and with high accuracy. The periodic nature of dance moves allowed us to specialize the classifier to efficiently detect and compensate for phase alignment. Scaling the classifier for K to be in the hundreds affects two system characteristics:

- **Computational workload** increases linearly with K. At runtime, the critical path in the system revolves around skeleton acquisition, gesture classification, and the system rendering player's avatar on screen. Since sensor's noise

is substantial, rendering based on its output is highly undesirable, gesture classification with an intuitive distance metric is crucial in providing an immersive experience.

- **Error rates** in the system do not scale according to the Central Limit Theorem because the discriminatory noise is not i.i.d., it is well under the control of the game designer. To address this challenge, our system is designed to help the choreographer identify how discriminative a new gesture is, compared to an existing gesture database.

There are many areas for future work. Certainly, one could explore relaxing some of the assumptions of our classifier, such as the musical beat alignment, integrating raw depth sensor data to improve classification, providing user feedback as to quality of their motion to the point where the system could infer a gesture oracle from motion patterns of users with average kinetic IQ, etc. In addition to identifying gestures, it may be possible to learn interesting axes within the subspace of each gesture. Such axes might correspond to attributes such as emotion or energy. As an analogy, previous work in computer graphics has shown how to synthesize animation using "verbs and adverbs" [RCB02]. The goal of the reverse process would be to recognize not only the verb (gesture) but also its adverbs (attributes).

Acknowledgements. We would like to thank Joel Pritchet, Chuck Noble, Tom Fuller, Wei-ge Chen, Darren Gehring, Ivan Tashev and iNIS for valuable discussions, for their help collecting the data and implementing the algorithm on the Xbox console.

References

[AF02] ARIKAN O., FORSYTH D. A.: Interactive motion generation from examples. *ACM Trans. Graph.* (2002). 2

[BD01] BOBICK A., DAVIS J.: The recognition of human movement using temporal templates. *IEEE Trans. on Pattern Anal. and Machine Intell* (2001). 2

[BH00] BRAND M., HERTZMANN A.: Style machines. In *Proc. of the conf. on Computer graphics and interactive techniques* (2000), SIGGRAPH '00. 2

[Bis06] BISHOP C.: *Pattern recognition and machine learning*, vol. 4. Springer New York, 2006. 6, 7

[BSP*04] BARBIČ J., SAFONOVA A., PAN J.-Y., FALOUTSOS C., HODGINS J. K., POLLARD N. S.: Segmenting motion capture data into distinct behaviors. In *Proc. of Graphics Interface* (2004). 2

[CB95] CAMPBELL L., BOBICK A.: Recognition of human body motion using phase space constraints. In *Proc. of Intl. Conf. of Computer Vision* (1995). 2

[FF05] FORBES K., FIUME E.: An efficient search algorithm for motion data using weighted pca. In *Proc. of the ACM SIGGRAPH/Eurographics Symposium on Computer animation* (2005). 3

[FRM94] FALOUTSOS C., RANGANATHAN M., MANOLOPOULOS Y.: Fast subsequence matching in time-series databases. In *Proc. of the ACM SIGMOD* (1994). 3

[Har] HARMONIX MUSIC SYSTEMS: www.dancecentral.com. 1

[HGP04] HSU E., GENTRY S., POPOVIĆ J.: Example-based control of human motion. In *Proc. of the ACM SIGGRAPH/Eurographics symposium on Computer animation* (2004). 1

[IWZL09] ISHIGAKI S., WHITE T., ZORDAN V. B., LIU C. K.: Performance-based control interface for character animation. *ACM Trans. Graph.* (2009). 2

[Joh73] JOHANSSON G.: Visual perception of biological motion and a model for its analysis. *Perceiving events and objects* (1973). 2

[Keo02] KEOGH E.: Exact indexing of dynamic time warping. In *Proc of the Intl. conf. on VLDB* (2002). 3

[KGP02] KOVAR L., GLEICHER M., PIGHIN F.: Motion graphs. *ACM Trans. Graph.* (2002). 2

[KOF05] KIRK A., O'BRIEN J., FORSYTH D.: Skeletal parameter estimation from optical motion capture data. In *Proc. of Conf. Computer Vision and Pattern Recognition* (2005). 3

[KPS03] KIM T., PARK S., SHIN S.: Rhythmic-motion synthesis based on motion-beat analysis. *ACM Trans. Graph.* (2003). 1

[KPZ*04] KEOGH E., PALPANAS T., ZORDAN V., GUNOPULOS D., CARDLE M.: Indexing large human-motion databases. In *Proc. of the Intl. Conf. on VLDB* (2004). 2, 3

[LCR*02] LEE J., CHAI J., REITSMA P., HODGINS J., POLLARD N.: Interactive control of avatars animated with human motion data. *ACM Trans. Graph.* (2002). 2

[LNL05] LV F., NEVATIA R., LEE M. W.: 3D human action recognition using spatio-temporal motion templates. In *Computer Vision in Human-Computer Interaction* (2005). 2

[LZWM05] LIU G., ZHANG J., WANG W., MCMILLAN L.: A system for analyzing and indexing human-motion databases. In *Proc. of the ACM SIGMOD* (2005). 3

[MR06] MÜLLER M., RÖDER T.: Motion templates for automatic classification and retrieval of motion capture data. In *Proc. of the ACM SIGGRAPH/Eurographics Symposium on Computer animation* (2006). 2, 3

[MRC05] MÜLLER M., RÖDER T., CLAUSEN M.: Efficient content-based retrieval of motion capture data. In *ACM SIGGRAPH* (2005). 3

[OFH08] ONUMA K., FALOUTSOS C., HODGINS J.: FMDistance: A fast and effective distance function for motion capture data. *Short Papers Proc. of EUROGRAPHICS* (2008). 3, 9

[RCB02] ROSE C., COHEN M. F., BODENHEIMER B.: Verbs and adverbs: Multidimensional motion interpolation. *IEEE Computer Graphics and Applications* (2002). 10

[SFC*11] SHOTTON J., FITZGIBBON A., COOK M., SHARP T., FINOCCHIO M., MOORE R., KIPMAN A., BLAKE A.: Real-time human pose recognition in parts from single depth images. In *Proc. Conf. Computer Vision and Pattern Recognition* (2011). 3

[SH08] SLYPER R., HODGINS J.: Action capture with accelerometers. In *Proc. of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (2008). 9

[SLC04] SCHULDT C., LAPTEV I., CAPUTO B.: Recognizing human actions: A local SVM approach. In *Proc. Intl. Conf. on Pattern Recognition* (2004). 2

[TC02] TZANETAKIS G., COOK P.: Musical genre classification of audio signals. *IEEE Trans. on Speech and Audio Processing* (2002). 5

[WTK87] WITKIN A., TERZOPOULOS D., KASS M.: Signal matching through scale space. *Intl. Journal of Computer Vision* (1987). 6