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 $\ast$).

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-
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 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 $b_i$ ($i = 1, 2, \ldots, N$) which have the length about an instance of an action, we would like to approximate $s$ as follows:

$$s \approx \sum_i a_i \star w_i b_i,$$

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

$$s \approx \sum_i w_i a \ast b_i = \sum_i a_i \ast b_i,$$

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 $a_i$ ($i = 1, 2, \ldots N$) 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 $s_j$, $j = 1, 2, \ldots M$, 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.,

$$s_j = \sum_i a^j_i \ast b_i + n_j,$$

where $a^j_i$ ($j = 1, 2, \ldots M$, $i = 1, 2, \ldots N$) 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:

$$\min_{(a^j_i , b_i)} \sum_j |s_j - \sum_i a^j_i \ast b_i|_2 + \mu_1 \sum_i |a^j_i|_1 + \mu_2 \sum_i |Fb_i|_1,$$

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:

$$S_j \approx \sum_i A^j_i \ast \hat{B}_i,$$

where $\cdot$ is the pairwise multiplication operation, $A^j_i = F a^j_i$, and $\hat{B}_i = F b_i$, respectively.

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

$$A^j_i \ast \hat{B}_i = \hat{X} A^j_i \hat{B}_i = \hat{X} A^j_i$$

Therefore, Eq. 5 is equivalent to

$$S_j \approx \left[ \begin{array}{c} \hat{B}_1 \\ \vdots \\ \hat{B}_N \end{array} \right] \left[ \begin{array}{c} A^j_1 \\ \vdots \\ A^j_N \end{array} \right] = \left[ \begin{array}{c} \hat{B}_1 \\ \vdots \\ \hat{B}_N \end{array} \right] F \left[ \begin{array}{c} a^j_1 \\ \vdots \\ a^j_N \end{array} \right] = \mathbf{B} a_j,$$

where $a^j = [a^j_1; \ldots; a^j_N]^T$, and

$$\mathbf{B} = \left[ \begin{array}{c} \hat{B}_1 \\ \vdots \\ \hat{B}_N \end{array} \right] \left[ \begin{array}{c} F \\ \vdots \\ F \end{array} \right].$$

Similarly, Eq. 5 can be rewritten as:

$$S_j \approx \left[ \begin{array}{c} \hat{A}^j_1 \\ \vdots \\ \hat{A}^j_N \end{array} \right] B_N = \left[ \begin{array}{c} \hat{A}^j_1 \\ \vdots \\ \hat{A}^j_N \end{array} \right] F b_j = \hat{A}^j F b_j,$$

where $\hat{A}^j = [\hat{A}^j_1, \ldots, \hat{A}^j_N]$, $b = [b_1; \ldots; b_N]^T$, the matrix $F$ as the first $l$ columns of the $F$, and

$$F = \left[ \begin{array}{c} F_1 \\ \vdots \\ F_l \end{array} \right].$$

Let $S$, $A$, $a$ and $b$ denote the concatenations of all possible $S_j$, $A^j$, $a^j$ 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 $b_i$ in time domain, from Eq. 4 we obtain:

$$\min_a |S - \mathbf{B} a|_2 + \mu_1 |a|_1$$

(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:

$$
\min_{b} |S - AFb|_2 + \mu_2|Fb|_1 \quad (10)
$$

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

$$
\min_{u} J(u) + H(u), \quad (11)
$$

with $u \in \mathbb{R}^n$, $J(u)$ 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, $J(u) = |Fb|_1$ 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

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

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

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

$$
d^{k+1} \leftarrow \arg \min_{d} |d|_1 + \lambda/2|d - \phi u^{k+1} - p^k|_2 \quad (14)
$$

$$
p^{k+1} \leftarrow p^k + \phi u^{k+1} - d^{k+1} \quad (15)
$$

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. 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 sequences of a certain joint is a 4D vector

$$
X(t) = [x_1(t), x_2(t), x_3(t), x_4(t)]^T \quad (15)
$$

The jerk of $X(t)$ is computed as

$$
J(t) = \left| \frac{d^3X(t)}{dt^3} \right|_2 \quad (16)
$$

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

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

for every joint, where $\text{Hilbert}(\cdot)$ is the Hilbert transform. This is a standard approach for computing the signal envelope [5]. Then we process $\text{Hilbert}(\cdot)$ using as a low pass
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 Bregman algorithm. OMP is a greedy algorithm that takes 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

μ₁ and μ₂ 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
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 $2.36^\circ$ degree per joint in every frame.

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

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. Table 2 compares the compression rate of some algorithms on CMU dataset.

![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].](image)

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

<table>
<thead>
<tr>
<th>Seq</th>
<th>Training Error</th>
<th>Walking, Bonn Error</th>
<th>Running, Bonn Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>Error</td>
<td>$2.36^\circ$</td>
<td>$3.18^\circ$</td>
<td>$3.56^\circ$</td>
</tr>
</tbody>
</table>

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. Table 2 compares the compression rate of some algorithms on CMU dataset.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Ours</th>
<th>Arikan</th>
<th>Wavelet</th>
<th>Zip</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ratio</td>
<td>$19:1(*)$</td>
<td>$37:1(**)$</td>
<td>$30:1$</td>
<td>$6:1$</td>
</tr>
</tbody>
</table>

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

---

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).

---
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 ($k$NN, $k = 3$) 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 algorithm, and the PCA algorithm.

![Figure 6. Action classification. Four actions, “walking”, “marching”, “running”, and “salsa” from the CMU dataset, were used in the experiment.](image)

Results show that our representation gives the best classification performance. This demonstrates 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).

<table>
<thead>
<tr>
<th>Controls</th>
<th>Patients</th>
</tr>
</thead>
<tbody>
<tr>
<td>4 healthy subjects</td>
<td>63(2.5), 63(2.5), 60(2.5), 60(3)</td>
</tr>
</tbody>
</table>

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