An Information Retrieval System for Motion Capture Data

Bastian Demuth[1], Tido Röder[1], Meinard Müller[1], and Bernhard Eberhardt[2]

[1] Universität Bonn, Institut für Informatik III
Römerstr. 164, 53117 Bonn, Germany
{demuth, roedert, meinard}@cs.uni-bonn.de
[2] Hochschule der Medien, Fachhochschule Stuttgart
Nobelstr. 10, 70569 Stuttgart, Germany
eberhardt@hdm-stuttgart.de

Abstract. Motion capturing has become an important tool in fields such as sports sciences, biometrics, and particularly in computer animation, where large collections of motion material are accumulated in the production process. In order to fully exploit motion databases for reuse and for the synthesis of new motions, one needs efficient retrieval and browsing methods to identify similar motions. So far, only ad-hoc methods for content-based motion retrieval have been proposed, which lack efficiency and rely on quantitative, numerical similarity measures, making it difficult to identify logically related motions. We propose an efficient motion retrieval system based on the query-by-example paradigm, which employs qualitative, geometric similarity measures. This allows for intuitive and interactive browsing in a purely content-based fashion without relying on textual annotations. We have incorporated this technology in a novel user interface facilitating query formulation as well as visualization and ranking of search results.

1 Introduction

In the past two decades, motion capture systems have been developed that allow to track and record human motions at high spatial and temporal resolutions. The resulting motion capture data, typically consisting of 3D trajectories of markers attached to a live actor's body, is used to analyze human motions in fields such as sports sciences and biometrics (person identification), and to synthesize realistic motion sequences in data-driven computer animation. Even though there is a rapidly growing corpus of motion data, there still is a lack of efficient motion retrieval systems that allow to identify and extract user-specified motions.

Previous retrieval systems often require manually generated textual annotations, which roughly describe the motions in words. Since the manual generation of reliable and descriptive labels is infeasible for large data sets, one needs efficient *content-based* retrieval methods such as techniques based on the query-by-example paradigm. The crucial point in such an approach is the notion of *similarity* used to compare the query with the documents to be searched. For

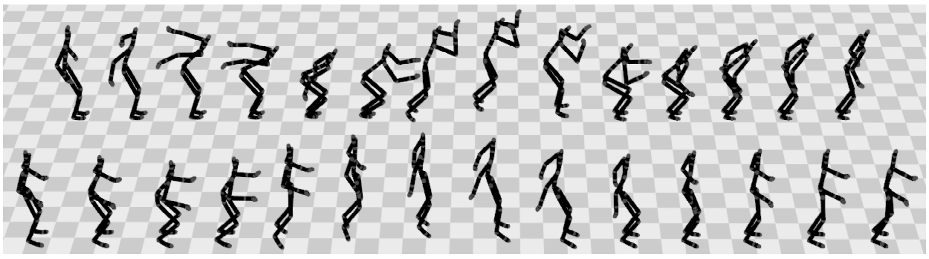


Fig. 1. Top: 14 poses from a forceful jump. Bottom: 14 poses from a weak jump.

Input:	Short query motion clip.
	Feature selection.
	Fault tolerance settings.
Procedure:	Automatic conversion of query motion into a sequence of geometric configurations (with respect to the selected features).
	Index-based retrieval, post-processing, and ranking.
Output:	Ranked list of hits.

Fig. 2. Overview of the retrieval process based on the query-by-example paradigm.

the motion scenario, two motions may be regarded as similar if they represent variations of the same action or sequence of actions, see [4]. These variations may concern the spatial as well as the temporal domain. For example, the jumps shown in Fig. 1 describe the same kind of motion even though they differ considerably with respect to timing, intensity, and execution style (note, e.g., the arm swing). In other words, *logically similar* motions need not be *numerically similar*, as is also pointed out in [4].

In this paper, we present a motion retrieval system that allows for efficient retrieval of *logically* related motions based on the query-by-example paradigm, see Fig. 2 for an overview. As opposed to previous approaches that are based on *quantitative*, *numerical features*, our approach is based on *qualitative*, *relational features*, which describe certain geometric constellations between specified points of the body. As will be described in Sect. 2, such relational features are not only robust to spatio-temporal variations (thus providing a basis for the identification of logically related motions) but are also ideally suited for indexing (speeding up the retrieval process considerably). The front end of our retrieval system consists of a graphical user interface facilitating intuitive query formulation as well as visualization and ranking of search results, see Sect. 3. Finally, we report on some of our experiments in Sect. 4. For more experimental details and result videos, we refer to `http://www-mmdb.iai.uni-bonn.de/projects/mocap/RetrievalGUI.html`.

We close this section with a discussion of related work. So far, only little work has been published on motion capture indexing and retrieval based on the query-by-example paradigm. To account for spatio-temporal variations, most previous approaches rely on the technique of *dynamic time warping* (DTW), see [2, 4, 8]. However, major drawbacks to DTW are its quadratic running time and storage requirements, making DTW infeasible for large data sets. To speed up similarity search, Keogh et al. [3] use an index structure based on bounding envelopes, allowing to identify similar motion fragments that differ by some uniform scaling factor with respect to the time axis. In comparing individual frames of the data streams, all of these approaches rely on *numerical* cost measures. As a first step towards logical similarity, Liu et al. [5] compare motions based on a so-called cluster transition signature that is immune to temporal variations. To bridge the semantic gap between logical similarity as perceived by humans and computable numerical similarity measures, Müller et al. [6] introduce a new type of qualitative geometric features and induced motion segmentations, yielding spatio-temporal invariance as needed to compare logically similar motions.

The main contribution of the present paper is to extend the concepts from [6] in the following ways: we define new classes of generic relational features grasping important aspects of an actor's motion patterns, which prove to be well-suited for content-based motion retrieval. A novel Feature Design GUI allows to instantiate the generic features into semantically meaningful feature sets, supported by suitable optimization and visualization tools. As a further contribution, we present a Retrieval GUI based on the query-by-example paradigm that includes a novel ranking strategy and facilitates merging operations on hits.

2 Relational Motion Features

The most common recording technology for motion capture data uses an array of digital cameras to three-dimensionally track reflective markers attached to a live actor's body, see, e.g. [7]. The tracking data can then be post-processed to obtain a multi-stream of 3D trajectories corresponding to the joints of a fixed skeletal *kinematic chain* as indicated by Fig. 3. A full set of 3D coordinates describing the joint positions of a kinematic chain for a fixed point in time is also referred to as a *pose*. A motion capture data stream is thought of as a sequence of poses or *frames*, typically sampled at 30–600 Hz.

In a sense, motion capture data has a much richer semantic content than, for example, pure video data of a motion, since the position and the meaning of all joints is known for every pose. This fact can be exploited by considering *relational features* that describe (boolean) geometric relations between specified points of a pose or short sequences of poses, see [6]. We explain the main idea of such features by means of some typical examples. Consider the test whether the right foot lies in front of (feature value one) or behind (feature value zero) the plane spanned by the center of the hip (the root), the left hip joint, and the left foot, cf. Fig. 3 (a). Interchanging left with right, one obtains the analogous feature for the other leg. A combination of these features is useful to identify locomotion such as walking or running. A similar feature is defined by the oriented plane fixed at the left and right shoulders and the root, shifted one humerus length to the front: checking whether the left hand is in front of or behind this plane, one

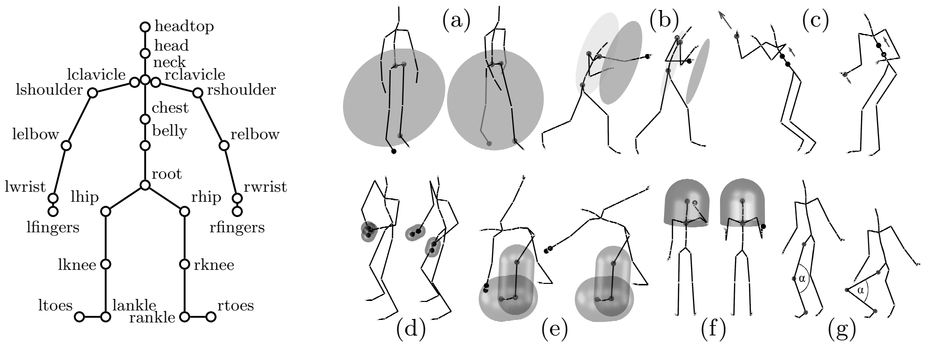


Fig. 3. Left: skeletal kinematic chain model consisting of rigid *body segments* flexibly connected by *joints*, which are highlighted by circular markers and labeled with joint names. Right: qualitative features describing geometric and kinematic relations between the body points of a pose that are indicated by thickened markers.

obtains a feature suitable to identify left punches, see Fig. 3 (b). The feature indicated by Fig. 3 (c) checks whether the right hand is moving into the direction determined by the belly-chest segment. Other types of relational features express whether specified points of the body are close together, yielding, for example, "touch" detectors for the two hands, a hand and a leg, or a hand and the head (Fig. 3 (d)–(f)). We also use relational features to check if certain parts of the body such as the arms or the legs are bent or stretched (Fig. 3 (g)).

By looking at maximal runs of consecutive frames yielding the same feature values for a fixed combination of frame-based relational features, one obtains a temporal segmentation of motion data streams, see Fig. 4. In order to compare two different motions, each motion data stream is coarsened by transforming it into a segment-based progression of feature values. Two motion clips are then considered as similar if they possess (more or less) the same progression of feature values. For further details, we refer to [6]. The main point is that relational features are invariant under global orientation and position, the size of the skeleton, and local spatial deformations, whereas the induced segmentation introduces robustness to temporal variations.

2.1 Feature Design

To facilitate retrieval in large motion databases containing a great variety of motions, it is essential to provide the end-user of our retrieval system with a semantically rich set of features covering different body parts and various aspects of motions. Of course, the requirements to such a feature set will heavily depend on the intended application. The goal of our retrieval system is to search for motion clips in view of their rough course of motion. In this paper, we describe an exemplary system based on a set of 22 features, see Table 1.

In constructing such a feature set, we started with a small set of *generic* boolean features, which encode certain joint constellations in 3D space and time.

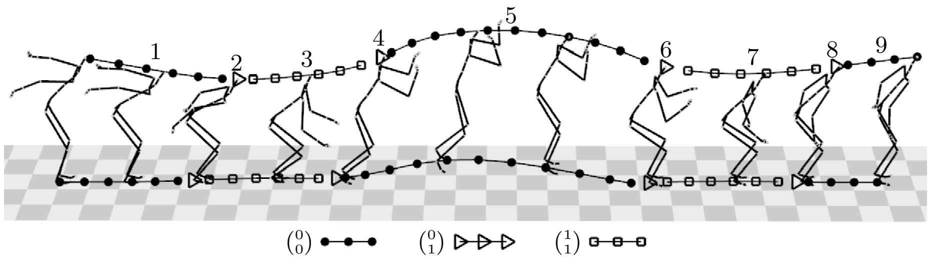


Fig. 4. Segmentation of a parallel leg jumping motion with respect to a combination of the feature "left knee bent" and the feature "right knee bent". Here the nine segments correspond to the sequence $\left(\binom{0}{0},\binom{0}{1},\binom{1}{1},\binom{0}{1},\binom{0}{0},\binom{0}{1},\binom{1}{1},\binom{0}{1},\binom{0}{0}\right)$ of feature values. Poses assuming the same feature values are indicated by identically marked trajectory segments. The trajectories of the joints 'headtop' and 'rankle' are shown.

gen. feature	**ID**	**set**	j_1	j_2	j_3	j_4	α	**description**
$F^{(j_1,j_2,j_3;j_4)}_{\alpha,\text{plane}}$	F_1	ℓ	root	lhip	ltoes	rankle	0.3 hl	foot in front
$F^{(j_1,j_2,j_3;j_4)}_{\alpha,\text{nplane}}$	F_3	ℓ	lhip	rhip	rhip	rankle	0.5 hl	leg sideways
$F^{(j_1,j_2,j_3)}_{\alpha,\text{bent}}$	F_5	ℓ	rhip	rknee	rankle		110°	knee bent
	F_7	u	rshoulder	relbow	rwrist		110°	elbow bent
$F^{(j_1)}_{\alpha,\text{fast}}$	F_9	ℓ	root	rankle			6 hl/s	foot fast
	F_{11}	u	rshoulder	rwrist			6 hl/s	hand fast
$F^{(j_1,j_2;j_3)}_{\alpha,\text{move}}$	F_{13}	ℓ	root	belly	rankle		2 hl/s	foot moves up
	F_{15}	u	belly	chest	rwrist		3 hl/s	hand moves up
	F_{17}	u	chest	belly	rwrist		3 hl/s	hand moves down
	F_{19}	u	neck	rshoulder	rwrist		3 hl/s	hand moves sideways
$F^{(j_1,j_2,j_3;j_4)}_{\alpha,\text{nmove}}$	F_{21}	u	root	lshoulder	rshoulder	rwrist	3 hl/s	hand moves forward

Table 1. The feature set used in our experiments. All described features pertain to the right half of the body. Analogous versions for the left body parts exist and are assigned the even IDs $F_2, F_4, \dots F_{22}$. The abbreviation "hl" denotes the relative length unit "humerus length", which is used to handle differences in absolute skeleton sizes.

In particular, we used the following generic features:

$$F^{(j_1,j_2,j_3;j_4)}_{\alpha,\text{plane}},\ F^{(j_1,j_2,j_3;j_4)}_{\alpha,\text{nplane}},\ F^{(j_1;j_2)}_{\alpha,\text{touch}},\ F^{(j_1,j_2,j_3)}_{\alpha,\text{bent}},\ F^{(j_1;j_2)}_{\alpha,\text{fast}},\ F^{(j_1,j_2;j_3)}_{\alpha,\text{move}},\ F^{(j_1,j_2,j_3;j_4)}_{\alpha,\text{nmove}}.$$

Each of these features assumes either the value one or the value zero and depends on a set of joints, denoted by $j_1, j_2, \ldots$, and on a threshold value $\alpha \in \mathbb{R}$. The first generic feature assumes the value one iff joint j_4 has a signed distance greater than α from the oriented plane spanned by the joints j_1, j_2 and j_3. For example, setting j_1 ='root', j_2 ='rhip', j_3 ='rankle', j_4 ='lankle', and $\alpha = 0.3$, one obtains the feature of Fig. 3 (a). The same test is described by $F^{(j_1,j_2,j_3;j_4)}_{\alpha,\text{nplane}}$, but here we define the plane in terms of a normal vector (given by j_1 and j_2), and fix it at j_3. The generic feature $F^{(j_1;j_2)}_{\alpha,\text{touch}}$ assumes the value one iff the two joints j_1 and j_2 are closer than α. Similar generic touch detectors can also be defined for two body segments, or a joint and a body segment, see Fig. 3 (d)–(f). The generic feature $F^{(j_1,j_2,j_3)}_{\alpha,\text{bent}}$ assumes the value one iff the angle between the segments determined by (j_1, j_2) and (j_2, j_3) is below the threshold α. For example in Fig. 3 (g), we set j_1 ='lhip', j_2 ='lknee', j_3 ='lankle', and $\alpha = 110°$. The generic feature $F^{(j_1,j_2)}_{\alpha,\text{fast}}$

assumes the value one iff joint j_2 has a relative velocity with respect to j_1 that is above α. Similarly, the feature $F_{\alpha,\mathrm{move}}^{(j_1,j_2;j_3)}$ considers the velocity of joint j_3 relative to joint j_1 and assumes the value one iff the component of this velocity in the direction determined by (j_1, j_2) is above α. For example, setting j_1 =‘belly’, j_2 =‘chest’, j_3 =‘rwrist’, one obtains the feature of Fig. 3 (c). $F_{\alpha,\mathrm{nmove}}^{(j_1,j_2,j_3;j_4)}$ has the same semantics, but the direction is given by the normal vector of the oriented plane spanned by j_1, j_2 and j_3. Of course, there are numerous other ways to define semantically meaningful generic features.

To determine reasonable joint combinations as well as suitable thresholds α for the generic features, we implemented a Feature Design GUI, which provides tools for visual feedback, statistical evaluations, and optimization. In designing our features, we incorporated most parts of the body, in particular the end effectors, so as to create a well-balanced feature set. One guiding principle was to cover the space of possible end effector locations by means of a small set of pose-dependent space “octants” defined by three intersecting planes each (above/below, left/right, in front of/behind the body). Obviously, this subdivision is only suitable to capture the rough course of a motion, since the feature function would often yield a constant output value for small-scaled motions. Here, our new features of type $F_{\alpha,\mathrm{move}}$ turned out to effectively grasp finer motion details. In general, our boolean features are designed to be zero for large portions of time (e.g., for a standing pose, all features assume the value zero), but still manage to capture important motion characteristics. Threshold specification is a delicate issue, since improper thresholds α may significantly degrade retrieval quality. To determine a suitable α for a given feature, we proceed as follows: we supply the system with a training set A of “positive” motions that should yield the feature value one for most of its frames and a training set B of “negative” motions that should yield the feature value zero for most of its frames. The threshold α is then determined by a one-dimensional optimization algorithm, which iteratively maximizes the occurrences of the output one for the set A while maximizing the occurrences of the output zero for the set B. Visual feedback about the resulting feature values aids the designer in fine-tuning α. Further boolean features can then be derived—supported by a textual editor—by taking boolean expression (AND, OR, XOR) of previously designed features.

2.2 Indexing

A major advantage of our approach is that all involved retrieval and indexing algorithms are time and space efficient. Here, the crucial point is that by incorporating spatio-temporal invariance in the relational features and induced segments, one can employ standard information retrieval techniques for fast content-based and fault-tolerant retrieval based on inverted lists, see [6]. In our system, we have divided the 22 features from Table 1 into two subsets F_ℓ (10 features) and F_u (12 features), as denoted by the column marked “set”. F_ℓ expresses properties of the lower part of the body (mainly of the legs), while F_u expresses properties of the upper part of the body (mainly of the arms).

Given a motion database (w.l.o.g. consisting of a single motion), we create an index I_ℓ for F_ℓ, which consists of the inverted list representation of the temporal segmentation of the motion, see Sect. 2. Analogously, we create a second index I_{u} for the feature set F_{u}. Note that in this approach, a feature set containing n features leads to 2^n possible feature vectors, each of which may give rise to an inverted list in the index. Thus, we restrict the maximum number of features per index to 12, corresponding to a maximum of 4096 inverted lists. Efficient retrieval can then be done by suitable union and intersection operations on the inverted lists, see [6] for further details.

3 User Interaction

Recall from Fig. 2 that a query specification consists of an example motion together with some fault tolerance settings and a suitable selection from the relational features listed in Table 1. In this section, we illustrate the typical workflow in our query-by-example system by means of a "jump" query as shown in Fig. 4, while presenting the corresponding elements of our Retrieval GUI.

Step 1: Specifying an example motion. In our system, a query motion is specified by selecting a frame range from a motion capture file, see Fig. 5 (a). Here, the selected jumping motion is a small excerpt from a sequence of pre-recorded sports movements, corresponding to segments 3–9 in Fig. 4. Note that in principle, it would also be possible to provide an example motion by *keyframing*, a process widely used in computer animation. Here, a motion is sketched by a few characteristic poses that are then interpolated. In a further input mode, the example motion could be generated on-line using suitable motion capturing hardware.

To get a better feeling for our query motion from Fig. 4, let us first focus on the movement of the legs. Both legs are kept parallel throughout the jump sequence and are stretched during the initial phase of arm-swing (segment 1 in Fig. 4). The legs are then bent into a half-squatting position (segments 2–3), preparing the following push-off (starting shortly before segment 4), during which the legs are stretched once more. In the landing phase (starting shortly before segment 6), the legs absorb the energy of the jump by bending as deep as before push-off. The jump sequence is concluded by stretching the legs into a normal standing position (segments 8–9). This sequence of bending and stretching the knees is characteristic for many kinds of parallel-leg jumping motions.

Step 2: Selecting suitable features. The above considerations show that it is reasonable to choose the features as in Fig. 4, where `kneeLeftAngle` and `knee-RightAngle` (corresponding to F^5/F^6 in Table 1) were selected. This is indicated in our GUI by a '`+`' mark, see Fig. 5 (b). In general, the strong semantics of geometric relations makes feature selection an intuitive process: the user can often anticipate which features will grasp important aspects of the query.

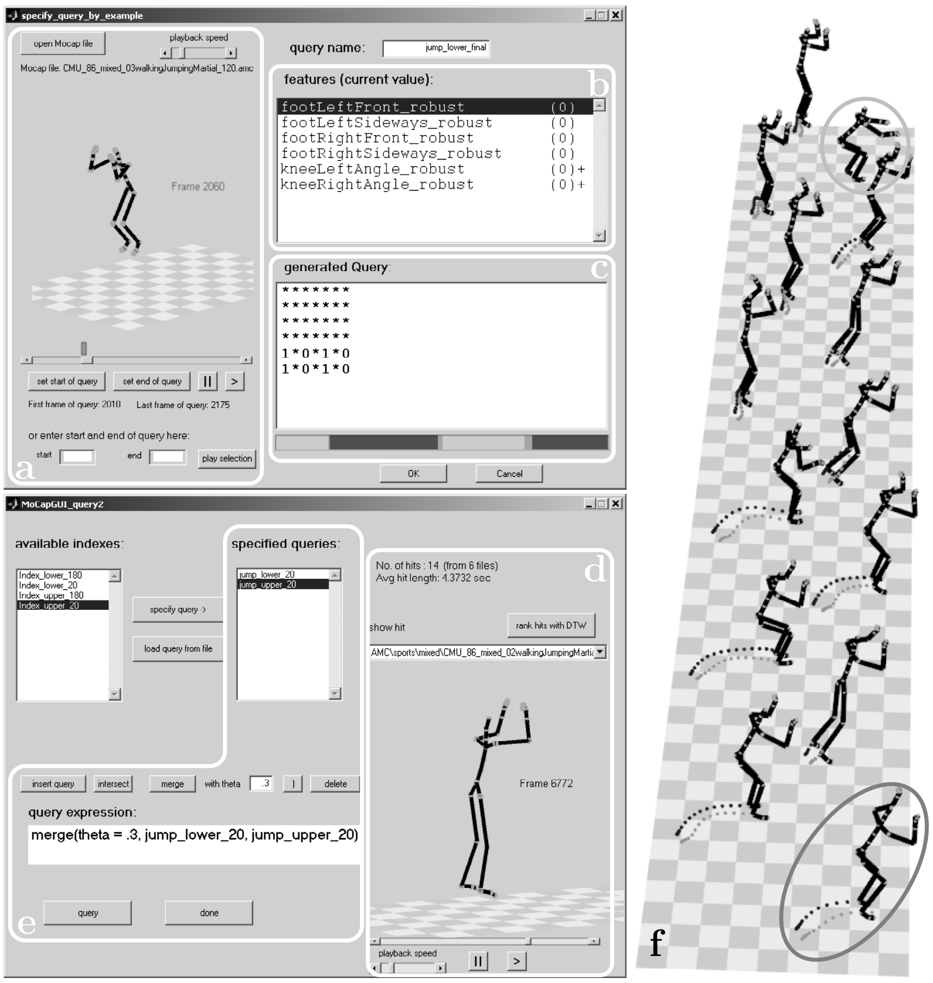
Since in Step 1, we only selected the frames corresponding to segments 3–9, we obtain the sequence $\left(\binom{1}{1},\binom{0}{1},\binom{0}{0},\binom{0}{1},\binom{1}{1},\binom{0}{1},\binom{0}{0}\right)$ with respect to F_5 and F_6, which is a subsequence of the feature progression given in the caption of Fig. 4. In our GUI, we represent this feature progression as a *query matrix* Q, where each column corresponds to a segment, each row corresponds to a feature, and where features from the chosen index that were not selected by the user are marked by rows of asterisks, e.g.,

$$Q = \left(\begin{array}{ccccccc} * & * & * & * & * & * & * \\ * & * & * & * & * & * & * \\ * & * & * & * & * & * & * \\ * & * & * & * & * & * & * \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 & 1 & 0 \end{array}\right) \begin{array}{l} F_2 \\ F_4 \\ F_1 \\ F_3 \\ F_6 + \\ F_5 + \end{array}$$

For the sake of simplicity, we omitted the features F_9, F_{10}, F_{13}, and F_{14}, which are constituents of I_ℓ. The features corresponding to each row are given to the right of Q, and the user-selected features are once more marked by a '+'. Such a query matrix is automatically generated by our system as soon as the frame range and the features have been selected (cf. Fig. 5 (c)). We visualize the structure of the current segmentation as a multicolored bar, as shown in the lower part of Fig. 5 (c). Here, each color corresponds to a feature vector, and the lengths of the colored bars are proportional to the segment lengths in frames.

Step 3: Setting up fault tolerance. The query matrix shown in Fig. 5 (c) differs from Q in that the user has inserted asterisks in columns 2, 4, and 6. These asterisks mask out irrelevant transitions between the alternating vectors $\binom{1}{1}$ and $\binom{0}{0}$ from the original feature sequence. In more detail, the feature vectors $\binom{0}{1}$ arise because the actor does not bend or stretch both legs at the same time. Instead, he has a tendency to keep the right leg bent a bit longer than the left leg. There are many possible transitions from, e.g., $\binom{1}{1}$ (legs bent) to $\binom{0}{0}$ (legs stretched), such as $\binom{1}{1} \rightarrow \binom{0}{0}$, $\binom{1}{1} \rightarrow \binom{0}{1} \rightarrow \binom{0}{0}$, or $\binom{1}{1} \rightarrow \binom{1}{0} \rightarrow \binom{0}{1} \rightarrow \binom{0}{0}$. This is the reason why we inserted the asterisk columns in Fig. 5 (c): each of the aforementioned transitions encodes the motion "stretching the legs", which is all that matters to our query. Our system automatically translates the asterisk notation into a suitable query.

Step 4: Querying, ranking and result presentation. Once the user has finalized the query matrix, the system starts the retrieval and then presents the hits to the user, who can browse the corresponding motion fragments in a graphical display, see Fig. 5 (d). The hits can be post-processed by means of a new DTW-based ranking strategy: a hit's ranking value is determined by the cost of a cheapest path in a cost matrix (c_{ij}), where each entry c_{ij} is the hamming distance between the i-th vector in the feature progression of the query and the j-th vector in the feature progression of the hit. At this stage, we use a segmentation that is induced by a compound feature function containing all features from Table 1, regardless of the features that were selected for the query. Note that DTW is only applied to a generally very small number of hits that were efficiently retrieved by our index-based approach and not to the entire database, as for most previous DTW-based methods. Furthermore, the cost matrices are typically very small because the query and the hits are compared at the segment level instead of the frame level, thus working on strongly downsampled motion representations.



Mocap file: CMU_86_mixed_03walkingJumpingMartial_120.amc

Frame 2060

First frame of query: 2010 Last frame of query: 2175

or enter start and end of query here: start end

query name: jump_lower_final

features (current value):

```
footLeftFront_robust        (0)
footLeftSideways_robust     (0)
footRightFront_robust       (0)
footRightSideways_robust    (0)
kneeLeftAngle_robust        (0)+
kneeRightAngle_robust       (0)+
```

generated Query:

```
*******
*******
*******
*******
1*0*1*0
1*0*1*0
```

available indexes: Index_lower_180, Index_lower_20, Index_upper_180, Index_upper_20

specified queries: jump_lower_20, jump_upper_20

No. of hits : 14 (from 6 files)
Avg hit length: 4.3732 sec

AMC\sports\mixed\CMU_86_mixed_02walkingJumpingMartia

Frame 6772

with theta .3

query expression:

merge(theta = .3, jump_lower_20, jump_upper_20)

Fig. 5. Left: query interface. Right: fourteen hits for a "jump" query on $\mathcal{D}^{20}$. The query motion (foreground) and a false positive hit (background) are highlighted.

Step 5: Combining queries. Our example query as it has been discussed so far yields a large number of hits in our test database, some of which are false positive hits. For example, many squatting motions exhibit the same progression of knee bending and stretching. To further refine our query, we can incorporate additional constraints regarding the upper part of the body, i.e., the index I_u. To this end, the user may specify several independent queries (on different indexes or even on the same index), which can then be combined in a query expression by means of *merge* and *intersect* operations, cf. Fig. 5 (e). Merging two sets of hits H_1 and H_2 means that a hit $h_1 \in H_1$ is reported as a result if there is a hit $h_2 \in H_2$ that overlaps h_1 by more than a fraction of $\theta \in (0, 1]$ (relative to the

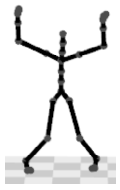
	jump	cartwheel	elbow-to-knee	jumping jack	punch
					
recall on $\mathcal{D}^{20}$	13/16	4/4	14/14	19/19	24/24
precision on $\mathcal{D}^{20}$	13/14	4/4	14/15	19/19	24/37
precision on $\mathcal{D}^{180}$	17/27	4/59	14/27	19/29	83/153
$p_5^{180} \mid p_{10}^{180} \mid p_{20}^{180}$	5 | 10 | 16	4 | 4 | 4	5 | 9 | 14	5 | 10 | 18	5 | 10 | 20
query time on $\mathcal{D}^{180}$ (s)	0.87	0.06	0.39	0.60	0.08
ranking time on $\mathcal{D}^{180}$ (s)	4.42	5.87	1.04	1.54	4.46
total length of hits in $\mathcal{D}^{180}$ (s)	136.45	125.89	15.08	37.87	128.53

Table 2. Summary of query results. See Sect. 4 for a discussion.

length of h_1, measured in seconds). Our intersection operator coincides with the set theoretical intersection of the hits interpreted as time intervals. The results of a combined "jump" query, which uses merging to incorporate a query that focuses on the arm motion, are discussed in Sect. 4 and visualized in Fig. 5 (f).

Step 6: Iteratively refining the query. Assisted by the system's feedback, a user can modify his query at several points in the query process. Initially, he may modify the frame range and feature selection so as to achieve a segmentation that is characteristic, but not too specific. Here, the query matrix and the segmentation bar support the user's decisions. Short segments in the segmentation bar hint at transitions that may be masked out by means of an asterisk column, cf. Step 4. After inspecting the hits for the initial query, the user may reconsider the current query settings. If, for instance, many false positive hits have been retrieved, it often helps to extend the query motion or to incorporate additional features that discern the query motion from the false positives. If, on the other hand, only few hits are retrieved, the user may decrease the complexity of the induced segmentation by reducing the number of selected features, by reducing the length of the example motion, or by decreasing the merging parameter θ.

4 Experimental Results

Our motion retrieval system was implemented in Matlab, and experimental results were obtained on a 3.6 GHz Pentium 4 with 1 GB of main memory. We evaluated the system on a subset $\mathcal{D}^{180}$ of the CMU database [1], which contains about one million frames of motion capture data ($\sim$180 minutes sampled at 120 Hz) and covers a wide range of motions.

The columns of Table 2 show the results for five exemplary queries: a forceful jump, a cartwheel, the gymnastics motions "elbow-to-knee" and "jumping jack", and a punch with the right fist. Further information and result videos for these queries can be found at `http://www-mmdb.iai.uni-bonn.de/projects/mocap/RetrievalGUI.html`. In order to determine the recall (proportion of relevant hits that were retrieved) of our queries, we manually annotated a subset $\mathcal{D}^{20}$ of $\mathcal{D}^{180}$ consisting of about 145,000 frames ($\sim$20 minutes). $\mathcal{D}^{20}$ includes a diverse range of about 15 different gymnastics motions, some walking, jogging, jumping, climbing sequences, several martial arts moves, pantomime, and basketball motions. The precision (proportion of retrieved hits that are relevant) of query results was evaluated on both $\mathcal{D}^{20}$ and $\mathcal{D}^{180}$. Our retrieval concept focuses on maximizing the recall for any given query, which may lead to a relatively low precision. However, we improve the quality of our retrieval results by applying DTW-based ranking as explained in Sect. 3. In order to evaluate this ranking, we counted the number of relevant hits within the top 5, top 10, and top 20 positions of the ranked hit list, denoted in Table 2 by p_5^{180}, p_{10}^{180}, and p_{20}^{180}, respectively.

Our query for parallel-leg jumps (first column of Table 2) successfully retrieved 13 of the 16 jumping motions that were contained in $\mathcal{D}^{20}$, leading to a recall of 13/16. The three remaining jumps were missed because the actors' knees were not bent far enough to trigger our features. The total number of hits was 14 (cf. Figure 5 (f)), only one of which was a false positive, namely, the squatting motion that is highlighted in Figure 5 (f). This leads to a precision of 13/14 on $\mathcal{D}^{20}$. On $\mathcal{D}^{180}$, four additional jumps and nine additional false positives were found, leading to a precision of 17/27. There were no false positives in the top 5 and top 10 positions of the ranked hit list, and only one of the 17 relevant hits did not appear in the top 20. Only 136.45 seconds of $\mathcal{D}^{180}$, or 1.26%, had to be inspected by the DTW-based ranking procedure. The index-based retrieval step took 0.87 seconds for the $\mathcal{D}^{180}$ database, whereas the ranking consumed another 4.42 seconds. Note that in general, the running time for the ranking step correlates with the total length of the retrieved hits.

The query for cartwheels starting with the right leg in front uses only two different features, F_9 and F_{10}, which test whether the feet move with a high velocity. The resulting feature sequence $\left(\binom{0}{0},\binom{0}{1},\binom{1}{1},\binom{1}{0},\binom{0}{0}\right)$ reflects how the left leg is kicked up in the air, quickly followed by the right leg. The left leg touches the ground first, leading to a low velocity in this phase, again followed by the right leg. This query is characteristic enough to lead to perfect precision and recall values on $\mathcal{D}^{20}$. Its precision on $\mathcal{D}^{180}$ is very low, as there are 55 false positive hits, but our ranking strategy places the cartwheels among the top 5 hits. In contrast to the three other examples, the cartwheel and punch queries operated on one index only, which makes merging superfluous—hence the low retrieval times of 0.06 and 0.08 seconds, respectively.

For the gymnastics motions "elbow-to-knee" and "jumping jack", a larger number of spurious hits was returned on $\mathcal{D}^{180}$, but our ranking succeeds in placing the relevant hits at the top of the list. The queries "cartwheel", "elbow-to-knee", and "jumping jack" achieve perfect recall values on $\mathcal{D}^{20}$. No additional relevant hits for these queries appear in the results for $\mathcal{D}^{180}$ because to our knowledge, all of these rather specific motions are already contained in the subset $\mathcal{D}^{20}$ of $\mathcal{D}^{180}$. For characterizing right-handed punches, we used F_7, F_{11}, F_{21}, and, to improve precision, F_{22}. Once again, the recall on $\mathcal{D}^{20}$ is perfect, while the precision is 24/37. On $\mathcal{D}^{180}$, the precision decreases to roughly 50%, but this is successfully compensated for by our ranking.

5 Conclusions and Future Work

In this paper, we presented a Retrieval GUI for content-based motion retrieval, where the query consists of a motion clip as well as a user-specified selection of motion aspects to be considered in the retrieval process. Based on the concept of quantitative relational features as introduced in [6], we suggested several generic boolean features, which can then be used—aided by our Feature Design GUI—to determine a set of semantically meaningful features covering a wide range of motion aspects. Being in a way conceptually orthogonal to computationally expensive DTW-based strategies, our technique is ideally suited to efficiently cut down the search space in a pre-processing step, thus making DTW-based techniques applicable to large data sets. This finding is supported by our experimental results, see Sect. 4.

Motion reuse based on morphing and blending as used in computer animation may require batch techniques to automatically retrieve suitable motion fragments. To this end, we plan to automate the feature selection step using statistical methods. Furthermore, we are developing and analyzing DTW-based ranking strategies based on different cost measures. First experiments showed that our relational approach to motion description not only constitutes a possible framework for flexible and efficient retrieval mechanisms, but also for automatic classification and annotation of motion data.

References

1. CMU, *Carnegie-Mellon MoCap Database.* Created with funding from NSF EIA-0196217. `http://mocap.cs.cmu.edu`, 2003.
2. K. Forbes and E. Fiume, *An efficient search algorithm for motion data using weighted PCA*, in SCA '05: Proc. 2005 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, New York, NY, USA, 2005, ACM Press, pp. 67–76.
3. E. J. Keogh, T. Palpanas, V. B. Zordan, D. Gunopulos, and M. Cardle, *Indexing large human-motion databases*, in Proc. 30th VLDB Conf., Toronto, 2004, pp. 780–791.
4. L. Kovar and M. Gleicher, *Automated extraction and parameterization of motions in large data sets*, ACM Trans. Graph., 23 (2004), pp. 559–568.
5. G. Liu, J. Zhang, W. Wang, and L. McMillan, *A system for analyzing and indexing human-motion databases*, in SIGMOD '05: Proc. 2005 ACM SIGMOD Intl. Conf. on Management of Data, New York, NY, USA, 2005, ACM Press, pp. 924–926.
6. M. Müller, T. Röder, and M. Clausen, *Efficient content-based retrieval of motion capture data.*, ACM Trans. Graph., 24 (2005), pp. 677–685.
7. Vicon, *3D optical motion capture.* `http://www.vicon.com`.
8. M.-Y. Wu, S. Chao, S. Yang, and H. Lin, *Content-based retrieval for human motion data*, in 16th IPPR Conf. on Computer Vision, Graphics and Image Processing, 2003, pp. 605–612.