

A Novel Indexing Approach for Efficient and Fast Similarity Search of Captured Motions

Chuanjun Li and B. Prabhakaran

Department of Computer Science,
University of Texas at Dallas, Richardson, TX 75083
{chuanjun, praba}@utdallas.edu

Abstract. Indexing of motion data is important for quickly searching similar motions for sign language recognition and gait analysis and rehabilitation. This paper proposes a simple and efficient tree structure for indexing motion data with dozens of attributes. Feature vectors are extracted for indexing by using singular value decomposition (SVD) properties of motion data matrices. By having similar motions with large variations indexed together, searching for similar motions of a query needs only one node traversal at each tree level, and only one feature needs to be considered at one tree level. Experiments show that the majority of irrelevant motions can be pruned while retrieving all similar motions, and one traversal of the indexing tree takes only several microseconds with the existence of motion variations.

1 Introduction

Continuous motion data can be generated by many real-time and off-line applications in life sciences and animations, and can be employed for gesture recognition, gait analysis and rehabilitation, sports performance, film and video games [8]. To decide whether a motion segment in a motion stream is a known motion in a large motion database, or to recognize motions in the continuous motion data, not only is a motion similarity measure needed [5], but also an efficient and fast pruning algorithm is necessary. The pruning algorithm should prune most impossible motions in a large database for a motion query in real time. To prune motions efficiently and fast needs to address several challenges:

- Datasets of motions have multiple attributes. Each attribute describes the angular values or coordinates of a joint of the motion *subject*, and dozens of attributes are needed to capture a complete subject motion.
- Datasets of motions are high dimensional and even similar motions can have different dimensions. One dimension is for one sampling of all attributes, and every motion can have different durations and thus different dimensions.

Due to these issues, direct indexing of motion data is difficult and inefficient.

This paper proposes a new method for indexing motion data with dozens of attributes. The feature vectors are extracted by obtaining the equal-length

dominating vectors from singular value decompositions (SVD) of motion data and by reducing vector dimensionalities. Corresponding feature values of all motion patterns are partitioned into several intervals. Motion or feature vector IDs are inserted into a tree of feature intervals by using the corresponding feature values. To take into consideration motion variations, a feature ID is allowed to be inserted into multiple neighboring feature intervals. Hence a feature vector ID can be in multiple leaf nodes instead of in only one leaf node. Searching for possible similar motions of a query needs only one node traversal at each tree level and takes only several microseconds.

2 Related Work

Equal length multi-attribute sequences are considered in [2]. A CS-Index structure is proposed for shift and scale transformations. In [4], multi-attribute sequences are partitioned into subsequences, each of which is contained in a Minimum Bounding Rectangle (MBR). Every MBR is indexed and stored into a database by using an R-tree or any of its variants.

Dynamic time warping (DTW) and longest common subsequence (LCSS) are extended for similarity measures of multi-attribute data in [9]. Before the exact LCSS or DTW is performed, sequences are segmented into MBRs to be stored in an R-tree. Based on the MBR intersections, similarity estimates are computed to prune irrelevant sequences.

Attributes of the data indexed in the previous work are less than ten. In contrast, our proposed indexing structure can handle dozens or hundreds of data attributes without loss of good performances. This work proposes a novel indexing approach which is different from that in [6], making it possible to search the indexing tree for similar motions in only several microseconds.

3 Geometric Structures Revealed by SVD

In this section, we give the definition and geometric interpolation of SVD for its application to the indexing of multi-attribute motion data.

SVD exposes the geometric structure of a matrix A . If the multi-dimensional row vectors or points in A have different variances along different directions and columns of A have zero means, the SVD of matrix A can find the direction with the largest variance. If columns of A do not have zero means, the direction along which row vector projections have the largest 2-norm or Euclidean length can be revealed by SVD. Figure 1 illustrates the data in an 18×2 matrix. The 18 points in the 18×2 matrix have different variances along different directions, hence data have the largest variance along v_1 as shown in Figure 1.

Along the direction of the first right singular vector, the projections of row vectors in A have the largest 2-norm, and along the second right singular vector direction, the projection 2-norm is the second largest, and so on. The singular values reflect the Euclidean lengths or 2-norms of the projections along the corresponding right singular vectors.

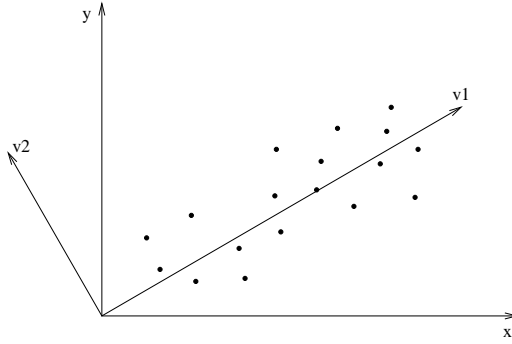


Fig. 1. Geometric structure of matrix exposed by its SVD

As shown in [1], any real $m \times n$ matrix A can be decomposed into $A = U\Sigma V^T$, where $U = [u_1, u_2, \dots, u_m] \in R^{m \times m}$ and $V = [v_1, v_2, \dots, v_n] \in R^{n \times n}$ are two orthogonal matrices, and Σ is a diagonal matrix with diagonal entries being the singular values of A : $\sigma_1 \geq \sigma_2 \geq \dots \geq \sigma_{\min(m,n)} \geq 0$. Column vectors u_i and v_i are unit vectors and are the i^{th} left and right singular vectors of A , respectively.

For similar motions with different lengths, their left singular vectors are of different lengths, but their right singular vectors are of the equal length. The singular values of matrix A are unique, and the singular vectors corresponding to distinct singular values are uniquely determined up to the sign, or a singular vector can have opposite signs [7]. For convenience, we will refer to the right singular vectors as singular vectors.

4 Feature Vector Extraction for Indexing

Motion matrices should have similar geometric structures if the corresponding motions are similar. Since the geometric similarity of matrix data can be captured by SVD, we propose to exploit SVD to generate representative vectors or feature vectors for motion matrices, and use these feature vectors for indexing the multi-attribute motion data.

As Figure 2 shows, the first singular values are the dominating ones among all singular values. Since the singular values reflect lengths or magnitudes of the row vector projections along their corresponding singular vectors, we can say that the first singular vectors are the dominating vectors. If two motions are similar, their corresponding first singular vectors u_1 and v_1 should be mostly parallel to each other geometrically, so that $|u_1 \cdot v_1| = |u_1||v_1|\cos(\theta) \doteq |u_1||v_1| = 1$, where θ is the angle between the two right singular vectors u_1 and v_1 , and $|u_1| = |v_1| = 1$ by the definition of SVD. Similarly, the first singular vectors are also very likely to be different from each other when two motions are different. Other corresponding singular vectors may not be close to each other even if two motions are similar as shown in Figure 3. This suggests that the first right

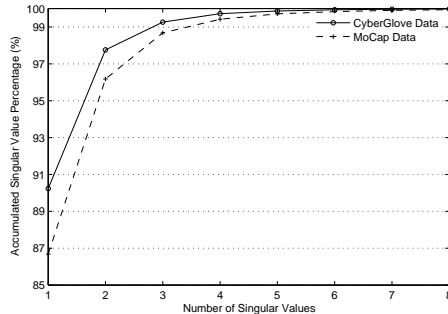


Fig. 2. Accumulated singular value percentages in singular value sums for CyberGlove data and captured human body motion data. There are 22 singular values for the CyberGlove data and 54 singular values for the captured motion data. The first singular values are more than 85% of the corresponding singular value sums.

singular vectors can be used to index multi-attribute motions for pruning the majority of different motions.

It is worth noting that for motions to be similar, other singular vectors and singular values should also be considered as shown in [5]. Although being necessary conditions for similarity measure, similar first singular vectors are sufficient for indexing purpose as to be demonstrated in Section 6.

Since the lengths or dimensions of the first singular vectors of multi-attribute motion data are usually larger than 15, dimensionality reduction needs to be performed on them first in order to avoid the so-called "curse of dimensionality." We use SVD further to reduce the dimensionality of the first singular vectors to be indexed. Let A be the matrix composing the first singular vectors of the motions to be indexed, and

$$A = W \Sigma Z^T$$

then $AZ = W \Sigma$ gives the projected/transformed first singular vectors of motion patterns in the coordinate system spanned by the column vectors of Z [3], and for a singular vector u_1 of a query motion, $u_1 Z$ gives a corresponding transformed singular vector of u_1 in the system spanned by the column vectors of Z .

Due to singular value decomposition, the component variations of the transformed first singular vectors are the largest along direction z_1 , and decreases along directions z_2, \dots, z_n as shown in Figure 4. The differences among the first singular vectors are optimally reflected in the first several dimensions of the transformed first singular vectors, hence we can index the first singular vectors by indexing only the first several components of the transformed singular vectors. Differences among all the other corresponding components are small even if motions are different, so the other components can thus be truncated and the dimensionalities are reduced to the first several ones. We refer to the transformed singular vectors after dimensionality reduction as the *feature vectors* of the mo-

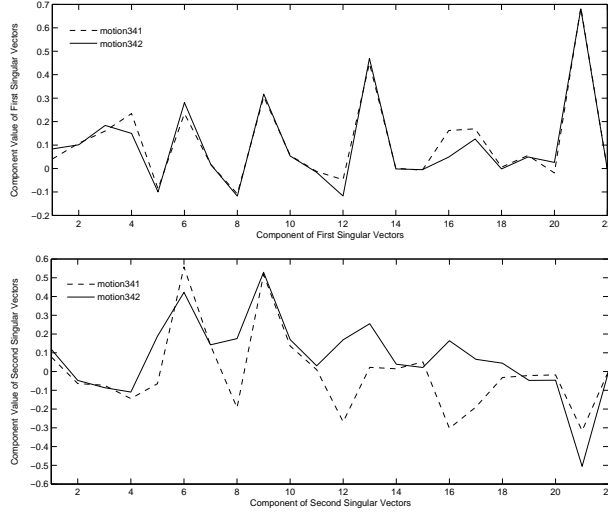


Fig. 3. Singular vectors of similar motions. The first singular vectors are similar to each other, while other singular vectors, such as the second vectors as shown at the bottom, can be quite different.

tions. If the first component of a feature vector is negative, all components of this vector are negated to obtain a consistent sign for feature vectors of similar motions [6].

5 Index Tree Construction

Let r be the dimension of the feature vectors, $r < n$. We designate one level of the index tree to each of the r dimensions. Let level 1 be the root node, level i includes nodes for dimension i , $i = 1, 2, \dots, r$, and level $r + 1$ contains leaf nodes. Leaf nodes contain motion identifiers P_k , and non-leaf nodes contain entries of the form

$$(I_i, cp)$$

where I_i is a closed interval $[a, b]$ describing the component value ranges of the feature vectors at level i , $-1 \leq a, b < 1$. Each entry has the address of one child node, and cp is the address of the child node in the tree.

The width and boundary of interval I_i depend on the distribution of i^{th} component values of feature vectors and the possible variations of the i^{th} feature vector components of similar motions. Let δ_i be the maximum difference of the i^{th} feature vector components of any similar motions, let x_i and y_i be the respective minimum and maximum values of the i^{th} components of all feature vectors, and let ϵ be the entry interval factor for adjusting entry intervals. Then the width of

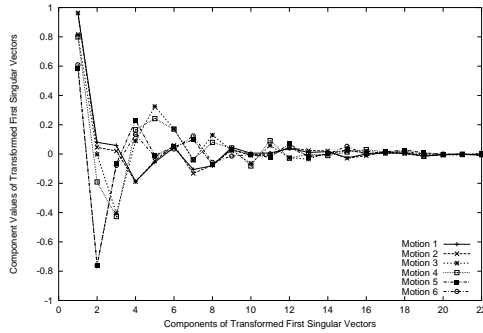


Fig. 4. Component distributions of the transformed first singular vectors

entry intervals at the i^{th} level is $\epsilon\delta_i$, and the number of entries of a node at level i is $\lceil (y_i - x_i)/(\epsilon\delta_i) \rceil$, limited by maximum number of entries per node allowed.

5.1 Insertion and Searching

Let the root node of the tree be T . The unique ID of a feature vector is inserted into the tree by comparing the i^{th} component c_i of the feature vector and the entry interval $[a, b]$ of the node traversed and can be inserted into multiple neighboring intervals:

- **Subtree Insertion:** If T is a non-leaf node, find all entries whose I_i 's overlap with $[c_i - \epsilon\delta_i, c_i + \epsilon\delta_i]$. For each overlapping entry, find the subtree whose root node T is pointed to by cp of the overlapping entry.
- **Leaf Node Insertion:** If T is a leaf node, insert the motion pattern identifier P_k of the feature vector in T .

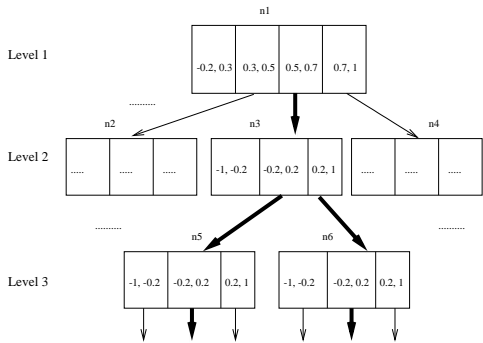


Fig. 5. An index tree example showing three non-leaf levels. Bold lines show where a feature vector is to be inserted.

Figure 5 illustrates how to insert an example feature vector into the first three levels of an example index tree. Root node at level 1 has four entries, each of which has a child node at level 2. Each node at level 2 and level 3 has three entries, and each of which has a child node at one lower level. Given a feature vector $f = (0.65, 0.15, -0.1, \dots)$, and let $\delta_1 = 0.04$, $\delta_i = 0.08$ for $i \geq 2$, and $\epsilon = 1.0$. Entries at the root node are checked with $[0.65 - 0.04, 0.65 + 0.04] = [0.61, 0.69]$. Only the third entry overlaps with it, hence the vector f is forwarded only to node n_3 of level 2. At level 2, the feature vector covering range is $[0.15 - 0.08, 0.15 + 0.08]$ or $[0.07, 0.23]$. The second and third entries of node n_3 overlap with the feature vector covering range $[0.07, 0.23]$, hence the feature vector will be forwarded to node n_5 and to node n_6 at level 3. At level 3, the feature vector covering range is $[-0.1 - 0.08, -0.1 + 0.08]$ or $[-0.16, -0.02]$. Only the second entries of nodes n_5 and n_6 overlap with this range, so the nodes pointed by the second entries of nodes n_5 and n_6 will be traversed for insertion. This process goes on until the leaf nodes are traversed for holding P_k of the feature vector f .

A query searching can be very simple: find the entry of the node whose interval $[a, b]$ covers the i^{th} component c_i of the query feature vector and traverse to the corresponding child node pointed by the entry. When a leaf node is reached, all the motion identifiers included in that leaf node are returned for the query. Since a node entry contains all possible similar motions in neighboring entries of the same node, only one entry is needed to be traversed for a search at each level of the tree, rather than multiple entries to be traversed as in [6].

5.2 Similarity Computation

After the index tree has been searched for a query, the majority of irrelevant motions should have been pruned, and similar motions and a small number of irrelevant motions are returned as the result of the query. To find out the motion most similar to the query, a similarity measure shown below as defined in [5] can be used to compute the similarity of the query and all the returned motions, and the motion with the highest similarity is the one most similar to the query.

$$\Psi(Q, P) = \frac{1}{2} \sum_{i=1}^k ((\sigma_i / \sum_{i=1}^n \sigma_i + \lambda_i / \sum_{i=1}^n \lambda_i) |u_i \cdot v_i|)$$

where σ_i and λ_i are the i^{th} singular values corresponding to the i^{th} right singular vectors u_i and v_i of square matrices of Q and P , respectively, and $1 < k < n$. Integer k determines how many singular vectors are considered and depends on the number of attributes n of motion matrices. Experiments with hand gesture motions ($n = 22$) and human body motions ($n = 54$) show that $k = 6$ is large enough without loss of pattern recognition accuracy in streams.

6 Performance Evaluation

Let N_{pr} be the number of irrelevant motions pruned for a query by the index tree, and N_{ir} be the total number of irrelevant motions in the database. We

define the pruning rate \mathcal{P} as

$$\mathcal{P} = \frac{N_{pr}}{N_{ir}} \times 100\%$$

6.1 Motion Data Generation

Motion data was generated for hand gestures by using CyberGlove and for dances and other human motions captured by using 16 Vicon cameras. There are 22 attributes for the CyberGlove data, and each attribute is for the angular values of one joint of the glove. There are 54 attributes for the motion capture data, and each attribute is for the positional values of one joint of a moving subject. The captured motion data had been transformed so that similar motions performed at different locations, following different paths, or at different orientations have "similar" data matrices. One hundred and ten different hand gestures were generated, and each one was repeated for 3 times, resulting in 330 data matrices of 22 columns. Sixty two different motions, including Taiqi and dances were performed, and each one was repeated for 5 times, resulting in 310 data matrices of 54 columns.

6.2 Index Struction Building

We experimented with different tree configurations for CyberGlove data and motion capture (MoCap) data. For CyberGlove data of 22 attributes, feature vectors have 5 to 10 components, or trees of 5 to 10 levels were tested. For MoCap data of 54 attributes, trees of 5 to 12 levels were tested. The entry interval factors ϵ we tested were 1.5, 1.2, 1.0, 0.9, 0.8, 0.7, 0.6 and 0.5. The smaller the entry interval factors, the smaller the entry intervals, and the more the number of entries in a node at all levels.

6.3 Pruning Efficiency

We issued one query for every one of the 330 CyberGlove motions and the 310 MoCap motions. Figure 6 shows that when all similar motions were retrieved and the feature vectors have 9 features, 95.7% irrelevant CyberGlove motions and 91% irrelevant MoCap motions could be pruned. When the entry interval factor ϵ is no less than 1.0, all similar motions can be retrieved, and when ϵ is less than 1.0, the most similar motions can still be retrieved and only a small number of less similar motions can be pruned as indicated by the high recalls as shown in Figure 7.

6.4 Computational Efficiency

We tested the average CPU time taken by a query using different tree configurations. All experiments are performed on one 3.0 GHz Intel processor of a GenuineIntel Linux box.

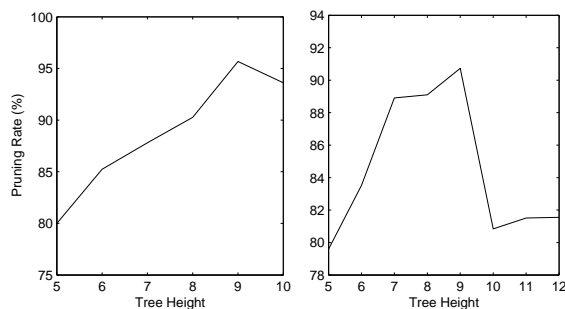


Fig. 6. Pruning rates of trees with different levels when all similar motions are to be retrieved. Left: CyberGlove data with $\epsilon = 1.0$; Right: MoCap data with $\epsilon = 0.8$.

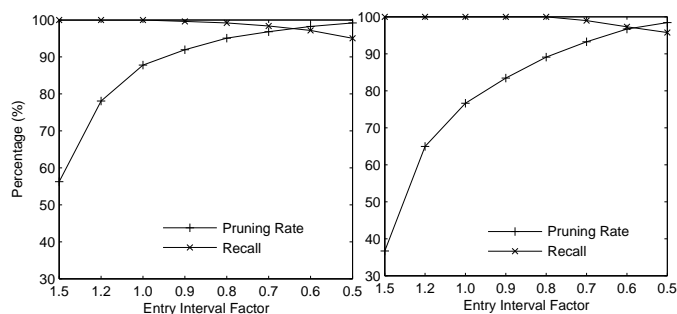


Fig. 7. Recalls and pruning rates for trees with height of 7 and different entry interval factors ϵ . Left: CyberGlove data; Right: MoCap data.

The search time of a query by using the proposed index structure takes less than $3 \mu s$ as shown in Figure 8. As a comparison, the search time of a query by the algorithm in [6] can take several milliseconds as shown in Figure 9. As a tradeoff, the proposed approach in this paper takes a little longer for inserting feature vectors. Nevertheless, each insertion still takes less than 35 milliseconds as shown in Figure 10 and is usually done off-line.

7 Conclusions

This paper has proposed a novel approach for indexing multi-attribute motion data of different lengths. Feature vectors are extracted from motion data matrices by using SVD properties, and an interval-based tree structure is proposed for indexing the feature vectors. Feature vector IDs can be inserted into multiple neighboring feature value intervals to cope with motion variations and can be in multiple leaf nodes. As an advantage of this design, search of similar motions can be done in several microseconds by traversing only one node once at each

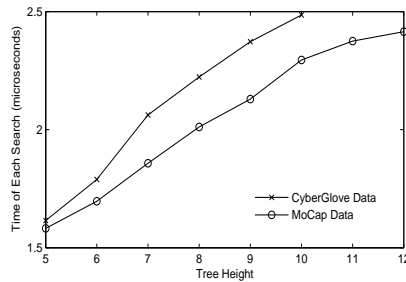


Fig. 8. Search time for one query by the proposed indexing approach

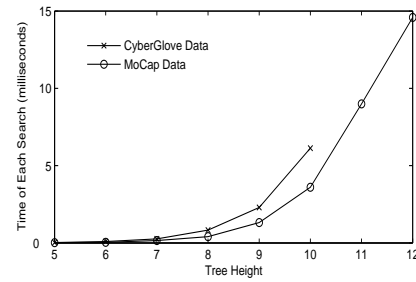


Fig. 9. Search time for one query by the index structure as proposed in [6].

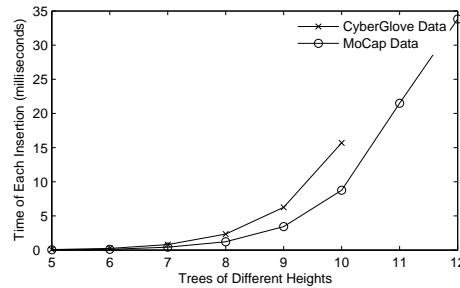


Fig. 10. Time taken for inserting a new motion ID in the indexing tree

tree level, and up to 95.7 % different CyberGlove motions and 91% captured human motions can be pruned.

References

1. G. H. Golub and C. F. V. Loan. *Matrix Computations*. The Johns Hopkins University Press, Baltimore, Maryland, 1996.
2. T. Kahveci, A. Singh, and A. Gurel. Similarity searching for multi-attribute sequences. In *Proceedings of 14th Int'l Conference on Scientific and Statistical Database Management*, pages 175 – 184, July 2002.
3. F. Korn, H. V. Jagadish, and C. Faloutsos. Efficiently supporting ad hoc queries in large datasets of time sequences. In *SIGMOD*, pages 289–300, May 1997.
4. S.-L. Lee, S.-J. Chun, D.-H. Kim, J.-H. Lee, and C.-W. Chung. Similarity search for multi-dimensional data sequences. In *Proceedings of 16th Int'l Conference on Data Engineering*, pages 599 – 608, Feb./Mar. 2000.
5. C. Li and B. Prabhakaran. A similarity measure for motion stream segmentation and recognition. In *Proceedings of the Sixth International Workshop on Multimedia Data Mining*, pages 89–94, August 2005.
6. C. Li, G. Pradhan, S. Zheng, and B. Prabhakaran. Indexing of variable length multi-attribute motion data. In *Proceedings of the Second ACM International Workshop on Multimedia Databases 2004*, pages 75–84, November 2004.
7. B. D. Schutter and B. D. Moor. The singular value decomposition in the extended max algebra. *Linear Algebra and Its Applications*, 250:143–176, 1997.
8. Online Vicon products introduction, <http://www.vicon.com/jsp/products/products.jsp>.
9. M. Vlachos, M. Hadjieleftheriou, D. Gunopulos, and E. Keogh. Indexing multi-dimensional time-series with support for multiple distance measures. In *SIGMOD*, pages 216–225, August 2003.