

Hierarchical Indexing Structure for 3D Human Motions

Gaurav N. Pradhan, Chuanjun Li, and Balakrishnan Prabhakaran

Department of Computer Science
University of Texas at Dallas, Richardson, TX 75083
{gnp021000, chuanjun, praba}@utdallas.edu

Abstract. Content-based retrieval of 3D human motion capture data has significant impact in different fields such as physical medicine, rehabilitation, and animation. This paper develops an efficient indexing approach for 3D motion capture data, supporting queries involving both sub-body motions (e.g., *Find similar knee motions*) as well as whole-body motions. The proposed indexing structure is based on the hierarchical structure of the human body segments consisting of independent index trees corresponding to each sub-part of the body. Each level of every index tree is associated with the weighted feature vectors of a body segment and supports queries on sub-body motions and also on whole-body motions. Experiments show that up to 97% irrelevant motions can be pruned for any kind of motion query while retrieving all similar motions, and one traversal of the index structure through all index trees takes on an average 15 μ sec with the existence of motion variations.

1 Introduction

Several scientific applications, especially those in medical and security field, need to analyze and quantify the complex human body motions. Sophisticated motion capture facilities aid in representing the complex human motion in the 3D space. The 3D human joint data from motion capture facility helps in analysis and comparison of the motions.

Focus of the Paper: Our main objective of this paper is to find similar 3D human motions by constructing the indexing structure which supports queries on sub-body motions in addition to whole-body motions. We focus on content-based retrieval for the sub-body queries such as *Find similar shoulder motions*, *Find similar leg motions* etc., or more regular query on whole body such as *Find similar walking human motion*. Some of the major challenges in indexing large 3D human motion databases are:

- 3D motions are multi-dimensional, multi-attribute and co-related in nature; associated segments of one sub-body (e.g. hand) must be processed always together along every dimension.
- Human motions exhibit huge variations in speed for similar motions as well as in directionality.

Proposed Approach: In our approach, we represent the positional information of different human body joints in a motion as a feature point. Using these feature points, a composite index structure for 3D human motions comprising five index trees is constructed. Each of the five index trees corresponds to one sub-body part (torso, left hand, right hand, left leg, and right leg). Each level of the index tree is designated to a joint of the corresponding sub-body part depending on the hierarchical structure of the human body joints. The mapped feature points of the joint associated with the level are grouped together. Each level prunes the irrelevant motions for the given query with respect to associated joint. And finally, each index tree gives the relevant motions for the query with respect to corresponding sub-body part. The output of relevant motions is then ranked using a similarity measure. For the whole motion query, the outputs from all index trees are merged and then ranked to get the most relevant motions for the whole-body query.

2 Related Work

In recent years, some approaches have been proposed on motion-retrievals from motion database. [12] constructed qualitative features describing geometric relations between specified body points of a pose and uses these features to induce a time segmentation of motion capture data streams for motion indexing. For each query a user has to select suitable features in order to obtain high-quality retrieval results. In [10], the authors cluster motion poses using piecewise-linear models and construct indexing structures for motion sequences according to the transition trajectories through these linear components. Similarly, posture features of each motion frame are extracted and mapped into a multidimensional vector in [3] for motion indexing. In [9], the authors use a hierarchical motion description for a posture, and use key-frame extraction for retrieving the motions.

Keogh et al. [6] use bounding envelopes for similarity search in one attribute time series under uniform scaling. The iDistance [14] is a distance-based index structure, here dataset is partitioned into clusters and transformed into lower dimension using similarity with respect to reference point of cluster. MUSE [13] extends [14] where partitioning of dataset at each level of the index tree is based on the differences between corresponding principal component analysis (PCA).

3 3D Motion Index Structure Design

We need to extract the feature characteristics from the motion matrix; such that joints' motions are represented as entities in the low dimensional feature space (fd-space). When we map the entire matrix for the two walking motions (Figure 1(a, b)), the mapped feature vectors are as shown in Figure 1(e) (Figure 1(f) zooms components corresponding to leg segments). Now, we can also map only the sub-matrix corresponding to leg motion alone, as shown in

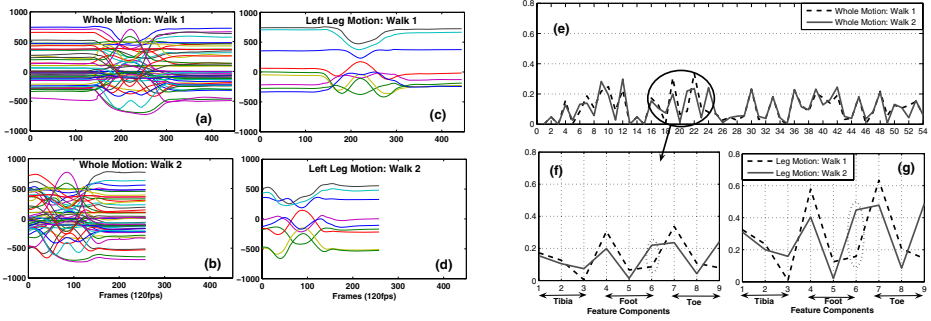


Fig. 1. (a)&(b): X, Y, Z trajectories of all segments for two similar walking motions. (c)&(d): Corresponding trajectories of only leg segments (tibia, foot, toe). (e) Feature components for whole body motions. (f) Feature components associated to only leg segments from(e). (g) Feature components for individual leg segments.

Figure 1(g). The differences between the same feature components are amplified and hence we can determine the similarity or dissimilarity between the two motions in a better way. The way this mapping of matrix/sub-matrix is done influences the query resolution. The details of mapping function are explained in Section 5.1.

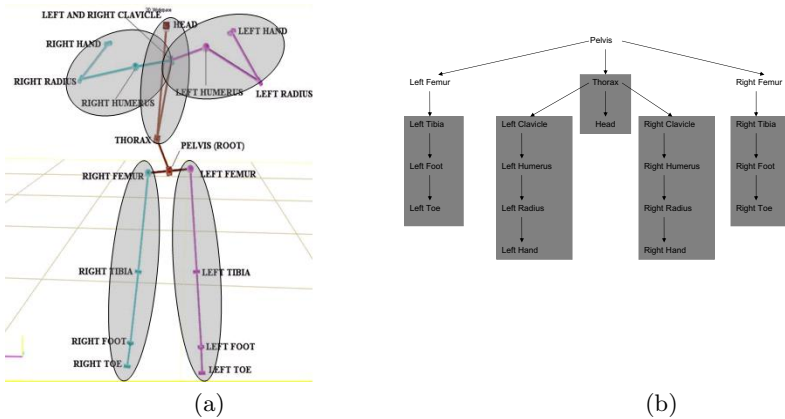


Fig. 2. (a)Segment structure for human body with five major sub-body parts. (b) Hierarchical tree structure of human Body segments.

This motivates us to design the index tree for each sub-body part depending on the hierarchical structure (Figure 2(b)) of the body segments (Figure 2(a)) so that we can resolve both sub-body motion as well as whole body motion queries efficiently. This structure consists of five branches with pelvis segment as the root. This human body structure inspires us to have a composite index structure consisting of five index trees corresponding to each branch.

4 Constructing Sub-body Index Trees

On mapping the joint data in 3D feature space, we can index these mapped points by constructing a corresponding index tree. Most mapping functions in literature [5], [2], [1] provide only similarity measures, i.e. they are not metrics. Due to the non-metric characteristic of the mapping functions available, it is difficult to *strictly* rank similar motions for a given query. Hence, it is better to retrieve the set of motions that lie within a threshold distance from the query motions feature point.

Now, let us consider two correlated joints such as tibia and foot. The movement of foot joint is constrained by or related to the tibia joint. This implies that for retrieving similar leg motions, we first retrieve the group of similar tibia motions, and only retrieved motions are considered for finding similar foot motions, and finally for similar toe motions. This leads us to the index tree structure for leg part of the body as shown in Figure 3.

The number of nodes constructed in Level j are equal to total number of groups present inside the nodes of immediate higher level (i.e. Level $j - 1$). Each node has a parent in form of group in immediate high level. In each node, joint feature vectors corresponding to Level j are mapped in 3D-indexed space. But, the patterns present only in parent group are mapped as a 3D-point inside the node.

A node of the index tree has the following structure,

$$\begin{aligned}
 N &: (G_1, G_2, \dots, G_e) \\
 G_i &: (R, S, C, \text{child - pointer})
 \end{aligned}
 \tag{1}$$

A node N consists of e groups of mapped points G_1, \dots, G_e formed by grouping the feature space. Each entry G_i consists of bounding hyper-rectangular region R , S is a set of n pattern identifiers whose mapped feature points are present in R and *child - pointer* is a pointer to the node in the next level of an index tree. C is the centroid of the group G_i .

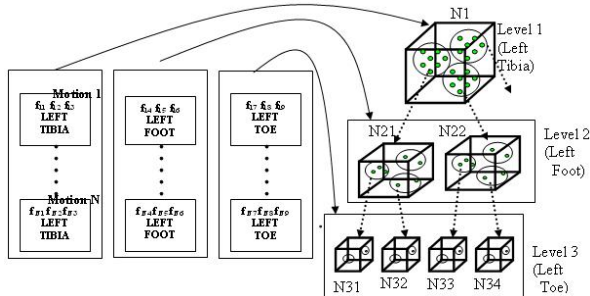


Fig. 3. Construction of hierarchical index tree for leg segments

An example: The construction of the left leg index tree is illustrated in Figure 3. The tibia feature points mapped in indexed space forms the root node

N1 in the Level 1 of left leg index tree. Let us assume that the indexed space is clustered to get three groups G1, G2 and G3, each containing feature points for similar motions. Every group G1, G2, and G3 becomes parent of node N21, N22 and N23 in Level 2 respectively. In node N21, foot feature points of all motions present in G1 are mapped in foot indexed space, which is further partitioned. This process is repeated for all nodes in Level 2. And the construction of the index tree is continued for last level in similar fashion.

4.1 Index Tree Operations

Pattern Insertion: Let us consider the insertion of a new motion in the left arm index tree. For this new motion, let f_x be the feature vector for left arm motion. Let $(f_{x(1-3)})$ feature point mapped in feature space corresponds to clavicle segment, $[f_{x(4-6)}]$ to humerus segment, $[f_{x(7-9)}]$ to radius segment and $[f_{x(10-12)}]$ to hand segment.

Using the threshold δ_c on each component of the feature vector of new inserting pattern, say for instant f_{xc} , we get a range defined as follows,

$$[f_{xc}] \implies [f_{xc} - \delta_c, f_{xc} + \delta_c] \quad (2)$$

In first level of arm index tree, we map the new motion of clavicle segment in feature space. So, the new pattern's clavicle segment feature vector becomes a hyper-rectangular region. The dimension of this region is $H_x = [(f_{x1} - \delta_1, f_{x2} - \delta_2, f_{x3} - \delta_3), (f_{x1} + \delta_1, f_{x2} + \delta_2, f_{x3} + \delta_3)]$. If this region overlaps with multiple groups inside node, we store new pattern identifier p_x in S structure of each overlapped group. The following routine shows the insertion procedure in *Node*,

The patterns in the next level will be inserted only in child nodes of the overlapped groups. The insertion procedure is same but the thresholds and H_x will change depending on the variations in components of similar feature vectors associated with next level.

Pattern Search: A query search can be very simple: find a group in a node whose boundaries covers the query feature vector or if not, the nearest group to query vector. This group will have copies of all the similar motions from neighboring groups. So there is no need to traverse multiple groups. The query is traversed forward to the corresponding child node pointed by the overlapped or nearest group. When a leaf node is reached, all the motion identifiers included in that leaf node are returned.

Ranking the similar motions: After the index tree has been searched for query, the majority of irrelevant motions should have been pruned. To find out most similar motions to given query we need to rank them in order of similarity. A similarity measure [7] shown in equation(3) can be used to compute the similarity of the query and all returned motions, and the motions with highest similarity has the highest rank or most similar to the query.

$$\Psi(Q, P) = \frac{1}{2} \sum_{i=1}^k ((\sigma_i / \sum_{j=1}^n \sigma_j + \lambda_i / \sum_{j=1}^n \lambda_j) |u_i \cdot v_i|) \quad (3)$$

where σ_i and λ_i are the i^{th} eigenvalues corresponding to the i^{th} eigenvectors u_i and v_i of square matrices of Q and P , respectively, and $1 < k < n$.

4.2 Handling Whole-Body Queries

In some cases, queries on the whole body motions would be more meaningful than queries on sub-body motions. For example, if we want to find motions similar to certain swimming stroke whole body considered together would be more useful.

The five index tree returns the respective similar motions for the sub-body parts. To get similar whole body motion, we need to merge these outputs by taking the intersection of all returned pattern sets. The common patterns in all output sets from index trees will form the answer for the whole body query. The ranking of similarity patterns is again decided by similarity measure using equation (3) with $n = 48$ (all segments).

5 Index Structure Implementation

With the global positions, it becomes difficult to analyze the motions performed at different locations and also in different directions. Thus, we do the local transformation of positional data for each body segment by shifting the global origin to the pelvis segment because it is the root of all body segments. The segments included in the five index trees are highlighted in Figure 2(b).

5.1 Mapping Function for Joint Matrices

An appropriate mapping function is required to map 3D motion joint matrices into 3D feature points in the feature space. In our implementation, we used the linearly optimal dimensionality reduction technique SVD [5] for this purpose. For any $m \times 3$ joint matrix A , the SVD is given as follows,

$$A^{m \times 3} = U^{m \times m} \cdot S^{m \times 3} \cdot V^{3 \times 3} \tag{4}$$

S is a diagonal matrix and its diagonal elements are called singular values. And columns of V matrix are called right singular vectors. We add up the three right singular vectors weighted by their associated normalized singular values to construct the features for a joint motion as follows:

$$f_c = \sum_{i=1}^3 (w_i \cdot v_{ci}) \tag{5}$$

where $w_i = \frac{\rho_i}{\sum_{j=1}^3 \rho_j}$, $\sum_{i=1}^3 w_i = 1$, $c = \{1, 2, 3\}$, and $[\rho_1, \rho_2, \rho_3]$ is singular value vector and v_{ci} is the c^{th} component of the i^{th} right singular vector and w_i is the normalized weight for the i^{th} right singular vector. The weighted joint feature vector of length 3 represents the contribution of the corresponding joint to the

motion data in 3D space and also captures the geometric similarity of motion matrices.

The feature vectors for joints are represented as mapped points in feature space. For the similar feature vectors, corresponding mapped points will be close to each other. This creates a need to do grouping of such points which can simplify the similarity search.

5.2 Node Grouping Approach

Several approaches have been suggested in the literature for grouping data [8], [4], [15]. In our work, we formed grouping in all nodes of index tree using the hierarchical, self-organizing clustering approach [11].

Hierarchical, Self-organizing Clustering Approach: In this approach, a node is spliced into two groups if heterogeneity is greater than defined threshold. Heterogeneity(H_t) is a measure to calculate the distribution of the mapped points in a given group. Scattered points in group give high heterogeneity value and closely distributed points give low heterogeneity value. The threshold(T_h) is determined by product of this measure and heterogeneity scaling factor α .

$$H_t = \sum_{j=1}^D \frac{\|x_j - C\|_2}{|D|} \quad T_h = H_t * \alpha$$

where C is the 3D-centroid of the node containing total D patterns and x_j is the mapped 3D coordinates of patterns inside group. We iteratively do the splicing on groups until heterogeneity values of all formed groups are below threshold T_h . These groups become the parent of the nodes in the next level.

Similarly, other index trees are constructed to build whole indexing structure. The (left/right) hand index tree has Level 1 associated with clavicle segment to Level 4 associated with hand segment.

Overcoming Space Partitioning problem: The DGSOT is a space partitioning approach. Due to different variations in performing similar motions, the corresponding mapped points may fall into different groups after clustering causing false dismissals in resulting output of similar motions for the query. Hence, the sizes of the group must be re-adjusted by some ‘‘threshold’’ to include most of the similar motions from the neighboring groups.

To solve the space partitioning problem, we capture the uncertainty of differences in similar motions for a joint in different feature dimension using standard deviation. In a database of M motions, we have E sets of pre-determined similar motions. Let $simDev_c$ be the standard deviation of the differences between similar motions for the c^{th} feature component.

Using $simDev_c$, we get the threshold δ_c to enlarge the group along the c^{th} dimension of the feature space as follows,

$$simTolerance_c = \delta_c = \epsilon * simDev_c \quad (6)$$

$simTolerance_c(\delta_c)$ is a final threshold to enlarge the group along c^{th} dimension. ϵ is an input parameter which varies in the range of 0.2 – 1. The larger ϵ gives high threshold and a group is enlarged to involve more feature points from neighboring groups along all feature dimensions. As a result, the pruning efficiency goes on decreasing and rate of false dismissals falls.

6 Performance Analysis

The human motion data was generated by capturing human motions using 16 high resolution Vicon cameras connected to a data station running Vicon iQ software. Our test bed consists of 1000 human motions, performed by 5 different subjects.

Let N_{pr} be the number of irrelevant motions pruned for a given query by the index tree, and N_{ir} be the total number of irrelevant motions in the database. We define the pruning efficiency \mathcal{P} as

$$\mathcal{P} = \frac{N_{pr}}{N_{ir}} \times 100\% \tag{7}$$

6.1 Pruning Efficiency

For each experiment, we issued 1000 queries to calculate the average pruning efficiency for the indexing tree as shown in Figure 4(a) and Figure 4(b).

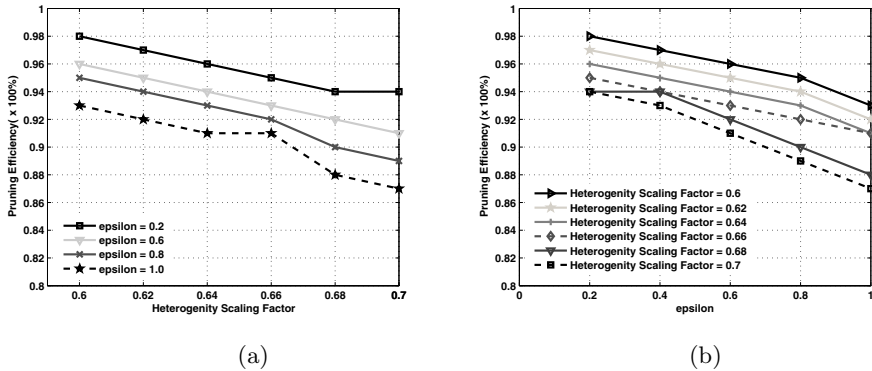


Fig. 4. (a) Pruning efficiency for different heterogeneity scaling factors. (b) Pruning efficiency for different input parameters (ϵ).

As we go on increasing α , the heterogeneity threshold goes on increasing, and more pattern-corresponding feature points get accumulated in the same partition, which reduces the pruning power due to inclusion of some irrelevant motions. There is a steady increase in the pruning efficiency as we decrease the heterogeneity scaling factor (Figure 4(a)). The effect on pruning efficiency was

also studied by keeping α constant and varying the input parameter ϵ (Figure 4(b)). As we increase ϵ , the *simTolerance*(δ) for all components goes on increasing as a result the pruning efficiency goes on decreasing.

For whole body motion query, average pruning efficiency achieved is 98% where we take “intersection” of the set of relevant motions from five index trees.

6.2 Recall

When a motion query is given to an index tree, ideally it should return all similar motions. However, in practice, resolution of some queries may have some irrelevant motions due to the non-metric nature of the similarity measures used as well as the variations in performing similar motions. Figure 5 shows the average recall for different configurations of the index tree. As input parameter ϵ goes on increasing, the average return of similar patterns for the given query (i.e. hits) goes on increasing.

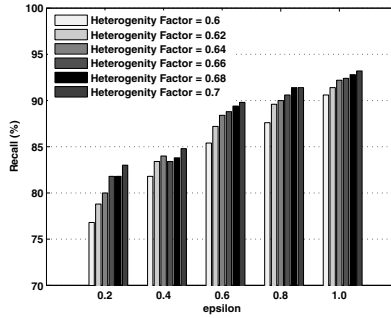


Fig. 5. Recall for different configurations of the index tree

6.3 Comparison with MUSE

The MUSE indexing structure works very well on indexing synthetic hand gesture motions generated using an instrumented device called CyberGlove [13]. Since hand gesture motions have multi-attributes and different variations just like captured 3D human motions, MUSE seems to be the most suitable indexing structure published so far for multi-attribute motion data. For performance comparison, we applied MUSE on our database of 3D human motions. The MUSE structure was constructed using 3 Levels. By querying 1000 3D motions, the pruning efficiency achieved was 5.5%, as compared to 97% of our index tree structure. Also, the average computational time required per query was 0.3 seconds. In our case, for the same set of queries, the average computational time per query is 15 μ sec. The lower bound defined by MUSE for pruning irrelevant motions [13] is not tight enough for 3D human motions.

7 Conclusions and Discussions

This paper considered content-based motion querying on a repository of multi-attribute 3D motion capture data. We proposed a composite index structure that maps on to the hierarchical segment structure of the human body. This composite structure comprises five independent index trees corresponding to the five identified body parts: body/thorax, two arms & two legs. In each of these five index trees, a tree level is assigned to one segment feature vector. At each level, similar feature points inside all nodes are grouped together to increase the pruning power of the index tree.

We tested our prototype using a database of approximately 1000 human motions. Our experiments show that up to 96~97% irrelevant motions can be pruned for any kind of motion query while retrieving all similar motions, and one traversal of the index structure through all index trees takes on an average 15 μ sec. Finally, our approach is also applicable to other forms of multidimensional data with hierarchical relations among the attributes.

References

1. R. Agrawal, C. Faloutsos, and A. Swami. Efficient similarity search in sequence databases. In *Proc. 4th Conference on Foundations of Data Organization and Algorithms*, pages 69–84, October 1993.
2. K.-P. Chan and A. W.-C. Fu. Efficient time series matching by wavelets. In *Proc. 15th International Conference on Data Engineering*, pages 126 – 133, March 1999.
3. S.-P. Chao, C.-Y. Chiu, J.-H. Chao, Y.-C. Ruan, and S.-N. Yang. Motion retrieval and synthesis based on posture features indexing. In *Proc. Fifth International Conference Computational Intelligence and Multimedia Applications*, pages 266–271, September 2003.
4. C. Ding and X. He. K-means clustering via principal component analysis. In *ICML '04: Proceedings of the twenty-first international conference on Machine learning*, page 29, New York, NY, USA, 2004. ACM Press.
5. G. H. Golub and C. F. V. Loan. *Matrix Computations*. The Johns Hopkins University Press, Baltimore, Maryland, 1996.
6. E. Keogh, T. Palpanas, V. B. Zordan, D. Gunopulos, and M. Cardle. Indexing large human-motion databases. In *Proc. 30th VLDB Conference*, pages 780–791, Toronto, Canada, 2004.
7. C. Li and B. Prabhakaran. A similarity measure for motion stream segmentation and recognition. In *Proc. MDM/KDD, The sixth International Workshop on Multimedia Data Mining*, Chicago, IL USA, August 2005.
8. B. Liu, Y. Xia, and P. S. Yu. Clustering through decision tree construction. In *CIKM '00: Proceedings of the ninth international conference on Information and knowledge management*, pages 20–29, New York, NY, USA, 2000. ACM Press.
9. F. Liu, Y. Zhuang, F. Wu, and Y. Pan. 3D motion retrieval with motion index tree. *Computer Vision and Image Understanding*, 92:265–284, June 2003.
10. G. Liu, J. Zhang, W. Wang, and L. McMillan. A system for analyzing and indexing human-motion databases. In *Proc. 2005 ACM SIGMOD International conference on Management of data*, 2005.

11. F. Luo, L. Khan, F. Bastani, I.-L. Yen, and J. Zhou. A dynamically growing self-organizing tree (dgsot) for hierarchical clustering gene expression profiles. *Bioinformatics*, 20:2605–2617, May 2004.
12. M. Muller, T. Roder, and M. Clausen. Efficient content-based retrieval of motion capture data. *ACM Transactions on Graphics (TOG)*, 24:677–685, 2005.
13. K. Yang and C. Shahabi. Multilevel distance-based index structure for multivariate time series. In *TIME*, Burlington, Vermont, USA, 2005. IEEE Computer Society.
14. C. Yu, B. C. Ooi, K.-L. Tan, and H. V. Jagadish. Indexing the distance: An efficient method to knn processing. In *Proc. VLDB '01*, pages 421–430, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.
15. T. Zhang, R. Ramakrishnan, and M. Livny. Birch: an efficient data clustering method for very large databases. In *SIGMOD '96: Proceedings of the 1996 ACM SIGMOD international conference on Management of data*, pages 103–114, New York, NY, USA, 1996. ACM Press.