Learning Shift-Invariant Sparse Representation of Actions

Yi Li, Cornelia Fermuller, and Yiannis Aloimonos
Computer Vision Lab
University of Maryland, College Park, MD 20742
{liyi, fer, yiannis}@umiacs.umd.edu

Hui Ji
Department of Mathematics
National University of Singapore
matjh@nus.edu.sg

Abstract

A central problem in the analysis of motion capture (MoCap) data is how to decompose motion sequences into primitives. Ideally, a description in terms of primitives should facilitate the recognition, synthesis, and characterization of actions. We propose an unsupervised learning algorithm for automatically decomposing joint movements in human motion capture (MoCap) sequences into shift-invariant basis functions. Our formulation models the time series data of joint movements in actions as a sparse linear combination of short basis functions (snippets), which are executed (or "activated") at different positions in time. Given a set of MoCap sequences of different actions, our algorithm finds the decomposition of MoCap sequences in terms of basis functions and their activations in time. Using the tools of L_1 minimization, the procedure alternately solves two large convex minimizations: Given the basis functions, a variant of Orthogonal Matching Pursuit solves for the activations, and given the activations, the Split Bregman Algorithm solves for the basis functions. Experiments demonstrate the power of the decomposition in a number of applications, including action recognition, retrieval, MoCap data compression, and as a tool for classification in the diagnosis of Parkinson (a motion disorder disease).

1. Introduction

Interpreting human behavior is a newly emerging area that has attracted increasing attention in computer vision. One of the intellectual challenges in modeling human motion is to come up with formalisms for describing and recognizing human actions in motion capture (MoCap) sequences. Fundamentally, the primitives should assist the recognition, synthesis, and characterization of human actions. From this perspective, the formalism of the primitives is essential to action representation.

Human actions by their nature are sparse both in action space domain and time domain. They are sparse in action space, because different actions share similar movements on some joints, and also different joints share similar movements. They are sparse in the time domain, because we do not want much overlap of the individual movements on a single joint. These observations make the concept of shift-invariant sparse representation as the primitives of human actions very attractive, where shift invariant means that the output does not depend explicitly on time, *e.g.*, the same action can have multiple realizations at different times.

Let us get into more detail. We are given many MoCap sequences. The data from a motion capture suit are time series of three rotation angles each at a number of joints on the human body. Each of these sequences consists of a number of instances of different actions (where an instance of an action could be a step of a "running" sequence, or a single "kick", or "jump"). Our goal is to obtain from these action sequences a set of basis functions that could be used for approximating the entire set of the actions.

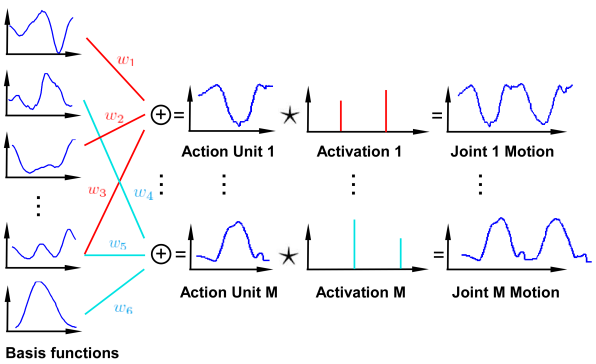


Figure 1. Modeling human motion in MoCap sequences using shift-invariant sparse representation. The short basis functions are sparsely selected and linearly combined to create action units for individual joints. The units may be shifted to different locations where multiple instances of the movement are realized. The time shift is modeled by the convolution (denoted by $\star$).

Our basis functions are chosen to be smooth functions and about the length of an instance of an action (Fig. 1). This enables us to achieve a useful underlying representation of different actions. The joint movements in an instance of an action are approximated by a sparse linear combina-

tion of basis functions ("Action Unit" in Fig. 1). To achieve a meaningful behavioral interpretation the weights are defined to be positive. Multiple instances of the same joint movement are realized by executing (or "activating") the linear combination of basis functions at different instances of time, but with different strength ("Activation" in Fig. 1). That is, all basis functions involved in the representation of a single joint are activated simultaneously. But different joints are activated separately.

Our action representation then is the weights of the basis functions along with their activations. Once these shift-invariant basis functions are learned, we can approximate any novel action sequence using a weighted combination of a number of these basis functions.

In our learning procedure, we solve for both the basis functions of the actions and the times when these functions are "activated". Solving them together would amount to a complicated non-convex optimization with a large number of variables. However, the optimization problem is convex in either the basis functions or the activations. Our method thus solves alternately for the two set of parameters. Recently developed L_1 minimization techniques allow us to solve these two problems effectively. Given a set of basis functions, a variant of Orthogonal Matching Pursuit [25] is used to obtain the activations by solving a non-negative L_1 minimization problem with a large number of variables. Given the activations, the Split Bregman Algorithm [14] is used to solve an L_1 regularized liner least square problem.

The characteristics of our decomposition approach are:

1. Our unsupervised algorithm learns a high-level sparse representation (the primitives) of action which allows recognizing actions in MoCap sequences effectively.

2. The shift-invariant modeling naturally handles the MoCap sequence composed of multiple instances of different actions.

3. The sparse activations explicitly express the coordination among different joints.

The rest of the paper is organized as follows. Sec. 2 discusses related work. Sec. 3 presents the algorithm for learning the basis functions. Sec. 4 summarizes an algorithm used for normalizing the length of MoCap sequences. Sec. 5 demonstrates the usefulness of our action representation on four applications, and Sec. 6 concludes the paper.

2. Related work

Finding motion primitives has been studied in a large body of work. D'Avella *et al.* [11] discovered that the muscles were activated together to perform actions. [9] applied non-negative matrix factorization to study torque patterns. [24] studied motion using a dynamical system. None of these models uses shift-invariant primitives.

A decomposition into shift-invariant features has been studied in acoustic signals classification ([3]). Shift invariant sparse coding [23] further improved the performance of classification. A major difference between their mathematical formulation and ours is that we enforce the weights to be positive, and the basis functions of individual joints to be be shifted coherently to realize an instance of an action.

L_1 minimization recently gained much attention with the emergence of compressive sensing [7] and has been applied frequently to image denoising [6], sparse representation of data [10], and for solving non-negative sparse-related problems [13]. Our approach involves solving an L_1 norm minimization in many variables. Although in principle, it is possible to solve an L_1 minimization problem by formulating it as a linear programming problem, such an approach is not efficient when many variables are involved. But recently a number of fast algorithms have been developed for approximating the optimal solution. For example, Basis Pursuit [8] solves the L_1 minimization by selecting the best bases. Orthogonal Matching Pursuit (OMP) [25] can reliably recover a signal with K nonzero entries given a reasonable number of random linear measurements of that signal. Alternatively, the Split Bregman Algorithm [14], approximates the optimal solution by iteratively solving efficiently a few simple sub-problems.

Temporal segmentation of human motion. A few studies proposed methods for breaking MoCap sequences into small action segments. Jenkins and Mataric [15] used a heuristic algorithm to partition human motion. Vecchio *et al.* [26], Bissacco [2] and Lu and Ferrier [19] assumed that human motion is ruled by autoregressive (AR) processes or state-space models and partitioned the sequences based on different model parameters. A comparison of partitioning algorithms in motor space can be found in [4].

Applications. The action basis could find direct application in action embodiment [22]. The segments are useful for action retrieval [12] and action classification [27], and can be used for compressing human motion [18].

3. Shift-invariant sparse modeling of actions

A MoCap sequence consists of the time series of three rotation angles each at a number of joints on the human body. In our approach, we approximate each time series by sparse linear combinations of shift-invariant sparse features.

For simplicity, we start our journey from the following example. A body joint rotation s ($1d$ time series) consists of the movements of multiple instances of the same action. Given N short basis functions b_i ($i = 1, 2, ...N$) which have the length about an instance of an action, we would like to approximate s as follows:

$$s \approx \sum_i a \star w_i b_i, \tag{1}$$

where a is the sparse activation for s, and $\star$ is the convolution operator (Fig. 1). The variables a and w_i are non-negative. Eq. 1 is equivalent to

$$\begin{aligned} s &\approx \sum_i w_i a \star b_i \\ &= \sum a_i \star b_i, \end{aligned} \tag{2}$$

where $a_i = w_i a$. This means we can model the shift of each individual basis function separately, with the additional constraint that all the activations a_i $(i = 1, 2, ...N)$ must have non-zero values at the same time when used for approximating s.

In the following formulation, we first discuss a solution for Eq. 2 in Sec. 3.1 and 3.2. The additional constraint is enforced when we solve the activations in Sec 3.3.1.

3.1. Problem formulation

Given a set of M $1d$ signals s_j, $j = 1, 2, ...M$, each of which is of length l and represents a time series of a joint movement, we want to approximate all s_j as the convolution between the activations and the basis functions, *i.e.*,

$$s_j = \sum_i a_i^j \star b_i + n_j, \tag{3}$$

where a_i^j $(j = 1, 2, ...M,\ i = 1, 2, ...N)$ are the sparse non-negative activations for the i^{th} basis function in the j^{th} signal, and n_j is the noise.

We enforce that the activations are sparse, and the basis functions are sparse in the Fourier domain. Therefore, the modeling poses the following L_1 regularized optimization problem:

$$\min_{(a_i^j, b_j)} \sum_j |s_j - \sum_i a_i^j \star b_j|_2 + \mu_1 \sum_{i,j} |a_i^j|_1 + \mu_2 \sum_i |F\hat{b_i}|_1, \tag{4}$$

where $|\cdot|_p$ is the L_p norm of the vector, F is the Fourier transform matrix, and $\hat{b_i}$ are the zero-padded b_i which are of length l.

Solving the activations and the basis functions together would amount to a non-convex optimization with a large number of variables. In Sec. 3.2, we re-formulate the problem in the frequency domain.

3.2. Formulating the problem in frequency domain

We show that the optimization problem is convex in either the basis functions or the activations. Therefore, a coordinate descent algorithm is used to alternately solve two large convex L_1 regularized problems.

The convolution in time domain is equivalent to the dot product in frequency domain. Therefore, Eq. 3 is equivalent to:

$$S_j \approx \sum_i A_i^j \cdot \hat{B}_i \tag{5}$$

where $\cdot$ is the pairwise multiplication operation, $A_i^j = F a_i^j$, and $\hat{B}_i = F\hat{b_i}$, respectively.

Denoting $\overline{X}$ as the square matrix whose diagonal is X, we have:

$$A_i^j \cdot \hat{B}_i = \overline{A_i^j}\hat{B}_i = \overline{\hat{B}_i} A_i^j \tag{6}$$

Therefore, Eq. 5 is equivalent to

$$\begin{aligned} S_j &\approx \begin{bmatrix} \overline{\hat{B_1}} & \dots & \overline{\hat{B_N}} \end{bmatrix} \begin{bmatrix} A_1^j \\ \vdots \\ A_N^j \end{bmatrix} \\ &= \begin{bmatrix} \overline{\hat{B_1}} & \dots & \overline{\hat{B_N}} \end{bmatrix} \begin{bmatrix} F & & \\ & \ddots & \\ & & F \end{bmatrix} \begin{bmatrix} a_1^j \\ \vdots \\ a_N^j \end{bmatrix} \\ &= \mathcal{B} a_j, \end{aligned} \tag{7}$$

where $a^j = [a_1^j; ...; a_N^j]^T$, and

$$\mathcal{B} = \begin{bmatrix} \overline{\hat{B_1}} & \dots & \overline{\hat{B_N}} \end{bmatrix} \begin{bmatrix} F & & \\ & \ddots & \\ & & F \end{bmatrix}.$$

Similarly, Eq. 5 can be rewritten as:

$$\begin{aligned} S_j &\approx \begin{bmatrix} \overline{A_1^j} & \dots & \overline{A_N^j} \end{bmatrix} \begin{bmatrix} \hat{B}_1 \\ \vdots \\ \hat{B}_N \end{bmatrix} \\ &= \begin{bmatrix} \overline{A_1^j} & \dots & \overline{A_N^j} \end{bmatrix} \begin{bmatrix} F & & \\ & \ddots & \\ & & F \end{bmatrix} \begin{bmatrix} \hat{b_1} \\ \vdots \\ \hat{b_N} \end{bmatrix} \\ &= \begin{bmatrix} \overline{A_1^j} & \dots & \overline{A_N^j} \end{bmatrix} \begin{bmatrix} F_l & & \\ & \ddots & \\ & & F_l \end{bmatrix} \begin{bmatrix} b_1 \\ \vdots \\ b_N \end{bmatrix} \\ &= \mathcal{A}^j \mathcal{F} b, \end{aligned} \tag{8}$$

where $\mathcal{A}^j = [\overline{A_1^j}, ..., \overline{A_N^j}]$, $b = [b_1; ...; b_N]^T$, the matrix F_l as the first l columns of the F, and

$$\mathcal{F} = \begin{bmatrix} F_l & & \\ & \ddots & \\ & & F_l \end{bmatrix}.$$

Let S, $\mathcal{A}$, a and b denote the concatenations of all possible S_j, $\mathcal{A}^j$, a^j and b_i in the column form. Eq. 4 is convex in either a or b, so we solve it alternately as two convex optimization problems.

Given the basis functions b in time domain, from Eq. 4 we obtain:

$$\min_a |S - \mathcal{B}a|_2 + \mu_1 |a|_1 \tag{9}$$

Eq. 9 is the sum of M independent subproblems, all of which are convex. We can approximate them separately.

Given the activations a, from Eq. 4 we obtain:

$$\min_b |S - \mathcal{A}\mathcal{F}b|_2 + \mu_2|\mathcal{F}b|_1 \tag{10}$$

Both Eq. 9 and 10 are convex. Therefore, the objective function Eq. 4 is always non-increasing using the updates. Eq. 9 is solved using the Orthogonal Matching Pursuit, and Eq. 10 is solved using the Split Bregman iterative algorithm. To avoid trivial results, we normalize the basis function in amplitude in each iteration.

3.3. Solving the problem using L_1 minimization

3.3.1 OMP for solving the activations

We use a variant of the Orthogonal Matching Pursuit (OMP) to solve Eq. 9. The variant amounts to implementing Orthogonal Matching Pursuit in a batch mode.

Orthogonal Matching Pursuit is a greedy algorithm. It progressively picks the new basis which minimizes the residual. The major advantages of this algorithm are its ease of implementation and its speed. This approach can easily be extended to related problems, such as finding non-negative bases [13].

In our modeling, a single body joint movement is a sparse combination of the basis functions, with the weights non-negative and the additional constraint that the activations must be coherent. This is solved as follows: Given the movement ($1d$ time series) of a joint, we progressively pick the new basis at the locations found in the previous steps, and minimize the residual of all the time series. We enforce the solution to be positive by checking which basis to choose and checking the weights found in the least-squares minimization.

During the optimization, a sparse subset of basis functions is automatically selected. In our implementation, we allow a maximum of 4 basis functions at a single activation at one joint, with the total number of basis functions being 15. This makes it easier to compare the weights of the same joint in action retrieval and classification.

3.3.2 Split Bregman Algorithm for solving the bases

As defined in the literature, the Split Bregman Iterative Algorithm is applied to the following problem

$$\min_u J(u) + H(u), \tag{11}$$

with $u \in R^n$, $J(u)$ is the L_1 norm of a function of u and is continuous but not differentiable function, and $H(u)$ is the L_2 norm of a function of u and is continuous differentiable. In our case, $J(u) = |\mathcal{F}b|_1$ and $H(u)$ is the L_2 norm of the approximation error.

By introducing $|d - \phi u|_2$ and $E(u, d) = |d|_1 + H(u)$, we rewrite Eq. 11 as

$$\min_{(u,d)} E(u, d) + \lambda/2|d - \phi u|_2. \tag{12}$$

The solution is given by iteratively updating the following three equations:

$$u^{k+1} \leftarrow arg\min_u H(u) + \lambda/2|d^k - \phi u - p^k|_2 \tag{13}$$

$$d^{k+1} \leftarrow arg\min_d |d|_1 + \lambda/2|d - \phi u^{k+1} - p^k|_2 \tag{14}$$

$$p^{k+1} \leftarrow p^k + \phi u^{k+1} - d^{k+1}$$

This "splits" Eq. 12 into the subproblems. Eq. 13 is a 2^{nd} order continuous differential function that can be solved efficiently. Eq. 14 is solved by shrinkage operation[1]. By the alternately update in the Split Bregman Algorithm, we obtain the optimal sparse solution for Eq. 10.

4. Preprocessing: normalization for handing actions with various speeds

It is important to handle action sequences of different speeds. For this we use our action segmentation algorithm. This algorithm breaks an action sequence into action segments. We then compute the average length of the action segments and use it to normalize the sequence.

Our goal is to find the discontinuities in the 3^{rd} order derivative of the time series. Motivation for this approach comes from the work of d'Avella *et al.* [11], who found that the change of the muscle force indicates the time of action change, and the change of muscle force is proportional to the 3^{rd} order derivative of the time series.

Our algorithm partitions a MoCap sequences by minimizing the sum of the pairwise distances of the envelope extrema of the different joints. In this algorithm, we use the quaternion representation for rotation.

The quaternion series of a certain joint is a 4D vector

$$\mathbf{X}(t) = [x_1(t), x_2(t), x_3(t), x_4(t)]^T \tag{15}$$

The jerk of $\mathbf{X}(t)$ is computed as

$$J(t) = |\frac{d^3(\mathbf{X}(t))}{dt^3}|_2 \tag{16}$$

To minimize the error in computing the derivative, we smooth the data using a low pass filter.

To measure the jerk better, we compute the jerk envelope

$$\text{Env}(t) = |\text{Hilbert}(J(t))|_2 \tag{17}$$

for every joint, where Hilbert$(\cdot)$ is the Hilbert transform. This is a standard approach for computing the signal envelope [5]. Then we process Hilbert$(\cdot)$ using as a low pass

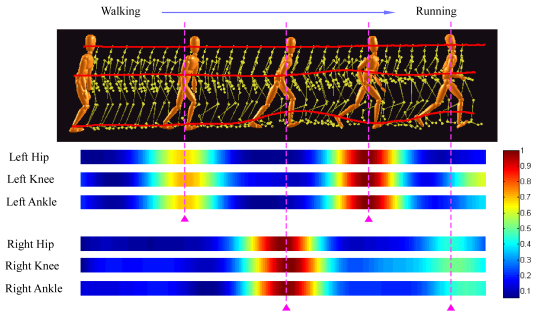


Figure 2. Estimating the average action speed by measuring the action discontinuities. A sequence "walking to running" from the University of Bonn dataset [20] is shown.The poses corresponding to the discontinuities are displayed as mannequin. The trajectories of the head, the left elbow, and the right ankle are drawn in red.

filter a Butterworth filter, and compute the envelope's extrema of the filtered Env(t) for every joint.

Fig. 2 gives an example of the speed-varying action "walking to running" from the University of Bonn dataset [20]. Here, we show the jerk envelopes of the joints "Hip", "Knee", and "Ankle" of both legs, respectively. The envelopes are color coded and normalized in magnitude. The breaking poses (mannequins) can be selected by finding the optimal "alignment" of the envelope extrema of the different joints (the purple dash lines in Fig. 2).

The alignment can be solved as an optimization problem using the envelope extrema of all joints. It is applied until the average pairwise distances is larger than a threshold ϵ.

5. Experiments

The following four experiments demonstrate the usefulness of our representation: First, we present the basis functions learned from our own dataset in Sec. 5.2. Second, we show that our basis functions are well suited for approximating novel actions. This allows us to substantially compress novel MoCap data (Sec. 5.3). Third, the experiments in Sec. 5.4 demonstrate that using only the magnitude of the activations, action retrieval and classification can be solved effectively. Finally, we show that the activations and the fitting error alone are very useful for motion related disease diagnosis, thus demonstrating the intuitive nature of the description (Sec. 5.5).

We used three datasets, our own, the Univ. Bonn dataset, and the CMU MoCap dataset [16]. Our own dataset consists of 55 different actions, which were captured with the MOVEN motion capture suit [21] at 100 fps. Each action sequence consists of at least 6 repetitions of the same action. Fig. 3 shows some of the actions in our dataset.

The convergence speed primarily depends on OMP and Bregamn algorithm. OMP is a greedy algorithm that takes

[1] $shinkage(x, y) = sgn(x)\max(|x| - y, 0)$.

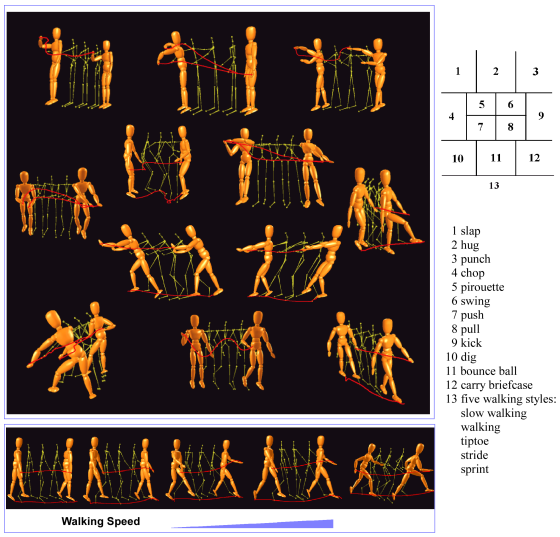


Figure 3. 17 out of the 55 actions in our data set. The rendering is as follows: poses corresponding to the discontinuities are displayed as mannequins; the transitions in between are illustrated by wire-frames; the trajectories of some joints are drawn in red.

linear time, and Bregman is proved to be very efficient for many problems that are difficult by other means [14]. Thus, our algorithm is very efficient. It takes only 10-15 iterations before convergence (≈5 mins in Matlab for our dataset).

5.1. Parameters

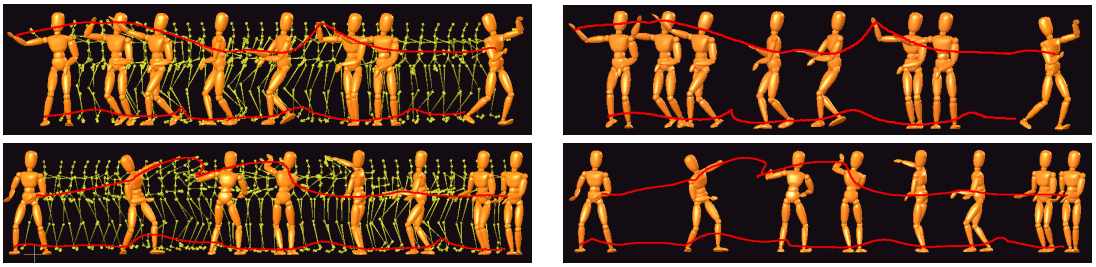
μ_1 and μ_2 in Eq. 4 determine the balance between the fidelity of the object function and the sparsity of the variables in the optimization. In all the experiments, they were both set to $\frac{1}{2}$. λ in the Split Bregman Algorithm (Eq. 12) is the parameter for penalizing the auxiliary variable. It was set to 1000 in our experiments. The length of the basis function was set to the frame rate of our MoCap suit (100). In our experiments, the normalization speeds up the convergence of the algorithm, therefore, MoCap sequences were also approximately normalized to 100 samples per action instance.

The initialization of Eq. 4 is randomly generated. This optimization is a large non-convex problem, and a common practice is to have a random guess at the beginning.

5.2. Learning the basis functions

Fig.3 visualizes 17 out of the 55 actions in our dataset. First, we applied our normalization algorithm. As found by visual inspection the discontinuities in the action sequences estimated by this algorithm correspond to the intuitive poses separating actions.

After normalization, the action decomposition algorithm is applied to the action sequences. Fig 5a shows the fifteen basis functions learned by the algorithm. Each column is a color-coded basis function. In our modeling, individual joints are described by four basis functions, from the above



(a) (b)

Figure 4. Side by side comparison between original motion frames (a) and reconstructed motion (b) using the estimated basis functions, demonstrated on two "salsa" sequences from the CMU dataset [16].

set of fifteen learned basis functions. Fig 5b shows the basis functions used by the individual joints. We can see that different joints may share the same basis functions. Please note that the combination of the basis functions that composes joint movements. Different joints may have different combinations for an action, therefore, it is better to plot the functions individually.

Despite these very small numbers, the approximation is very good. The first column in Table 1 shows that the error residual in the approximation was very small. The residual was measured by the total fitting error divided by the number of frames and the number of joints in the dataset. On average, our representation approximates the training sequences with only 2.36 degree per joint in every frame.

The result shows that the primitives are effective and compact representations of the actions in the datasets.

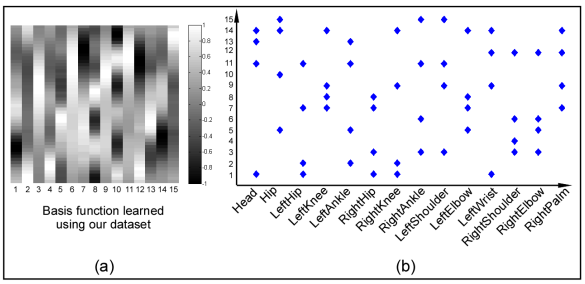


Figure 5. The basis functions learned by the algorithm (a) and their usage for individual joints (b) (denoted by the blue diamonds)

Table 1. Average fitting error for different MoCap sequences using the basis functions learned in Sec. 5.2.

Seq	Training	Walking,Bonn	Running, Bonn
Error	2.36°	3.18°	3.56°

5.3. Motion approximation and compression

We use the basis functions to approximate novel actions. This further leads to effective compression of MoCap data. In this experiment, the novel sequences were first normalized, and Eq. 9 is then used to compute the activations and approximate the sequences. An averaging filter is used to handle the possible discontinuities between actions.

A useful representation of action should have the generalization capability of expressing unseen actions. First, we used the basis function learned from our dataset to approximate two sets of the sequences in the Bonn dataset, which were captured by an optical motion capture suit by different subjects. We then measured the fitting error. As shown in the 2^{nd} and 3^{nd} columns in Table 1, they are very small.

Comparing lossy compression results objectively is very difficult. As pointed out by [1], the fitting error may not be a good predictor of visual quality. Therefore, the subjective judgments were used. Fig. 4b visualizes the approximation using two "salsa" dances in the CMU dataset. We can see that the poses of reconstructed movement (Fig. 4b) approximate those in the original sequences (Fig. 4b) very well. This side by side comparison shows that the shift-invariant decomposition effectively handles the complicated novel actions.

Another advantage of our decomposition approach is that it leads to high compression rate. To effectively compress MoCap sequences composed of arbitrary actions is very useful both for storage and for visualization, but it is also a challenging problem. In our approaches, a joint movement that has 100 data samples can be described by only four coefficients. Thus, we achieved approximately $25:1$ compression rate by default[2]. Table 2 compares the compression rate of some algorithms on CMU dataset.

Table 2. Comparison of different MoCap data compression algorithms. (*): the compression rate without quantizing the weights. (**): the compression rate with weight quantization. The ratios of the other algorithms are copied from [1]

Algorithm	Ours	Arikan	Wavelet	Zip
Ratio	19:1(*) 37:1(**)	30:1	6:1	1.4:1

We archived competitive compression rate compared to the state-of-the art algorithm. More importantly, the primitive-based compression is fundamentally invariant to

[2]For complicated actions, we may use more activations to approximate the time series, based on the normalization ratio. In addition, a small amount of overhead is required (*e.g.*, storing the scaling factor and the basis functions).

the frame rate. A major difference between the our approach and previous approaches is that we explicitly model the human actions. Therefore, the change in frame rate only changes the number of samples in the basis functions, but not the activation positions in time.

5.4. Action retrieval and classification

In the following experiments, we demonstrate the usefulness of our description for action classification and retrieval. First, our preprocessing algorithm breaks the action sequences into action segments. Each segment is treated as one complete action. Then, we decompose every single action using Eq. 9 allowing for only 1 activation. Finally, action retrieval and classification can be solved effectively using only the magnitude of the activations as the weights.

We considered it more helpful to provide the intuition of the usefulness of the representation using simple Euclidean distance and a neireast neighborhood classifier. This demonstrates how much the representation contributes to the retrieval and classification, without tuning parameters in a sophisticated classifier. Therefore, we chose to compare our representation to decomposition methods.

5.4.1 Segment-based action retrieval

We compared our algorithm with the Sparse Principal Component Analysis [10] algorithm and the Principal Component Analysis algorithm. For both algorithms the segments are normalized to have the same length.

Retrieval is evaluated on our dataset using the so-called Bullseye test [17]. 6 segments per action sequence were selected[3]. A leave-one-out trial was performed for every segment. The retrieval rate is ratios of the correct hits in top 12 candidates for all trials.

The performance of the three algorithms is shown in Table 3. Our algorithms achieved higher accuracy (86.07%) in the Bullseye test. This indicates that the repeated action segments in an action sequence have similar representation. The result demonstrates that our decomposition algorithm has the power of finding the similar movements.

Table 3. Performance comparison (Bullseye) of action retrieval on the segments of our dataset. Three algorithms, namely Sparse PCA, PCA and our algorithm, were used in the comparison. The segments were normalized for Sparse PCA and PCA.

Algorithm	Ours	Sparse PCA	PCA
Accuracy	86.07%	82.64%	78.87%

5.4.2 Segment-based action classification

We classify actions performed by different subjects. Four actions ("walking", "marching", "running", and "salsa") from the CMU dataset, were used in the experiment.

[3]For action sequences which had a larger number of segments, we randomly selected 6 segments

We compared our algorithm with the Sparse Principal Component Analysis algorithm and the Principal Component Analysis algorithm. To demonstrate the usefulness of the weights, we chose a very simple k-nearest neighborhood (kNN, $k = 3$) classifier. For each partitioning algorithm, we randomly selected 50% of the estimated segments in each action category as the training samples, and used the remaining as the test samples. Figs. 6a-c show the confusion matrices of the classification using the coefficients obtained by our algorithm, the Sparse PCA and the PCA algorithm.

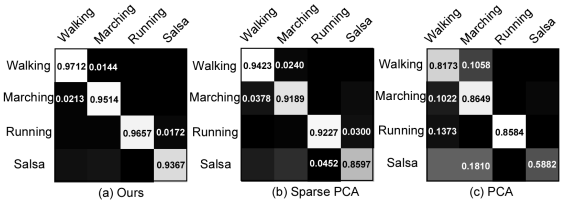


Figure 6. Action classification. Four actions, "walking", "marching", "running", and "salsa" from the CMU dataset, were used in the experiment. A very simple k-nearest neighborhood (kNN, $k = 3$) classifier was chosen. (a)-(c) show the confusion matrices of the classifier using the weights of the proposed algorithm, the Sparse PCA algorith, and the PCA, respectively. For each algorithm, 50% of the estimated segments in each action category were randomly selected as the training samples and the remaining as the test samples.

Results show that our representation gives the best classification performance. This demonstrate that our shift-invariant representation models the nature of the human action, and the sparse linear decomposition facilitates the performance of classification.

5.5. Motion disorder diagnosis

The activations are a natural measurement for describing body coordination. If the activations for different joints are not well aligned, the subject might have a problem in controlling her/his motions. Another measure is the approximation error.

We demonstrate the applicability of the basis functions in modeling the Parkinson motion disorder, which is characterized by degenerative muscle movements. The primary symptoms are the results of decreased coordination caused by insufficient control. This problem is highly difficult because the correct modeling for the coordination among different body parts is challenging.

In this experiment, we captured the MoCap data for four patients diagnosed with the PD disease and four healthy controls (Table. 4). Fig. 7 shows the scenario when the experiments were performed. Subjects were asked to perform a number of actions repeatedly.

For this application, we learned the basis functions and the activations for each subject individually. Three common

actions, "Finger To Nose", "Catching a Tennis Ball", and "Bread Cutting", were recorded. Figs. 8a-c show the plot of the activation alignment score and the average approximation error. The alignment score between two sequences is the zero-mean standard deviation of the differences between corresponding elements. The activation alignment score is defined as the largest value of the pairwise alignment scores.

Fig. 8d shows the chart for classifying the patients. As can be seen the two measurements are sufficient to separate controls from patients. Referring to 8a-c, the data points are well separated.

The diagnosis shows that the activations in our decomposition approach characterize the underlying rhythm in the parts of the bodies. This suggests that our approach is well suited for further understanding the principles of coordinated actions.

Figure 7. Collecting Parkinson Disease data.

Table 4. Parkinson disease patients' age information. The disease level is measured by the Hoehn and Yahr scale which ranges from 1-5 (shown in parentheses).

Controls	Patients
4 healthy subjects	63(2.5), 63(2.5), 60(2.5), 60(3)

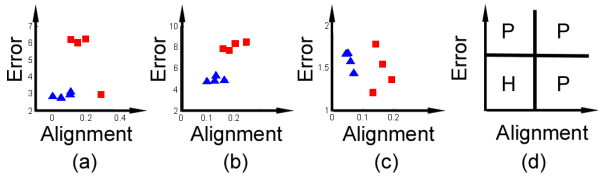


Figure 8. Parkinson disease diagnosis by measuring the alignment and the average approximation error. From left to right, the results for "Finger To Nose", "Catching a Tennis Ball", "Bread Cutting", and the diagnosis chart, respectively. In a)-c), blue triangles denote the healthy controls, and red squares denote the patients. d) suggests a chart for diagnosis. P: patient. H: healthy control.

6. Conclusion

This paper presented an algorithm for finding basic primitives to represent human motion data. Body movements in MoCap sequences are decomposed into the shift-invariant basis functions and their activations. The decomposition is solved by alternately updating two large convex problems using L_1 minimization techniques. Experiments show that the compact representation is effective for motion approximation, MoCap data compression, action retrieval, and classification with application to disease diagnosis.

References

[1] O. Arikan. Compression of motion capture databases. *ACM Trans. Graph.*, 25(3):890–897, 2006. 6

[2] A. Bissacco. Modeling and learning contact dynamics in human motion. In *CVPR'05*, pages 421–428, 2005. 2

[3] T. Blumensath and M. E. Davies. Sparse and shift-invariant representations of music. *IEEE Transactions on Audio, Speech & Language Processing*, 14(1):50–57, 2006. 2

[4] D. Bouchard. *Automated Motion Capture Segmentation using Laban Movement Analysis*. PhD thesis, University of Pennsylvania, 2008. 2

[5] R. Bracewell. *The Fourier Transform & Its Applications*. McGraw-Hill Science, June 1999. 4

[6] J. Cai, H. Ji, C. Liu, and Z. Shen. Blind motion deblurring from a single image using sparse approximation. In *CVPR'09*, pages 104–111, 2009. 2

[7] E. Candes. Compressive sampling. *Int. Congress of Mathematics*, pages 1433–1452, 2006. 2

[8] S. S. Chen, D. L. Donoho, and M. A. Saunders. Atomic decomposition by basis pursuit. *SIAM Rev.*, 43(1):129–159, 2001. 2

[9] M. Chhabra and R. Jacobs. Properties of synergies arising from a theory of optimal motor behavior. *Neural Comput.*, 18(10):2320–2342, 2006. 2

[10] A. D'aspremont, L. E. Ghaoui, M. I. Jordan, and G. R. G. Lanckriet. A direct formulation for sparse pca using semidefinite programming. *SIAM Review*, 49(3):434–448, 2007. 2, 7

[11] A. d'Avella, P. Saltiel, and E. Bizzi. Combinations of muscle synergies in the construction of a natural motor behavior. *Nature Neuroscience*, 6:300–308, 2003. 2, 4

[12] Z. Deng, Q. Gu, and Q. Li. Perceptually consistent example-based human motion retrieval. In *I3D '09*, pages 191–198, 2009. 2

[13] D. L. Donoho and J. Tanner. Sparse nonnegative solution of underdetermined linear equations by linear programming. *In Proc of NAS*, 102(27):9446–9451, 2005. 2, 4

[14] T. Goldstein and S. Osher. The split bregman method for l1-regularized problems. *SIAM J. on Imaging Sciences*, 2(2):323–343, 2009. 2, 5

[15] O. Jenkins and M. Mataric. Deriving action and behavior primitives from human motion data. In *IROS'02*, pages 2551–2556. 2

[16] C. G. Lab. http://mocap.cs.cmu.edu. 5, 6

[17] H. Ling and D. W. Jacobs. Shape classification using the inner-distance. *IEEE Trans. Pattern Anal. Mach. Intell.*, 29(2):286–299, 2007. 7

[18] G. Liu and L. McMillan. Segment-based human motion compression. In *SCA '06*, pages 127–135. 2

[19] C. Lu and N. Ferrier. Repetitive motion analysis: Segmentation and event classification. *IEEE Trans. Pattern Anal. Machine Intell.*, 26(2):258–263, 2004. 2

[20] 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. 5

[21] X. MVN. http://www.moven.com. 5

[22] N. Pollard, J. Hodgins, M. Riley, and C. Atkeson. Adapting human motion for the control of a humanoid robot. In *ICRA '02*. 2

[23] R. G. R. Raina, H. Kwong, and A. Y. Ng. Shift-invariant sparse coding for audio classification. In *UAI*, 2007. 2

[24] S. Schaal, P. Mohajerian, and A. Ijspeert. Dynamics systems vs. optimal control a unifying view. *Progress in Brain Research*, 165:425–445, 2007. 2

[25] J. A. Tropp and A. C. Gilbert. Signal recovery from random measurements via orthogonal matching pursuit. *IEEE Transactions on Information Theory*, 53(12):4655–4666, 2007. 2

[26] D. Vecchio, R. Murray, and P. Perona. Decomposition of human motion into dynamics based primitives with application to drawing tasks. *Automatica*, 39:2085–2098, 2003. 2

[27] A. Yang, R. Jafari, S. Sastry, and R. Bajcsy. Distributed recognition of human actions using wearable motion sensor networks. *Ambient Intelligence and Smart Environments*, 2009. 2