Efficient video-based retrieval of human motion with flexible alignment

Ankur Gupta* John He† Julieta Martinez* James J. Little* Robert J. Woodham*

University of British Columbia

*{ankgupta,julm,little,woodham}@cs.ubc.ca †j.o.h.n@alumni.ubc.ca

Abstract

We present a novel and scalable approach for retrieval and flexible alignment of 3d human motion examples given a video query. Our method efficiently searches a large set of motion capture (mocap) files accounting for speed variations in motion. To align a short video clip with a part of a longer mocap sequence, we experiment with different feature representations comparable across the two modalities. We also evaluate two different Dynamic Time Warping (DTW) approaches that allow sub-sequence matching and suggest additional local constraints for a smooth alignment. Finally, to quantify video-based mocap retrieval, we introduce a benchmark providing a novel set of per-frame *action labels for 2 000 files of the CMU-mocap dataset, as well as a collection of realistic video queries taken from YouTube. Our experiments show that temporal flexibility is not only required for the correct alignment of pose and motion, but it also improves the retrieval accuracy.*

1. Introduction

The problem of aligning two temporal sequences arises naturally in diverse applications such as financial analysis [7], medical diagnosis [28], handwriting analysis [3], and speech recognition [21]. Along with the alignment, these applications often require a measure of similarity between time-series data. Although efficient domain-specific techniques for some problems exist (e.g., [16] for mocap-to-mocap comparison), computing the similarities of time-series in large datasets with variable length sequences remains a challenging task in general.

In this paper we focus on finding an efficient and scalable method for alignment and distance computation between human motion sequences across two modalities — video and motion capture. The problem occurs naturally when searching large databases of motion capture data; often, mocap examples contain an actor performing *multiple* activities (*e.g*., the actor may get up, walk, run and kick in a martial arts sequence); however, we want to search for *a single* action depicted in a short video query. In short, we wish to (a) retrieve mocap files with similar actions and (b) align the video query to the retrieved mocap subsequence.

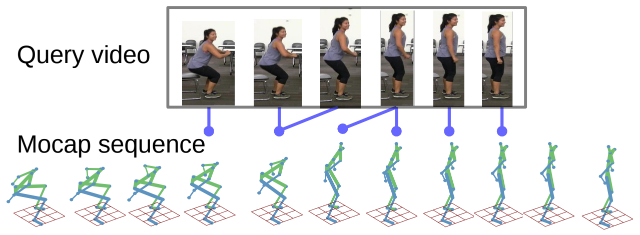


Figure 1: An illustration of flexible alignment between a short query video and a longer motion capture sequence.

A key motivation for our work is that an effective search through existing mocap files can save the effort of collecting the data multiple times. For example, efficient mocap retrieval can help animators find relevant clips to reuse in animation. Another motivation is the use of similar, aligned mocap sequences to blend them and generate new animations satisfying higher-level goals and constraints [13]. In addition to computer animation, retrieving similar motion capture sequences given a short video as a query is a crucial step in various vision applications such as 3d human pose tracking [22] and action recognition [8].

The related problem of large-scale video retrieval often relies on inflexible matching (*i.e*., one-to-one frame correspondence) to align sequences efficiently [23]. We hypothesize that, due to the style and speed variations in human motion, *flexible* temporal alignment is a better approach to improve matching quality — when comparing semantically similar human motion with slight temporal variations, inflexible alignment tends to assign a poor score due to the lack of exact alignment.

Dynamic Time Warping (DTW) is a widely used technique for flexible alignment of any two temporal sequences.

DTW is guaranteed to find the optimal alignment given pairwise frame similarities between two sequences maintaining the monotonicity constraint, which is central to temporal matching. Unfortunately, DTW is better suited for alignment than for retrieval, because it assumes that the first and last frames of two sequences are always aligned. This assumption is reasonable when the sequences are roughly of the same length, but, in a scenario such as ours where we want to find the best alignment of a *short* (*e.g*., 40-frame) video sequence to a *long* (*e.g*. 2-minute) mocap sequence, DTW is no longer effective (see Figure 1). The naïve solution of running DTW in a sliding-window fashion is computationally expensive, with an asymptotic cost cubic in the number of frames [35], while relatively efficient alternatives [15] often make other restricting assumptions.

In this work, first we formalize the task of video-based mocap retrieval and alignment by proposing a novel benchmark (V3dR); second, we establish baselines for the task using efficient temporal matching techniques (flexible and inflexible) and evaluate them thoroughly on the benchmark. Finally, we suggest effective features for matching video and mocap that are comparable across the two modalities.

2. Related Work

Mocap retrieval. Efficient retrieval of human motion data can help animators find relevant mocap clips to reuse in an animation. Various input modalities including hand-drawing [4, 5], wooden puppets [6, 19], and Kinect [12] have been used to provide this functionality. However, monocular video remains a largely unexplored and challenging input modality.

The retrieval of a single 3d pose given an image of a person has been used for 3d human pose estimation [25] and tracking [22, 33]. Ren *et al*. [22] search for 3d pose examples using Haar-like features based on silhouettes from multiple synchronized views; they add temporal consistency and restrict search using motion graphs. More recently Yasin *et al*. [33] have used features inspired by content-based mocap retrieval for 3d pose tracking. The above methods retrieve poses *per frame*, and then add temporal consistency for tracking. In contrast, we are interested in aligning the whole video sequence to a mocap example.

In the only work that we know to deal with a direct video-to-3d matching, Gupta *et al*. [8] retrieve short mocap sequences given a video input as part of a cross-view action recognition pipeline. Although they show qualitative results, a quantitative evaluation of the retrieval problem remains unaddressed.

Flexible alignment. Dynamic time warping (DTW) is a popular algorithm used to align temporal sequences and to cluster time-series data. DTW uses dynamic programming to find the minimum-cost alignment of two sequences subject to constraints suitable for time-series data (*i.e*., monotonicity and local continuity). FastDTW provides an efficient approximation to DTW by solving the problem iteratively at multiple scales, achieving an asymptotic complexity of $\mathcal{O}(n)$ [24] in the length of the sequence, as opposed to $\mathcal{O}(n^2)$ for the original DTW. One of the limiting assumptions in DTW and its variants is that the first and the last frames of the two sequences must be aligned (the end-point constraint). This works well when comparing similar-sized sequences; however, the end-point constraint is not suitable for sequences of different lengths.

Some recent methods have attempted to relax the end-point constraint partially, by fixing one of the ends but letting the other float [26, 27]. DTW-S [35] matches sequences of different sizes and allows flexibility at both ends by assuming *balanced* alignment, *i.e*., the warping uses an equal number of frames from both sequences. This assumption is violated when we match actions performed at different speeds. Subsequence DTW [15] is an efficient method for relaxing the end-point constraint; however, it is biased towards choosing a shorter database subsequence for a given query (more details in Section 4). Normalizing the score with a measure of path length [2, 18] can remove this bias, but leads to a non-smooth alignment. We suggest a simple local constraint to smooth the alignment.

An alternate approach to sequence alignment is to treat the warping as a discrete version of a monotonic function. For instance, GCTW [36] poses alignment as an optimization over a continuous space. GCTW can also incorporate floating end-points; however, due to a non-convex objective function, GCTW requires effective initialization to avoid local minima. In our work, we build upon the original dynamic programming formulation, which does not require initialization. For efficiency, we implement all the methods on top of the FastDTW [24] code.

Mocap alignment/retrieval benchmarks. The two main large and freely available mocap datasets widely used in research are CMU [1] and HDM05 [17], but neither provide the frame-level annotations needed to evaluate the alignment of similar human motion. A recently introduced dataset, Human3.6M [9], provides a large number of annotated video frames with synchronized mocap. However, the action labels correspond to very high level activities (such as making purchases and greeting) which can exhibit high variability depending on the particular context. Hence, none of these benchmarks are suited for testing both retrieval *and* alignment. To bridge this gap we introduce a new benchmark called V3dR.

3. V3dR: Video-Based 3d Motion Retrieval

We formulate the task of 3d motion retrieval as, given a video query, finding a snippet from a large mocap database

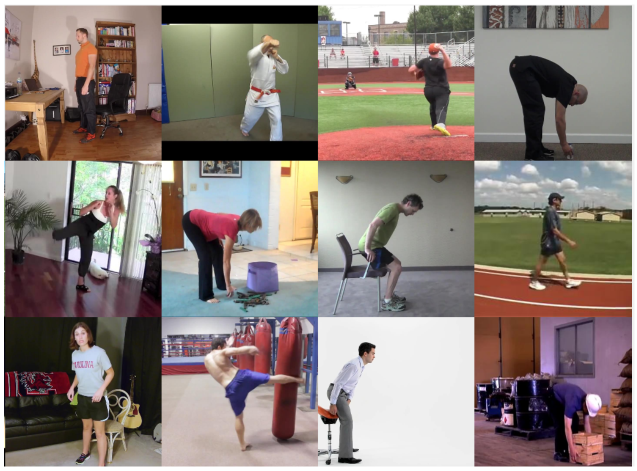


Figure 2: Example queries obtained from YouTube videos as a part of our benchmark.

that resembles the human motion in the video. Following previous work, which assumes that an action category is a good proxy for 3d motion [32, 34], we define similarity between a video query and the retrieved sequence using their action labels.

As part of the benchmark, we have assembled a series of video queries, a database of 3d motion sequences (mocap), ground truth annotations and two protocols for evaluation.

Queries. We prepare two sets of queries to evaluate our method: videos captured in a controlled environment, as well as a more challenging set of videos in-the-wild downloaded from YouTube. To factor out occlusion and camera motion, we choose videos where the full body is mostly visible and there is little or no camera motion. The videos are typically short snippets of 1 to 4 seconds in length. Figure 2 shows some of the frames from example video sequences.

We obtained another set of video queries from the IXMAS dataset [30], a benchmark that consists of short video sequences where 10 actors perform simple actions in a laboratory environment. This dataset is typically used to evaluate cross-view action recognition, and contains videos captured from 5 distinct camera views. While most of the YouTube videos have similar camera elevations, these video snippets from IXMAS add variation in viewpoint. We pick our queries randomly from cameras 1-4, excluding the fifth camera due to its sharp elevation angle, which we find unrealistic in natural videos.

Both sets contain the following 8 classes: *sit down*, *get up*, *turn*, *walk*, *punch*, *kick*, *pick up* and *throw overhead*. Each group of queries has 20 videos per action, with a single person performing the action. This amounts to a total of 160 queries per set. We also provide manually-annotated bounding boxes around the person in each video.

3d human motion database. We use the CMU-mocap dataset [1] – the largest publicly-available mocap database that we are aware of. From the 2 549 sequences, we kept the 2 000 shortest, which results in around 4.5 hours of 3d human motion. To make mocap and video data comparable we sub-sample each mocap sequence to 24 fps.

Ground truth. We annotated the mocap sequences with the same action labels as the queries. Since one mocap sequence may contain more than one action, we annotated the 4.5 hours of motion data *per frame*. These annotations are not necessarily temporally exclusive (*e.g.*, a person might walk and turn at the same time). 976 sequences were annotated with at least one class, resulting in 1024 distractor sequences. Figure 3 shows an example annotation.

Both IXMAS and CMU-mocap are publicly available, and we are releasing our set of annotations, as well as our list of video queries, to facilitate future comparisons[1].

Evaluation. We propose two tasks to evaluate performance on V3dR, which correspond to different use cases of video-to-3d-motion retrieval. The first evaluation protocol is called *detection modality*. In this modality, a retrieved example is counted as a true positive if it contains the action featured in the video. In the second protocol, *localization modality*, retrieval is expected to produce both a ranking and a frame number that localizes the action in each 3d sequence in time. A retrieved 3d sequence is counted as a true positive if it contains the queried action *and* its localization is correct. To evaluate localization, we compare the query label against the ground truth mocap annotation at the frame number returned by the algorithm.

We evaluate our *detection modality* using mean Average Precision (mAP) which is defined as the mean of the APs over all queries, and serves as a single number to evaluate performance per class. We evaluate our *localization modality* using recall@N curves [10]. For each query, we plot the number of true positives (both in sequence and localization) in the top N retrieved sequences over the total number of positive examples in the database. This results in a monotonically increasing curve for increasing N. We show the average of these curves over all queries.

4. Retrieval with flexible alignment

To help the discussion, first we present some background and introduce notation. Let us consider two sequences X, Y of length T_x and T_y, consisting of vector values $(\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_{T_x})$ and $(\mathbf{y}_1, \mathbf{y}_2, ..., \mathbf{y}_{T_y})$, $\mathbf{x}_i, \mathbf{y}_i \in \mathbb{R}^d$ where X is a query, and Y is one of the database sequences. The function $\mathrm{dist}(i_x, i_y)$ returns the distance between the two frames $\mathbf{x}_{i_x}$ and $\mathbf{y}_{i_y}$ of the respective sequences. A warp can be described using a pair of functions

[1]http://www.cs.ubc.ca/labs/lci/v3dr/

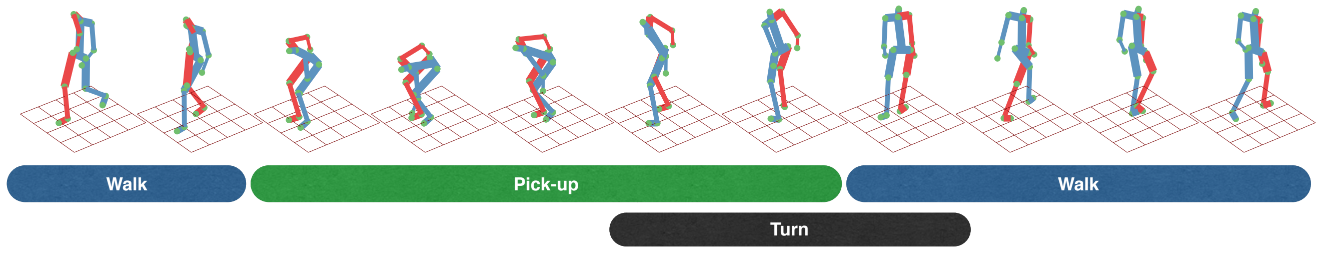
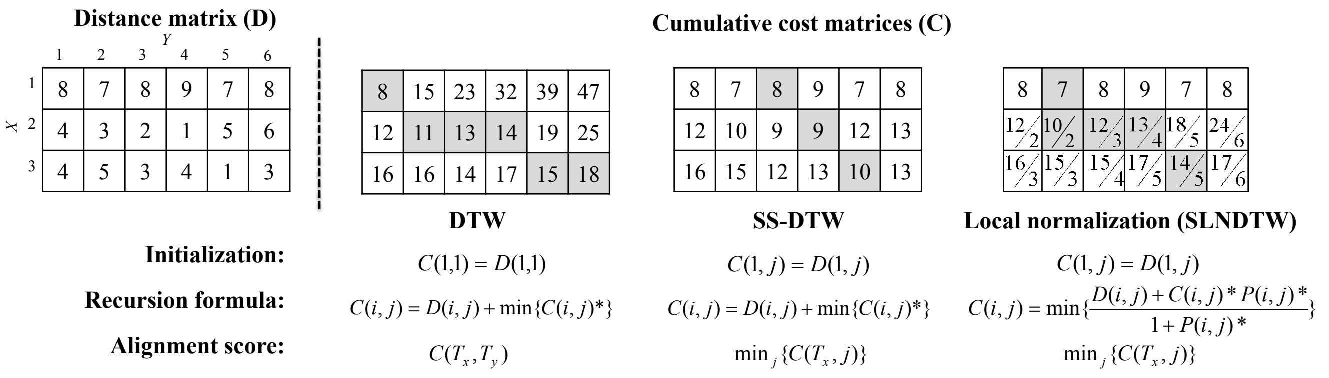


Figure 3: An example of mocap annotations provided in the V3dR benchmark. At the top, we show a few frames of a mocap sequence. The corresponding action labels are shown below. Note that the annotations are not temporally exclusive. As shown above, one frame can have multiple labels. We use these annotations to evaluate video to mocap alignment.



Distance matrix (D)

X \ Y	1	2	3	4	5	6
1	8	7	8	9	7	8
2	4	3	2	1	5	6
3	4	5	3	4	1	3

Cumulative cost matrices (C)

DTW

8	15	23	32	39	47
12	11	13	14	19	25
16	16	14	17	15	18

SS-DTW

8	7	8	9	7	8
12	10	9	9	12	13
16	15	12	13	10	13

Local normalization (SLNDTW)

8	7	8	9	7	8
12/2	10/2	12/3	13/4	18/5	24/6
16/3	15/3	15/4	17/5	14/5	17/6

	DTW	SS-DTW	Local normalization (SLNDTW)
Initialization:	$C(1,1) = D(1,1)$	$C(1,j) = D(1,j)$	$C(1,j) = D(1,j)$
Recursion formula:	$C(i,j) = D(i,j) + \min\{C(i,j)^*\}$	$C(i,j) = D(i,j) + \min\{C(i,j)^*\}$	$C(i,j) = \min\{\frac{D(i,j) + C(i,j)^* P(i,j)^*}{1 + P(i,j)^*}\}$
Alignment score:	$C(T_x, T_y)$	$\min_j\{C(T_x, j)\}$	$\min_j\{C(T_x, j)\}$

Figure 4: A toy example to illustrate different warping algorithms. The grey boxes in the cumulative cost matrices C represent the chosen warp path for the same distance matrix D shown on the left. Each algorithm initializes some cells of C, fills the rest according to a recursion formula, and then chooses a final score $\text{dist}_s(X, Y)$ for the alignment. The set of indices $\{(i-1, j), (i-1, j-1), (i, j-1)\}$ is abbreviated as $(i, j)^*$. In the case of local normalization, $P(i, j)$ is the length of the chosen normalized warp path for the subproblem up to $D(i, j)$.

$\phi(.) = \{\phi_x(.), \phi_y(.)\}$, which map aligned frame indices to the original X, Y indices. The distance between the warped sequences can be defined as

$$\text{dist}_s(X, Y) \equiv \sum_{i=1}^{T_\phi} \text{dist}(\phi_x(i), \phi_y(i)) \quad (1)$$

where T_ϕ is the warped length after alignment. A *warp path* is the sequence of indices $(\phi_x(i), \phi_y(i))$, for i from 1 through T_ϕ.

DTW. In the original Dynamic Time Warping (DTW) formulation, warp paths are constrained such that $(\phi_x(i+1), \phi_y(i+1)) - (\phi_x(i), \phi_y(i))$ equals either $(1, 0)$, $(1, 1)$, or $(0, 1)$. This property enforces monotonicity and continuity of DTW's warp paths, which are needed for a well-behaved alignment of time-series (for details see [21]). Another important consideration in DTW is the *end-point* constraint which mathematically translates to

$$\phi_x(1) = 1, \phi_x(T_\phi) = T_x; \phi_y(1) = 1, \phi_y(T_\phi) = T_y, \quad (2)$$

and forces the algorithm to use the full length of both sequences.

Given these constraints, DTW solves the minimization problem $\text{argmin}_\phi \, \text{dist}_s(X, Y)$ using dynamic programming to obtain the optimal warp path. The distance function $\text{dist}(.)$ is used to fill a matrix $D \in \mathbb{R}^{T_x \times T_y}$, where each element $D(i, j)$ is the distance between $\mathbf{x}_i$ and $\mathbf{y}_j$ (see Figure 4 for an example D matrix). DTW aggregates the cost of matching subproblems in a cumulative cost matrix C such that $C(T_x, T_y)$ returns the score of the optimal alignment between the original sequences. Figure 4 also shows the initialization and the update rules for filling matrix C. Even though DTW returns the lowest-cost warp path, because of the aforementioned end-point constraint it cannot localize a query within a larger database file.

SS-DTW. Müller [15] presents a modified version of DTW, called Subsequence DTW (SS-DTW), that removes the end-point constraint. Instead of finding the lowest-cost alignment using both the sequences, we can minimize

$$\underset{y_1,y_2,\phi}{\operatorname{argmin}} \operatorname{dist}_s(X, Y(y_1 : y_2)), \tag{3}$$

where $Y(y_1 : y_2)$ is a subsequence of Y *i.e.*, $T_y \geq y_2 \geq y_1 \geq 1$. The solution to this problem can also be obtained using a dynamic program, as shown in Figure 4. Note the changes, compared to DTW, in the initialization of the cumulative cost matrix and calculation of the final score. Conceptually, this can be thought of as placing two "special frames" at each end of the query sequence X, which can align with any extraneous frames in Y for no cost.

4.1. Adding normalization

Although SS-DTW loosens the end-point constraint, it is biased towards matching shorter database subsequences. The increased warp path length due to a longer subsequence severely penalizes the SS-DTW objective. Since it is just as likely for a motion occurring in a database sequence to be slower than a motion occurring in a query sequence as it is the other way around, this imbalance in flexibility is problematic. To get rid of this bias we consider a new objective that normalizes the distance by the warp path length

$$\underset{\phi}{\operatorname{argmin}} \sum_{i=1}^{T_\phi} \frac{1}{T_\phi} \operatorname{dist}(\phi_x(i), \phi_y(i)). \tag{4}$$

Note that T_ϕ, the length of the warp path, is dependent on ϕ. This problem is known as the *normalized edit distance problem* [14], which can again be solved using dynamic programming. However, in this case the dynamic program needs another dimension for path length, leading to a runtime of $\mathcal{O}(n^3)$ in the sequence length, which is not suitable in a retrieval setting.

Since this expensive normalization is not practical, we use a local normalization approach (SLNDTW) presented in [18]. This $\mathcal{O}(n^2)$ approximation ($\mathcal{O}(n)$ when built on top of FastDTW [24]) works well in practice. Rather than building a three-dimensional cumulative cost array – over query frames, database frames, and the warp path length – SLNDTW just keeps track of the path length (obtained greedily) so far in a separate matrix P. See Figure 4 for the update rule. Note that the initialization depicted here differs slightly from the original SLNDTW initialization. Here we only focus on the normalization of the score. Although SLNDTW does not return an optimal path, as defined by Equation 4, it often finds a warp path very close to the best normalized path on our data (see Figure 5).

4.2. Smoothing warp paths

Since the original DTW objective is the sum of distances between warped frames of the two sequences, the warp path is regularized. Adding an extra frame in the warp path adds a positive value to the overall cost. But once we normalize the cost by path length, the objective has no preference for shorter path lengths. As a result, normalization leads to "jagged" warp paths. However, a smooth warp path is more desirable because it better describes the distortions due to variations in speed. In order to achieve smoothness, we use simple slope weighting [21] which applies local constraints by associating small multiplicative costs to horizontal and vertical movements in the warp path. Thus the final update rule becomes

$$C(i,j) = \min \left\{ \frac{D(i,j) + w^* C(i,j)^* P(i,j)^*}{1 + P(i,j)^*} \right\}$$

where w^* are weights $\{w_{10}, w_{11}, w_{01}\}$ for indices $(i,j)^* \equiv \{(i-1,j), (i-1,j-1), (i,j-1)\}$ respectively. We choose $w_{01} = w_{10} = 1.15$ and $w_{11} = 1$ for all our experiments.

5. Experiments

We test our approach on the V3dR dataset (described in Section 3). First we describe some of the implementation details, and then present our results.

5.1. Implementation details

Frame similarity measure. Following Gupta *et al.* [8], we represent each video frame as an aggregation of dense trajectory features [29], using the publicly available implementation of Wang *et al.*[2]. To ensure a consistent number of features across the video, we compute trajectories in a sliding-window manner. We use mocap dense trajectories [8][3] as an equivalent feature description for mocap. We set the trajectory length of 10 frames in all our experiments. To aggregate these features for each frame, we use Fisher Vectors [20], instead of the Bag of Words (BoW) aggregation used by Gupta *et al.* [8]. We fit a Gaussian Mixture Model (GMM) to the trajectory descriptors obtained from the query videos, and use these GMM parameters to generate the Fisher Vector representation for both video and mocap frames.

In addition to trajectory-based motion descriptors, we note that 2d pose can provide complementary information for retrieval. Recent work has shown that even the noisy pose estimate obtained from 2d pose detection is very effective as a high-level feature for action recognition [11]. Following this intuition, we use the relational pose features described by Jhuang *et al.* [11][4]. These features describe relative positions and orientations of 2d joints, and do not depend on their absolute location. We normalize each dimension of the relational feature descriptor to obtain a zero-

[2]http://lear.inrialpes.fr/people/wang/dense_trajectories

[3]https://github.com/UBC-CVLab/mocap-dense-trajectories

[4]https://github.com/UBC-CVLab/rel-pose-feats

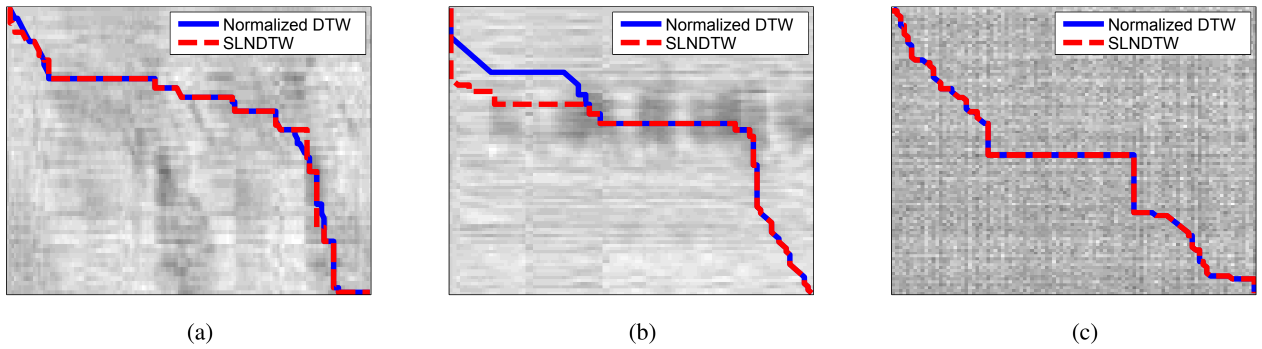


Figure 5: A comparison between exact normalization (as defined in Equation 4), using dynamic programming over a 3d cumulative cost array, and approximate normalization used in SLNDTW [18]. The corresponding distance matrix (D) is shown behind each warp path, with darker shades representing smaller distances. We show results using (a) a query from IXMAS and a CMU-mocap file, (b) a query from YouTube and a CMU-mocap file, and (c) two sequences of length 100 containing random, l_2-normalized 256-dimensional vectors. We notice that the approximate local normalization (SLNDTW) works fairly well in all three cases, while being several times faster. Note that we have kept the end-point constraint in this figure for simplicity.

mean and unit variance across all video queries, and then l_2-normalize each descriptor. We use the Flexible Mixture of Parts [31] to obtain 2d poses from video. For mocap sequences, a comparable representation is obtained by projecting the corresponding 3d joint locations to a given viewpoint (discussed in the next paragraph). We concatenate pose and trajectory features after separately PCA projecting them to 128 dimensions, and obtain a 256-dimensional descriptor for each video and mocap frame.

We compare the effectiveness of these trajectory and pose descriptors in Table 1. For this comparison, we fix the final feature dimension to 256 and use CTE [23] as our retrieval algorithm. We note that the pose features perform better compared to trajectories on most action classes. However, a combination of the two descriptors gives the best retrieval performance as evaluated by mAP. Thus, we use a concatenation of pose and trajectory (P + T) features for all further experimentation.

Viewpoint independent alignment. Following Gupta *et al.* [8], we generate descriptors for mocap as seen for 18 different viewpoints. Since we assume orthographic projection, we only need to specify elevation, and azimuthal angles. We choose the same angles as in [8]. We match a video query against all mocap viewpoints, and the best match for each mocap file is obtained by choosing the lowest-cost alignment across all viewpoints.

DTW and CTE implementations. We implement all the retrieval methods (except CTE) as modifications on the publicly available FastDTW implementation by Stan Salvador[5]. Thus, we build on an $\mathcal{O}(n)$ approximation of DTW.

[5]https://code.google.com/p/fastdtw/

We used our own CTE implementation with the same set of parameters as described in [8]. While CTE implicitly assumes a dot-product similarity measure, for DTW-based approaches, we define the distance between two normalized frame descriptors, $\mathbf{p}$ and $\mathbf{q}$, to be $(1 - \mathbf{p} \cdot \mathbf{q})$.

Furthermore, to report alignment in the case of CTE, we return the mocap frame matching the central video frame of the query; and for SS-DTW and SLNDTW we use the mocap frame at the middle of the warp path.

5.2. Retrieval and alignment performance

We show the recall@N results averaged over each of the query sets in Figure 6. Flexible matching methods, SS-DTW and SLNDTW (+smooth), consistently perform better at localizing and retrieving relevant sequences compared to inflexible approaches (CTE and [8]). These results demonstrate the importance of flexibility in human motion retrieval and alignment.

Table 1 shows our results on the detection modality. Here we report mean average precision numbers per class averaged over both the IXMAS and YouTube query sets. We again observe that flexible matching consistently outperforms the previous method of Gupta *et al.* [8]. Additionally, using SLNDTW (+smooth), we are able to improve the overall mAP by 16% over the inflexible baseline (CTE) and around 8% over SS-DTW.

We show a few qualitative alignment results to compare different approaches in Figure 7. We note that the flexible methods are able to deal with the stylistic variations in the query video and the retrieved mocap sequence, leading to a better temporal alignment between the two sequences.

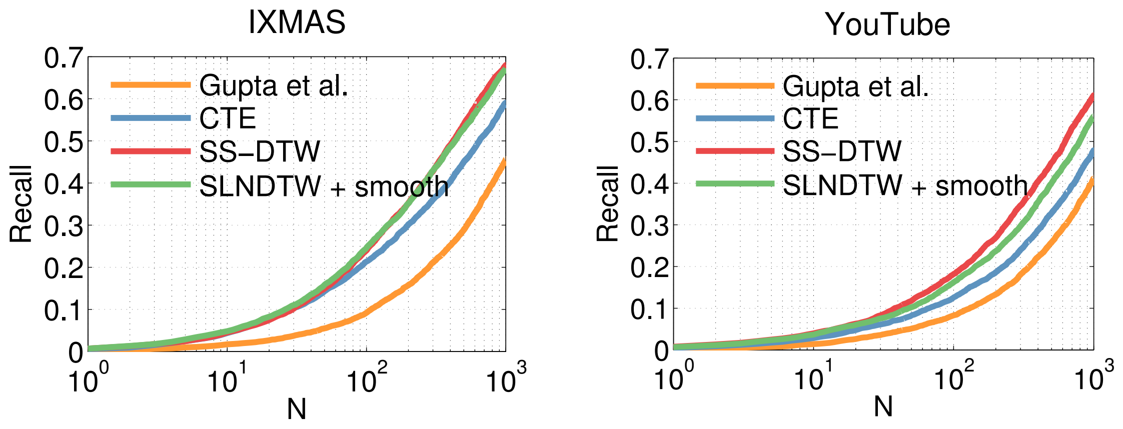


Figure 6: Recall on the *localization modality* of the video-based mocap retrieval benchmark, averaged over all action types. SS-DTW and SLNDTW (+smooth) perform similarly for both IXMAS and YouTube query sets. Gupta *et al.* [8] use CTE with trajectory-only descriptors, while we use pose-and-trajectory descriptors for both CTE and DTW-based methods.

				Gupta *et al.* [8]	CTE			SS-DTW	SLNDTW	+smooth
#	Action	# of ex.	Chance		Traj (T)	Pose (P)	P+T	P+T	P+T	P+T
1	Sit down	30	0.015	0.038	0.041	0.132	0.126	**0.169**	0.139	0.155
2	Get up	56	0.028	0.076	0.069	0.172	0.175	0.203	0.194	**0.220**
3	Turn	194	0.098	0.148	0.147	0.196	0.201	0.195	0.221	**0.230**
4	Walk	739	0.373	0.598	0.611	0.568	0.663	0.585	**0.688**	0.669
5	Punch	13	0.007	0.032	0.021	0.030	0.048	**0.070**	0.064	0.059
6	Kick	23	0.012	0.017	0.023	0.017	0.023	**0.042**	0.029	0.027
7	Pick up	76	0.038	0.147	0.158	0.291	0.314	0.392	**0.438**	**0.438**
8	Throw oh.	17	0.009	0.015	0.018	0.027	0.022	**0.035**	0.023	0.027
	Overall mAP		0.072	0.134	0.136	0.179	0.196	0.211	0.225	**0.228**

Table 1: Per-class and overall mAP on the *detection modality* of the video-based mocap retrieval benchmark. We show an average performance using both IXMAS and YouTube queries. *# of ex.* is the number of files in the database containing the given action. *Chance* corresponds to the expected performance of uniformly random retrieval. We highlight the best value in each category with boldface and underline the second best value.

6. Discussion

We have presented an efficient method for flexible matching and distance computation between a short video query and a mocap sequence. We have also introduced a novel benchmark (V3dR) to evaluate the challenging task of video-based retrieval of human motion. V3dR provides frame-level action annotations for around 400 000 frames of the CMU-mocap dataset along with video queries with variety in viewpoints, realistic clothing and backgrounds. We hope that V3dR will encourage further research in this area.

Additionally, we have shown that the relational pose features, previously used for action recognition, are also effective for human motion retrieval. This finding allows us to use any state-of-the-art 2d pose detector trained on images to retrieve similar mocap sequences using videos without any additional training data.

Acknowledgements. This work was supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC) and the Institute for Computing, Information and Cognitive Systems (ICICS) at UBC, and enabled in part by WestGrid and Compute Canada.

References

[1] CMU motion capture database. http://mocap.cs.cmu.edu/. 2, 3

[2] X. Anguera and M. Ferrarons. Memory efficient subsequence dtw for query-by-example spoken term detection. In *Multimedia and Expo (ICME)*, 2013. 2

[3] C. Bahlmann, B. Haasdonk, and H. Burkhardt. Online handwriting recognition with support vector machines-a kernel approach. In *Frontiers in handwriting recognition*, 2002. 1

[4] M.-W. Chao, C.-H. Lin, J. Assa, and T.-Y. Lee. Human motion retrieval from hand-drawn sketch. *Visualization and Computer Graphics*, 18(5), 2012. 2

[5] M. G. Choi, K. Yang, T. Igarashi, J. Mitani, and J. Lee. Retrieval and visualization of human motion data via stick figures. *Computer Graphics Forum*, 31(7), 2012. 2

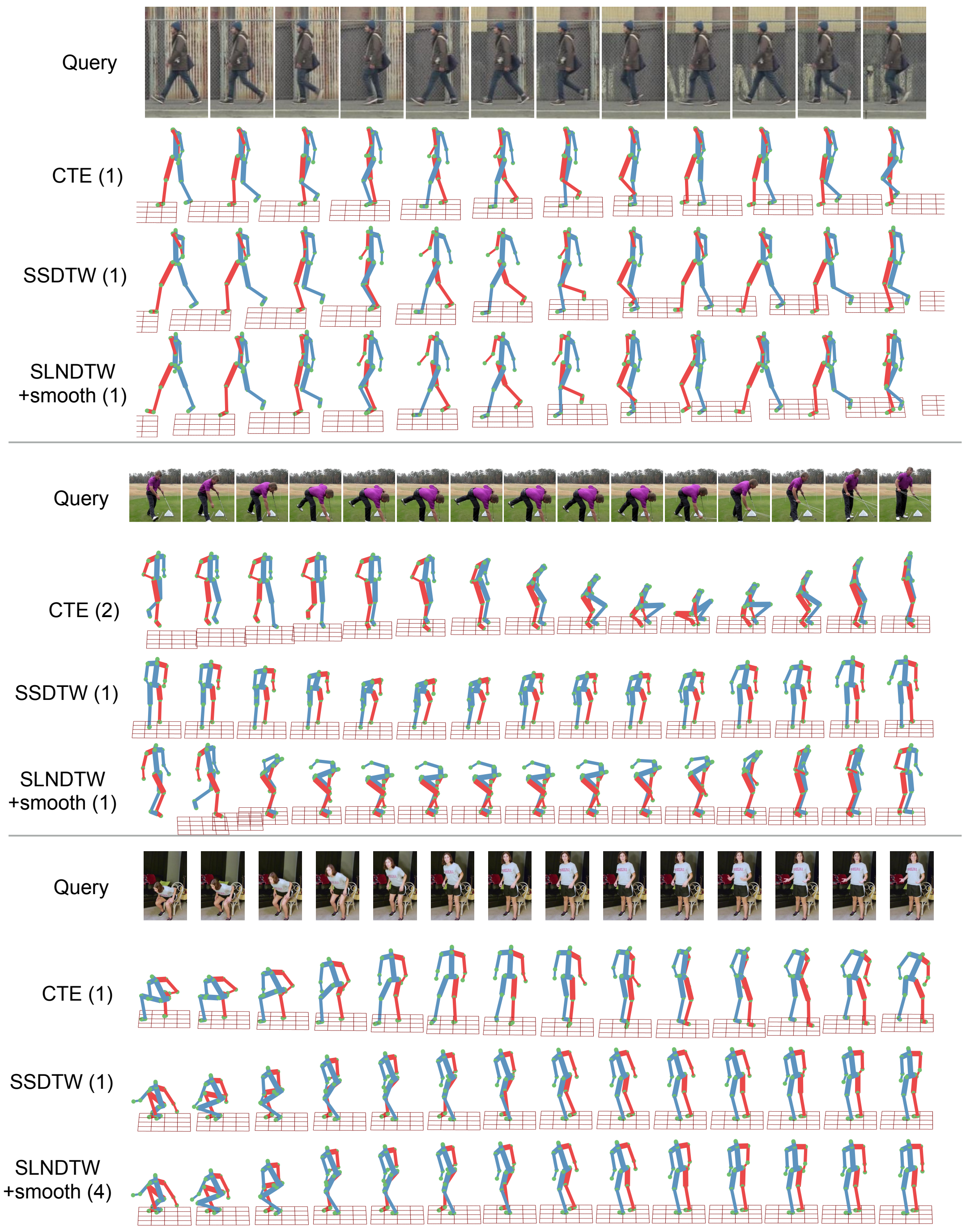


Figure 7: A few representative alignments for YouTube videos (best viewed in color). We show the query sequence at the top. The corresponding frames of the retrieved mocap sequences are shown below (right limbs are marked in red). For each retrieval algorithm we show the top ranked true-positive. The rank is shown in parentheses. **Top:** Repetitive and whole-body movements such as walking are relatively easy to match. All the algorithms perform well on this example. Note the matching is able to accurately capture the walk cycle as well as the right orientation (person's heading direction) in all cases. **Middle:** In this pick-up sequence, the flexible matching algorithms (SS-DTW and SLNDTW) are able to capture the bend down and get up movements. However, the rigid matching (CTE) only aligns with the final get-up movement and the initial frames do not match well. SLNDTW is also able to get the right pose and as well as the viewpoint. **Bottom:** Here, the person gets up and stands still for a few frame. All the algorithms match the get-up movement, but the CTE match turns and faces the opposite direction. In contrast, we get a much better alignment using the flexible matching techniques.

[6] T.-C. Feng, P. Gunawardane, J. Davis, and B. Jiang. Motion capture data retrieval using an artist's doll. In *ICPR*, 2008. 2

[7] T.-c. Fu, F.-l. Chung, R. Luk, and C.-m. Ng. Stock time series pattern matching: Template-based vs. rule-based approaches. *Engineering Applications of Artificial Intelligence*, 20(3), 2007. 1

[8] A. Gupta, J. Martinez, J. J. Little, and R. J. Woodham. 3d pose from motion for cross-view action recognition via non-linear circulant temporal encoding. In *CVPR*, 2014. 1, 2, 5, 6, 7

[9] C. Ionescu, D. Papava, V. Olaru, and C. Sminchisescu. Human3.6m: Large scale datasets and predictive methods for 3d human sensing in natural environments. *TPAMI*, 36(7), 2014. 2

[10] H. Jégou, M. Douze, and C. Schmid. Product quantization for nearest neighbor search. *TPAMI*, 33(1), 2011. 3

[11] H. Jhuang, J. Gall, S. Zuffi, C. Schmid, and M. J. Black. Towards understanding action recognition. In *ICCV*, 2013. 5

[12] M. Kapadia, I.-k. Chiang, T. Thomas, N. I. Badler, and J. T. Kider, Jr. Efficient motion retrieval in large motion databases. In *ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, 2013. 2

[13] L. Kovar and M. Gleicher. Automated extraction and parameterization of motions in large data sets. *TOG*, 23(3), 2004. 1

[14] A. Marzal and E. Vidal. Computation of normalized edit distance and applications. *PAMI*, 15(9), 1993. 5

[15] M. Müller. *Information retrieval for music and motion*, volume 2. Springer, 2007. 2, 4

[16] M. Müller, T. Röder, and M. Clausen. Efficient content-based retrieval of motion capture data. *TOG*, 24(3), 2005. 1

[17] M. Müller, T. Röder, M. Clausen, B. Eberhardt, B. Krüger, and A. Weber. Documentation mocap database HDM05. Technical Report CG-2007-2, Universität Bonn, June 2007. 2

[18] A. Muscariello, G. Gravier, and F. Bimbot. Variability tolerant audio motif discovery. In *Advances in Multimedia Modeling*. 2009. 2, 5, 6

[19] N. Numaguchi, A. Nakazawa, T. Shiratori, and J. K. Hodgins. A puppet interface for retrieval of motion capture data. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 2011. 2

[20] F. Perronnin, J. Sánchez, and T. Mensink. Improving the Fisher kernel for large-scale image classification. In *ECCV*, 2010. 5

[21] L. Rabiner and B.-H. Juang. *Fundamentals of Speech Recognition*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1993. 1, 4, 5

[22] L. Ren, G. Shakhnarovich, J. K. Hodgins, H. Pfister, and P. Viola. Learning silhouette features for control of human motion. *TOG*, 24(4), 2005. 1, 2

[23] J. Revaud, M. Douze, S. Cordelia, H. Jégou, et al. Event Retrieval in Large Video Collections with Circulant Temporal Encoding. In *CVPR*, 2013. 1, 6

[24] S. Salvador and P. Chan. Fastdtw: Toward accurate dynamic time warping in linear time and space. In *KDD workshop on mining temporal and sequential data*, 2004. 2, 5

[25] G. Shakhnarovich, P. Viola, and T. Darrell. Fast pose estimation with parameter-sensitive hashing. In *CVPR*, 2003. 2

[26] A. A. Smith, A. Vollrath, C. A. Bradfield, and M. Craven. Similarity queries for temporal toxicogenomic expression profiles. *PLoS Computational Biology*, 4(7), 2008. 2

[27] A. A. Smith, A. Vollrath, C. A. Bradfield, and M. Craven. Clustered alignments of gene-expression time series data. *Bioinformatics*, 25(12), 2009. 2

[28] V. Tuzcu and S. Nas. Dynamic time warping as a novel tool in pattern recognition of ecg changes in heart rhythm disturbances. In *International Conference on Systems, Man and Cybernetics*, 2005. 1

[29] H. Wang, A. Klaser, C. Schmid, and C.-L. Liu. Action recognition by dense trajectories. In *CVPR*, 2011. 5

[30] D. Weinland, R. Ronfard, and E. Boyer. Free viewpoint action recognition using motion history volumes. *CVIU*, 104(2), 2006. 3

[31] Y. Yang and D. Ramanan. Articulated human detection with flexible mixtures of parts. *TPAMI*, 35(12), 2013. 6

[32] A. Yao, J. Gall, and L. Gool. Coupled Action Recognition and Pose Estimation from Multiple Views. *IJCV*, 100(1), 2012. 3

[33] H. Yasin, B. Krüger, and A. Weber. Motion tracking, retrieval and 3d reconstruction from video. *IJMUE*, 9(2), 2014. 2

[34] T.-H. Yu, T.-K. Kim, and R. Cipolla. Unconstrained monocular 3d human pose estimation by action detection and cross-modality regression forest. In *CVPR*, 2013. 3

[35] Y. Yuan, Y.-P. P. Chen, S. Ni, A. G. Xu, L. Tang, M. Vingron, M. Somel, and P. Khaitovich. Development and application of a modified dynamic time warping algorithm (DTW-S) to analyses of primate brain expression time series. *BMC Bioinformatics*, 12(1), 2011. 2

[36] F. Zhou and F. de la Torre. Generalized canonical time warping. *TPAMI*, 38(2), 2016. 2