Eurographics/ACM SIGGRAPH Symposium on Computer Animation (2004)
R. Boulic, D. K. Pai (Editors)

Motion Map: Image-based Retrieval and Segmentation of Motion Data

Yasuhiko Sakamoto, Shigeru Kuriyama and Toyohisa Kaneko

Toyohashi University of Technology, Japan

Abstract

Recent proliferation of motion capture systems enables motion data to be saved as an archive system, and the data are usually extracted by selecting an appropriate file by its name or annotation explaining the content of motions. Such semantic-based retrieval, however, is not suited to unstructured files that include many types of elemental motions, due to the difficulty in giving comprehensible annotations. Moreover, expected motion clips are often included as a part of entire sequences, and the data therefore should be manually clipped using some authoring tools.

This paper proposes an image-based user interface for retrieving motion data using a self-organizing map for supplying recognizable icons of postures. The postures are used as keys for retrieval, and the desirable segments of the motion data can be accurately extracted by specifying their starting and ending postures. The number of possible motion segments is flexibly controlled by changing the scope of postures used as the keys.

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Three-Dimensional Graphics and Realism]: Animation H.3.3 [Information Search and Retrieval]: Retrieval models H.5.2 [User Interfaces]: Graphical user interfaces (GUI)

1. Introduction

In creating humanoid animations, intensive use of motion capture data has become a common strategy for producing plausible movements. Thus editing these motion data requires smart tools for cutting and pasting some successive frames. Such tools must be interactive in retrieving a desirable segment of a motion clip. File names or attached annotations are usually determined to show some hints about the content of included motions. However, making such annotations often requires elaborate work for the careful selection of appropriate words that explain whole aspects or features included in the motion data. In addition, data classification and categorization are subject to individual interpretation, making it difficult to select appropriate words for complex or subtle motions in such a way that most users can identify the same images of motions without misunderstanding the content. More importantly, cutting and concatenating motion clips are essential tasks for editing animations. Semantic-based retrieval, however, lacks the capability of accurately clipping the proper segment of the data.

For these reasons, we have developed a new type of graphical user interface for retrieving motion data using non-verbal, image-based keys. We utilize the technology, called self-organizing maps (SOM), for mapping large data sets of high dimensionality onto two-dimensional discrete space. This technology is often used in the application of information visualization because of its capability to distribute data elements while preserving the topological relations of metrics in a higher dimensional space onto a 2D square map. Representative postures are automatically selected by clustering nodes of the map, and are then converted to image icons to be used for navigating the selection of key-postures. By this graphical interface, users can easily detect desirable motion segments by simply clicking key-nodes.

Not only postures but also their velocity or acceleration are essential variables for representing motions, but such time-dependent information requires additional visual cues or indications on each postures, which often complicates the appearance of a map. Since an exact amount of velocity or acceleration is difficult to perceive with static images, semantic selection may be adequate for narrowing down the data including similar postures and different dynamics. For these reasons, our system employs no visual information for dynamical attributes.

The outline of this paper is as follows: Section 2 explains the relevant works and Section 3 introduces a method of computing a motion map and constructing a graphical user interface by embedding posture icons and drawing motion trajectories. Section 4 explains a way of efficiently searching for motion segments among a large amount of data set by narrowing possible files with successive keys. In Section 5, we show the results of usability tests, and finally give concluding remarks and discussion.

2. Related work

Data-driven motion generations were first proposed for blending motion segments to reconstruct new motions [UAT95, WH97, RCB98, BH00]. Recent availability of motion capture data enables variety of behaviors to be automatically generated by concatenating elemental motion clips with transition graphs [TH00, KGP02, LCR*02, AFO03]. These method requires detecting similar motion segments for blending and transitions, and automated extraction method is proposed [KG04]. This method uses a motion clip as a query, and is therefore unsuited to retrieve data from scratch. In other words, existing method of extracting motion segments requires a motion sample to detect matching segments. Our method, on the other hand, extracts motion segments by conjecturing postures included in a requested motion clip.

Our system displays many selectable images for user inputs, and such strategy was first introduced into computer graphics community as design galleries [Mea97] for setting parameters of rendering and animation. However, we have not found similar work related to the retrieval of motion capture data using relevant images. Our methodology bears some similarity to the methods of multivariate data analysis proposed in the field of information visualization. Since enormous space is required for listing all relevant works in the field, we here refer to only surveying papers and books [CMS99, HMM00, Spe01, dOL03].

The key technology of our image-based system is the smart distribution of a huge number of postures. The appropriate mapping of the posture space enhances the ability to detect desirable postures. This problem is considered nonlinear dimensionality reduction whose numerical methods include principal component analysis (PCA), multidimensional scaling (MDS), isomap [TdSL00], locally linear embedding [RS00], and self-organizing maps (SOM) [Koh88, Koh90, Koh01]. These methods can arrange complex high-dimensional data sets while reflecting the metric relationships between the data elements in the topology of their arrangement, by which some kind of abstraction is automatically produced. Our method utilizes SOM for mapping posture data in a 2D plane because of its capability in distributing the data uniformly. In other words, SOM distributes input samples (postures) so that their probability of placement on the map becomes uniform. Some methods of information visualization introduced SOM for the classification of textual data with respect to their context [RI96] and ontology [EP02], whereas our method constructs SOM with the numerical data representing a posture at each frame. Constructing SOM usually requires a huge computational cost. This defect, however, does not affect the real-time response of interactive retrievals because the computation of SOM can be preprocessed.

3. Motion map

Our system uses human postures as keys for retrieval, but it is not a simple task to specify postures from scratch. Using sketch-based input can supply an intuitive user-interface [DAC*03], but still requires many manipulations for specifying even a single posture. We therefore introduce a strategy of displaying many postures to be selected as keys, which requires a technique for efficiently distributing all necessary postures to cover whole display space.

We here employ SOM for the arrangement of posture data included in motion files, and call the resulting map of postures a *motion map*, which is used in constructing a graphical user interface for motion data retrieval. In the following subsection, we explain the computation of the motion map in detail.

3.1. SOM for posture space

For each sample posture, an input vector $\mathbf{x} \in \Re^n$ is defined as

$$\mathbf{x} = \{\mathbf{P}_1\,,\, \mathbf{P}_2\,,\, \ldots\,,\, \mathbf{P}_{n_J}\}\,,$$

where n_J denotes the total number of joints, and 3D vector $\mathbf{P}_i$ is the relative position of i-th joints from a root position. For normalizing root orientations, they are rotated along a vertical axis so as to turn a posture toward a standard direction. Figure 1 shows the sampled joints for input vectors whose total dimensions is $n = 30\ (n_J) \times 3\ (\mathit{space\ dimension}) = 90$.

The motion map is defined as a grid array whose nodes are uniformly arranged, and each node of the (i, j)-th location associates a parametric real vector $\mathbf{m}_{i,j} \in \Re^n$, which is called a *model vector*, by which discrete locations of postures are determined. It is noteworthy that the model vector $\mathbf{m}_{i,j}$ is not a substitution of the input vector $\mathbf{x}$, but is an image (or approximation) computed by taking a mixture of the input vectors. The original incremental SOM algorithm updates the model vector of each node with the input vector as follows:

$$\mathbf{m}_{i,j}(t+1) = \mathbf{m}_{i,j}(t) + h_{i,j}(\mathbf{x},t)[\mathbf{x}(t) - \mathbf{m}_{i,j}(t)]\,,\ \ \mathbf{m}_{i,j}(0) = 0\,,$$

where $t = 0, 1, 2, \ldots$ denotes the number of iterations. The

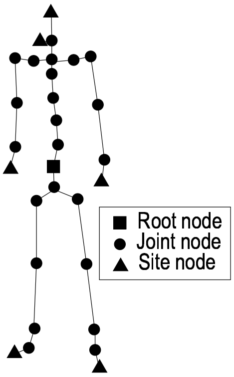

Figure 1: *Joint locations used as input vectors.*

position of each posture is then determined by the node location (p,q) whose model vector has the minimum distance against the input vector of the posture as

$$\mathbf{m}_{p,q} = \min_{i,j}\{d(\mathbf{x}, \mathbf{m}_{i,j})\} , \quad (1)$$

where $d(\mathbf{x}, \mathbf{m}_{i,j})$ represents a distance between $\mathbf{x}$ and $\mathbf{m}_{i,j}$ that is defined by Euclidean distance in $\Re^n$;

$$d(\mathbf{x}, \mathbf{m}_{i,j}) :=\| \mathbf{x} - \mathbf{m}_{i,j} \| \ .$$

The function $h_{i,j}(\mathbf{x},t)$ is called a neighboring function which plays an important role as a smoothing kernel. We adopted a widely used kernel using the Gaussian function as the neighboring function

$$h_{i.j}(\mathbf{x},t) = \alpha(t) \exp\left(-\frac{(i-p)^2+(j-q)^2}{2\sigma^2(t)}\right) ,$$

where the index (p,q) is obtained by the equation (1). The function $\alpha(t)$ denotes the learning-rate factor, and $\sigma(t)$ is the width of the kernel. The values of these functions are monotonically decreased according to the progress of iterations. We experimentally designed these functions as

$$\alpha(t) = \begin{cases} 0.8*(1.0-\frac{t}{6n_L}) & if\ \alpha(t) > 0.01 \\ 0.01 & otherwise \end{cases} ,$$

$$\sigma(t) = 0.25*(n_x+n_y)*(1.0-\frac{t}{n_L}) ,$$

where n_L denotes the total number of iterations that is experimentally set at 1000 for ensuring convergence, and n_x and n_y are the numbers of the grid along each 2D coordinate.

Constructing the SOM involves a large computational cost because motion data usually contain huge samples (over 30 postures per second). Moreover, the distribution of postures on the resulting map is often biased according to the density of similar postures. For these reasons, we remove similar postures from samples used as input vectors. For each posture included in motion files, we compare the dissimilarity against all postures registered as samples. The posture in process is added to the samples if its dissimilarity metric, which is computed by the distance of their input vectors $d(\mathbf{x}, \mathbf{y})$, exceeds a given threshold against all the samples. By this preprocess, we can efficiently decrease the number of input vectors (samples) and thereby reduce the bias caused by the dense postures.

Figure 2 shows the example of a motion map, in which posture images are drawn at their corresponding nodes, where the circled area zooms in on the map. The motion map was computed by 436 posture samples extracted from 55,114 frames of 51 data files, where all motions are measured against the same performer. It took about 17 minutes on a Pentium-4 2.4 GHz processor for 21×21 nodes.

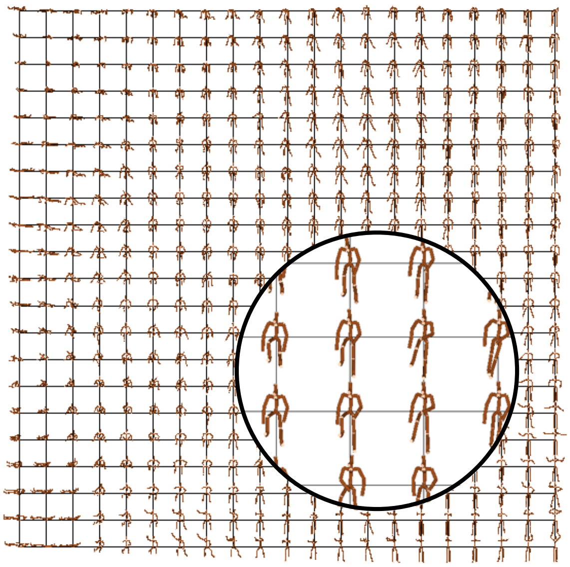

Figure 2: *Motion map.*

3.2. Generations of posture icons

All postures allocated at every node can be used as keys, but iconic representation of all images, as shown in Figure 2, often imposes psychological load in selecting adequate postures. Simply reducing the resolution of the grid array, however, compromises the accuracy of the search. Therefore, it is necessary to select adequate representative postures for display so as to cover the entire outlines of all the motion files without losing any of the essential components.

We here introduce a clustering technique to automatically extract such postures. Motion map arranges postures so that their topological relationship reflects the distance relation on input vector space, but the uniformity of distance cannot be ensured between neighboring postures. From this property, the existing methods, such as K-means [Mac67], sometimes clusters the nodes into disconnected regions on the map.

Moreover, partitioning algorithms of clustering methods require predetermining the number of clusters, but it is difficult for users to choose its optimal value. Therefore, we introduce a clustering method that divides regions by detecting borders between neighboring nodes.

For each node of the motion map, we first compute the average of the difference against four neighboring nodes as

$$\tilde{\mathbf{m}}_{i,j} = \frac{1}{4} \sum_{(k,l)=(i\pm1,j),(i,j\pm1)} (\mathbf{m}_{i,j} - \mathbf{m}_{k,l}) \ ,$$

and we then create a vertical border at the node that satisfies

$$\tilde{\mathbf{m}}_{i-1,j} < \tilde{\mathbf{m}}_{i,j} \quad \textbf{and} \quad \tilde{\mathbf{m}}_{i,j} > \tilde{\mathbf{m}}_{i+1,j} \ ,$$

where this border represents a ridge or valley of the map. The horizontal and slanting borders are similarly created by comparing the $\tilde{\mathbf{m}}_{i,j}$ with $\tilde{\mathbf{m}}_{i,j\pm1}$ and $\tilde{\mathbf{m}}_{i\pm1,j\pm1}$, respectively.

After creating all borders, we divide the map by labeling the connected region. The labeling technique is similar to those used in clustering raster images; it scans each line of nodes from top to bottom, and each node is visited from left to right while copying or creating labels according to the existence of the borders. Similar scanning process follows for merging the labels so that the connected nodes have a unique label if they have no border. As a result, all the nodes that have the same label comprise a clustered region. This map-driven clustering automatically determines the number of clusters according to the pattern of the distributed postures. Figure 3 shows the label of clustered regions obtained by our method, where the thick lines indicate borders.

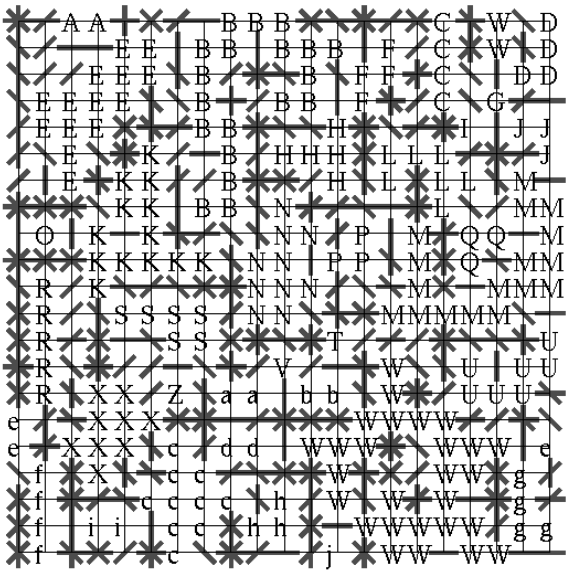

Figure 3: *Clustered regions of motion map.*

An icon of the posture, which navigates the selections of key-postures, is then generated from the model vector of the node that is nearest to the center of each clustered region. The iconic images generated at every region are shown in Figures 6 and 9. Instead of using the actual human appearance of postures, our iconic representation uses simple skeletal objects so as to reduce superfluous visual information. This simplification relies on the study of structural object perception using Geon [HB92, ITW01]. However, this hypothesis should be confirmed through a usability test.

Figure 4 shows the graphical user interface of our prototype system. For enhancing usability, the image of the posture at the node pointed by a mouse is rendered from different viewpoints, and the result of the retrieval is displayed using a pop-up list menu with a button that prompts to expand searching scope.

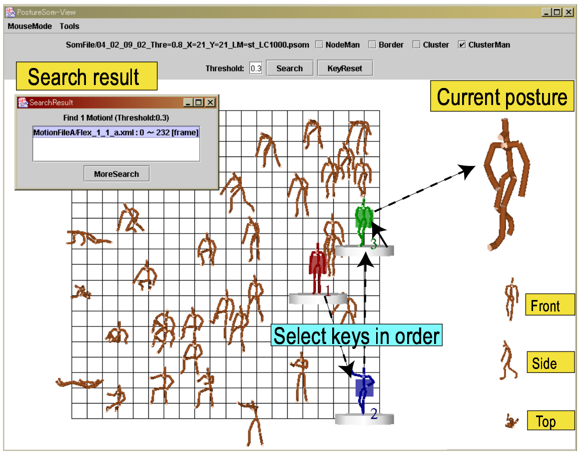

Figure 4: *Prototype of GUI.*

3.3. Motion trajectory

Each motion file is a sequence of postures, and it can be represented as a trajectory on the map drawn by a polyline connecting corresponding nodes. The trajectory requires a high resolution grid so as to have distinguishable shapes of the trajectories between the motion files. However, the computational cost in creating a motion map increases in proportion to the number of nodes. We therefore introduce virtual nodes to increase the resolution of the map with small computational cost. The virtual nodes are automatically created by uniformly subdividing square regions, as shown in Figure 5, and the model vectors at the created nodes are simply determined using a spherical bilinear interpolation of the quaternions at the surrounding actual nodes. This computation requires little cost and could be preprocessed for greater detail with the cost of storage. The trajectory of the motion data is drawn by traversing the nearest nodes for each posture per-frame, computed by the equation (1). Figure 6 shows an example of the trajectory drawn for a walking motion. This visualization technique is useful for observing the sequence of postures included in a motion clip. In addition, this interface could be utilized as a tool for analyzing the features of each motion clip or concatenating motion clips by detecting frame periods that have similar postures.

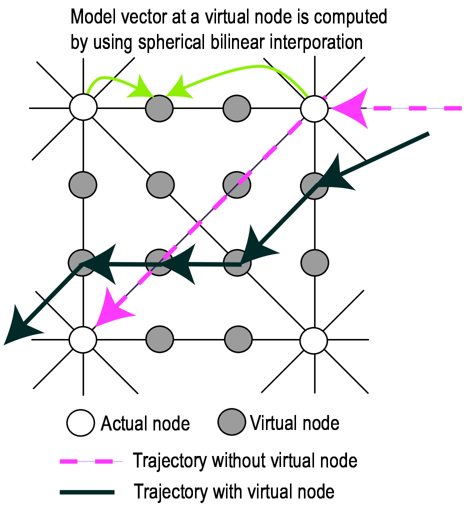

Figure 5: *Creation of virtual nodes.*

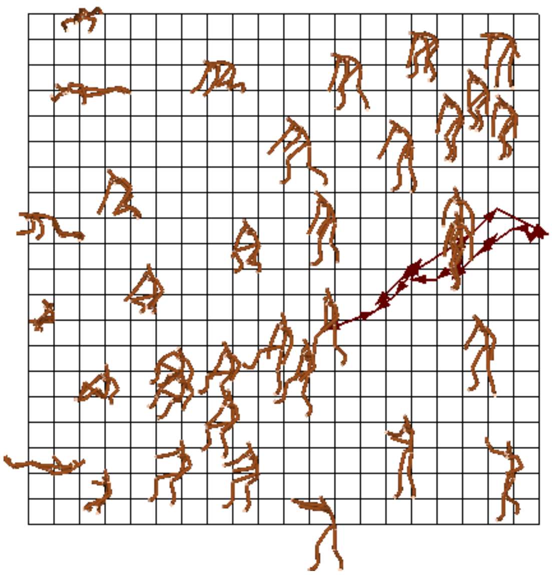

Figure 6: *Trajectory of walking motion.*

4. Retrieval mechanism of a motion segment

After constructing a motion map, our system creates a hash table for efficiently searching possible files and segments. The entry of the table is an index of each node, and each element of the table consists of file ids and frame indices whose postures correspond to the node of the entry, where the correspondence is determined by computing the equation (1). Successive frames belonging to the same node are compactly stored by their start and end indices. The process of searching motion segments is as follows (see Figure 7):

Step.1 All elements of the first key-node entry are copied to a retrieval list.

Step.2 Each element of the retrieval list searches the corresponding element of the following key-node entry that has the same file id and the end frame index larger than its own end index. If the corresponding elements are plurally detected, the element of the smallest end index is selected, by which the shortest segment is intentionally composed.

Step.3 The element of the retrieval list is deleted if corresponding element is not detected. Otherwise, the value of its end index is replaced with those of the corresponding element.

Step.4 If the next key-node exists, go back to Step.2, otherwise go to the next step.

Step.5 Display all surviving elements of the retrieval list as the result of the searching process.

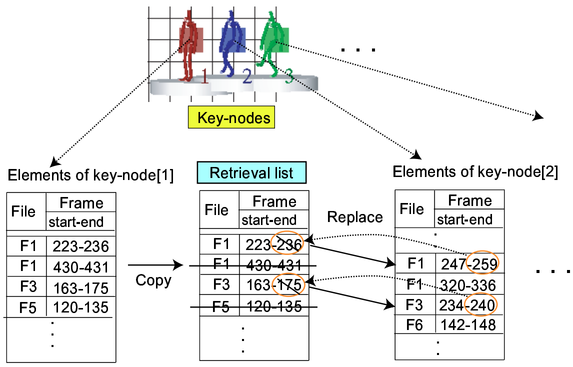

Figure 7: *Schematic representation of searching process.*

As the hash table registers each posture of motion data against a single node entry, it is often difficult for users to precisely select a key-node. For increasing flexibility and reliability in retrieval processes, our system supplies a way of selecting key-nodes by introducing a distance threshold T that determines the subset of nodes. All nodes are treated as keys if their model vectors satisfy the inequality of $d(\mathbf{m}_{p,q}, \mathbf{m}_{i,j}) < T$, where $\mathbf{m}_{p,q}$ is the model vector of the selected node. All these key nodes then compose the hash table elements for the entry of the selected node. The threshold T is gradually increased by a constant value through user interactions. Figure 8 shows the effects of changing the threshold T to expand a region of the key-nodes, where the dark quadrilaterals denote such regions.

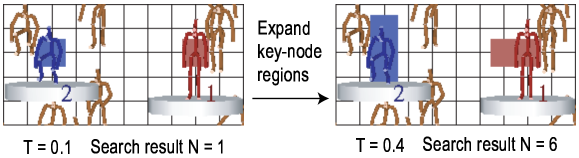

Figure 8: *Expanding key-nodes for extended search.*

5. Usability test

We experimented two types of usability tests for 10 subjects using a desktop PC of Pentium-4 2.4 GHz. The first test employed a data set that was captured by ourselves, and the second test employed those extracted from a commercial data set.

The first test evaluates the efficiency in retrieval by measuring the time and the number of trials spent in detecting a requested motion segment. Experimental data set consists of ordinary and task motions including various moving and manipulating behaviors whose motion map is shown in Figure 6. The map was learned from 51 data files, from which many motion segments can be clipped. Therefore, it includes several hundreds of motion segments that can be retrieved. Among such huge number of motion collection, we selected typical 8 motion clips as targets for retrieval. They were randomly displayed on a PC console using a commercial animation tool with a realistic shape of a humanoid. After providing an explanation on the usage of the systems, all subjects were allowed to freely operate the system for 3 trials of retrieval operations for their adaptations.

Table 1 shows the results of this usability test. The column of Target represents the kind of motion selected as target, and the column of Trial indicates the average (and maximum in brackets) number of trials for all subjects, spent in detecting the appropriate motion segment. The column of Time denotes the average time required for each trial, and Failure indicates the rate of subjects that could not detect the adequate segment.

Table 1: *Usability test (ordinary motions)*

Target	Trial (ave [max])	Time (sec.)	Failure
Carry	1.5 [2]	11.1	0.2
Throw	1.0 [1]	17.1	0.3
Push	1.3 [2]	17.1	0.3
Sit	1.2 [2]	20.7	0.0
Reach	1.8 [2]	17.0	0.2
Kick	1.6 [4]	18.3	0.0
Spear	1.4 [3]	28.5	0.1
Lie	1.1 [2]	10.0	0.0

A unique posture of lying makes retrieving task easy, which is demonstrated by the small number and the short period of trials. We found that posture icons are often missed if they are located at the peripheral of the motion map, and this affects the long trial times and many failures for the Throw and Push whose representative postures are located at the lower right corner. Posture icons are also difficult to select if their locations have many nearby posture icons; for example, the Carry caused failures and the Sit took the long trial time. The motions of the Reach and Kick are relatively basic movements, and thus they are included in various motions. We consider that this ubiquitous property affects the large number and the long period of trials. The conspicuous long trial time for the Spear may be caused by the ambiguity against both carrying and reaching motions.

The second usability test employed 17 motion files of various dancing styles. The motion map in Figure 9 was computed by 329 posture samples extracted from 3095 frames, and it took 6.5 minutes for 19×19 nodes. These motion files seem to be appropriately named, but the content is perceivable only to persons with experience. This test is utilized for evaluating the capability of our system in narrowing down possible files from a single posture.

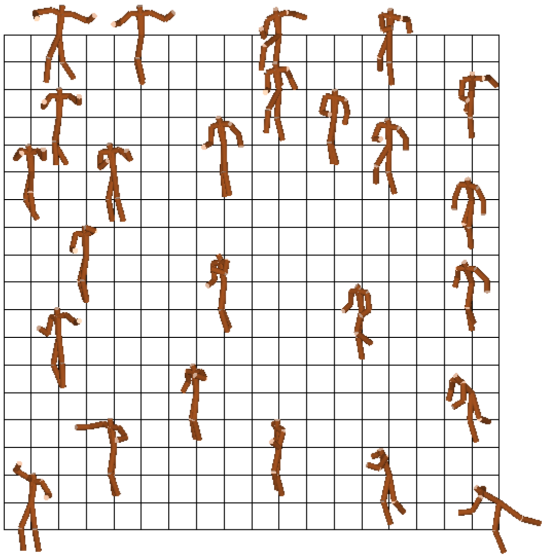

Figure 9: *Motion map for 17 dance styles.*

After similarly displaying the movie of each dancing style, all subjects, who have little knowledge about the styles of dance, were instructed to select a key-posture that is supposed to be special to the style. Each trial was continued until the resulting retrieval list includes the file of the target style.

Table 2 shows the same information as Table 1 where the additional column of Choice is the average number of possible files in the retrieval list. A small number of files is retrieved except the Develop, which demonstrates the capability in fully narrowing down possible files. The style of the Develop can be categorized into escort dance, and it includes support postures that are also included in other 8 files. This property is considered as a cause of the many possible files.

Table 2: *Usability test (dance motions)*

Target	Trial	Time	Failure	Choice
Develop	1.8 [4]	21.1	0.0	7.9
Generic	1.2 [2]	15.7	0.1	1.1
Chorus	1.2 [2]	6.41	0.0	1.5
Wiggin	1.9 [4]	21.9	0.0	2.2
Jive-kick	1.8 [3]	15.2	0.0	2.9

6. Conclusions

We have proposed a new type of retrieval system using key-postures for detecting motion segments from a large amount of data set. The contributions of our system are summarized as follows:

- Motion map is automatically constructed without using any manual categorizations and annotations.
- Motion map simply arranges the postures of motion data all together on a plane, which can hide a complex structure of file directories.
- Motion map displays a sequence of posture by drawing the trajectory of node transition onto a high resolution grid array.
- Posture icons supply simple visual cues for key selections by using representative postures obtained with a map-driven clustering.
- Motion clips are automatically extracted by giving starting and ending postures.
- Segments of similar motions are simultaneously retrieved, which allow users to check all possible motion clips at one try. This property is especially useful to motion syntheses that blend plural similar motion clips.

Iconic and non-verbal representation can supply intuitive user interfaces common in different languages and cultures. This property is especially useful in detecting specific motion segments from a long motion sequence of an artistic performance, such as a dance or drama, which are difficult to consistently annotate using a few words.

The target motions selected in the first usability test are easily annotated, and our image-based retrieval seems to have no big advantages over semantic-based ones. More investigation is needed with regard to the target motions that are difficult to annotate. We actually tried to compare the trial time against the semantic-based retrieval system that was developed as a hyper-text document. The time required to detect a relevant motion file for each target was measured for 3 subjects on the same condition as the proposed map-based retrieval. The efficiency varies depending on each subject and target, and we could not find any meaningful differences or features from the results. However, there was a tendency for many subjects to quickly adapt our interface and easily increase their performances. The efficiency in semantic-based retrieval deeply depends on the design of hyper-links, but there is no common data set for fairly evaluating the efficiency of motion retrieval. We should explore standard criteria for measuring the performance of retrieval.

Our system cannot distinguish the same motion category of different subjects (performers) because the difference between individuals is negligible. In addition, arrangement of static postures cannot reflect the dynamical feature of the motions; for example, it is impossible to distinguish and retrieve motions that have the same movements but at different velocities. Such defects could be compensated by adding semantic controls, or introducing some graphic interactions that can reflect dynamical variables. Our experimental data (of the first usability test) includes several hundreds of motion clips, but the scalability of the motion map should be fully investigated against further scales of data set.

The motion map has the potential to be utilized not only for data retrieval, but also for analyzing minute differences of individual movements by zooming in the motion trajectories drawn on a high-resolution map. Moreover, it could be used for interactive data editing. For example, a suitable frame correspondence during a transition period can be visually detected by drawing the trajectory of each motion sequence on the map. Trajectory drawings also could supply an intuitive tool for evaluating motion graphs, and motion blending might be controlled by manually drawing an interpolation path on the map. Such editing methodology may produce a new type of sketching interface in creating character animations.

Acknowledgment

This work is supported by the Grant-in-Aid for Scientific Research (C) 15500108 of the Japan Society for the Promotion of Science, and the grant of Toukai Foundation for Technology.

References

[AFO03] ARIKAN O., FORSYTH D. A., O'BRIEN J. F.: Motion synthesis from annotations. *ACM Transactions on Graphics 22*, 3 (2003), 402–408.

[BH00] BRAND M., HERTZMANN A.: Style machines. In *Proceedings of ACM SIGGRAPH 2000* (2000), pp. 183–192.

[CMS99] CARD S. K., MACKINLAY J. D., SHNEIDERMAN B.: *Readings in Information Visualization Using Vision to Think*. Morgan Kaufmann Publishers, 1999.

[DAC*03] DAVIS J., AGRAWALA M., CHUANG E., POPOVIC Z., SALESIN D.: A sketching interface for articulated figure animation. In *Symposium on Computer Animation 03* (2003), pp. 320–328.

[dOL03] DE OLIVEIRA M. C. F., LEVKOWITZ H.: From visual data exploration to visual data mining: A survey. *IEEE Transactions on Visualization and Computer Graphics 9*, 3 (2003), 378–394.

[EP02] ELLIMAN D., PULIDO J.: Visualizing ontology components through self-organizing maps. In *Proc. of Sixth International Conference on Information Visualization (IV'02)* (2002), pp. 434–438.

[HB92] HUMMEL J. E., BIEDERMAN I.: Dynamic binding in a neural network for shape recognition. *Psychological Review 99*, 3 (1992), 480–517.

[HMM00] HERMAN I., MELANÇON G., MARSHALL M. S.: Graph visualization and navigation in information visualization: A survey. *IEEE Transactions on Visualization and Computer Graphics 6*, 1 (2000), 24–43.

[ITW01] IRANI P., TINGLEY M., WARE C.: Using perceptual syntax to enhance semantic content in diagrams. *IEEE Computer Graphics & Applications 21*, 5 (2001), 76–85.

[KG04] KOVAR L., GLEICHER M.: Automated extraction and parameterization of motions in large data set. *ACM Transactions on Graphics* (2004).

[KGP02] KOVAR L., GLEICHER M., PIGHIN F.: Motion graphs. *ACM Transactions on Graphics 21*, 3 (2002), 473–482.

[Koh88] KOHONEN T.: Self-organized formation of topologically correct feature maps. *Biological Cybernetics 43* (1988), 59–69.

[Koh90] KOHONEN T.: The self-organizing map. *Proceedings IEEE 78*, 9 (1990), 1464–1480.

[Koh01] KOHONEN T.: *Self-Organizing Maps*. Springer, 2001.

[LCR*02] LEE J., CHAI J., REITSMA P. S. A., HODGINS J. K., POLLARD N. S.: Interactive control of avatars animated with human motion data. *ACM Transactions on Graphics 21*, 3 (2002), 491–500.

[Mac67] MACQUEEN J.: Some methods for classification and analysis of multivariate data. In *5th Berkeley Symposium* (1967), vol. 1, pp. 281–297.

[Mea97] MARKS J., ET AL.: Design galleries: A general approach to setting parameters for computer graphics and animation. In *Proceedings of SIGGRAPH 97* (1997), pp. 389–400.

[RCB98] ROSE C., COHEN M. F., BODENHEIMER B.: Verbs and adverbs: Multidimensional motion interpolation. *IEEE Computer Graphics & Applications 18*, 5 (1998), 32–40.

[RI96] RUSHALL D., ILGEN M.: Depict: Documents evaluated as pictures. visualizing information using context vectors and self-organizing maps. In *IEEE Symposium on Information Visualization (INFOVIS'96)* (1996), pp. 100–107.

[RS00] ROWEIS S. T., SAUL L. K.: Nonlinear dimensionality reduction by locally linear embedding. *SCIENCE 290*, 22 (2000), 2323–2326.

[Spe01] SPENCE R.: *Information Visualization*. Addison-Wesley, 2001.

[TdSL00] TENENBAUM J. B., DE SILVA V., LANGFORD J. C.: A global geometric framework for nonlinear dimensionality reduction. *SCIENCE 290*, 22 (2000), 2319–2322.

[TH00] TANCO L. M., HILTON A.: Realistic synthesis of novel human movements from a database of motion capture examples. In *Workshop on Human Motion (HUMO 2000)* (2000), pp. 137–142.

[UAT95] UNUMA M., ANJYO K., TAKEUCHI R.: Fourier principles for emotion-based human figure animation. In *Proceedings of SIGGRAPH 95* (1995), pp. 91–96.

[WH97] WILEY D., HAHN J.: Interpolation synthesis of articulated figure motion. *IEEE Computer Graphics & Applications 17*, 6 (1997), 39–45.