Biped Cartoon Retrieval Using LBG-Algorithm Based State Vector Quantization

Siriprapa Pakdee
Department of Computer Engineering
Faculty of Engineering
Chulalongkorn University
siriprapa4901@yahoo.com

Pizzanu Kanongchaiyos
Department of Computer Engineering
Faculty of Engineering
Chulalongkorn University
pizzanu@cp.eng.chula.ac.th

ABSTRACT

Efficiency of cartoon motion retrieval depends on indexing method. Keyword based indexing methods for restoring cartoon motion is simple, but sometimes the process of addressing index to the desired motion is not trivial. Although the methods based on clustering clips of motions or keyframes can be used for retrieval automatically, the complexity of time and reliability depend on size of the database. This research proposes a method for biped cartoon motion retrieval using state vectors of keyframes as the motion indexes. The state vectors derived from the input motion is quantized by our enhanced LBG Algorithm to compute the proper number of indexes for the set of input keyframes. The results show that not only the precision and recall of the cartoon motion retrieval is increased accurately but the process for restoring and retrieval is also performed automatically.

Keywords

Computer Animation, Biped Cartoon, Vector Quantization, State Vector, LBG Algorithm

1. INTRODUCTION

Computer games and animation industry at present is growing increasingly. Each animated movie or computer game has a lot of motion information which is expensive for construction and maintenance. In addition, creating a new animated movie or game needs a lot of time and resources, so the reuse of animation components is needed for decreasing the cost and time.

Storing cartoon motions in a database is an important method for reusing cartoon motion. If the storing method of cartoon motion databases is not efficient enough, it will affect the retrieval of cartoon motions. A method for restoring motions using motion index [Keo04a] have proposed a novel technique to speed up similarity search under uniform scaling, based on bounding envelopes for indexing human motions. Liu et al. [Liu05a] demonstrate a data-driven approach for representing, compressing, and indexing human-motion based on piecewise-linear components obtained using K-means clustering. Forbes and Fiume [For05a] presented a search algorithm to use with sampled motion data. They also developed a representation for motion data introducing a meaningful distance metric for input poses. The methods have been applied to the searching human motion in human motion database for motion synthesizing. Edmunds et al. [Edu05a] demonstrate a method of generating long sequences of motion by performing various similarity-based "joins" on a database of captured motion sequences, while Basu et al. [Bas05a] have proposed how to find similar frames to cluster them. For matching motions, query by example and cluster graph is used to find stored components having same significant DOFs. These methods for storing cartoon motions have some advantage of ability to store the cartoon motion automatically and rapidly, but their time complexity are still not good enough for large database, while their reliabilities of the retrieval motions depend on the size of the database.

This research proposes a method for biped cartoon motion retrieval using state vectors of keyframes as indices. The state vectors derived from the input motion is quantized by our enhanced LBG

Algorithm to compute the proper number of indexes for the set of input key frame.

Most algorithms for finding the representation of a vector set properly is based on Linde–Buzo–Gray (LBG) Algorithm [Lin80b, Non05a] and the pair-wise nearest neighbor (PNN) Algorithm [Ari03a] but their efficiency is limited by the user defined number of codewords which is the power of two. Improper size of the codebook causes the density problem such as the set of vector shown in Figure 1, while the proper number of codewords gives more suitable density in set of vector as shown in Figure 2.

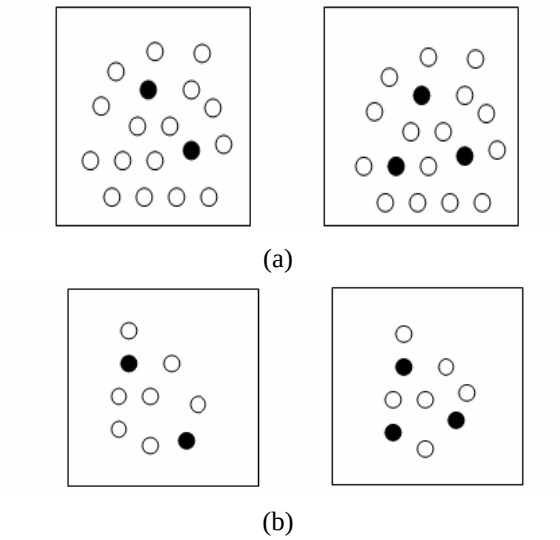

Figure 1. Problem of codebook having improper number of codewords: (a) Code book of two words and three words representing high density vector set (b) Code book size of two words and three words representing vector set having lower density

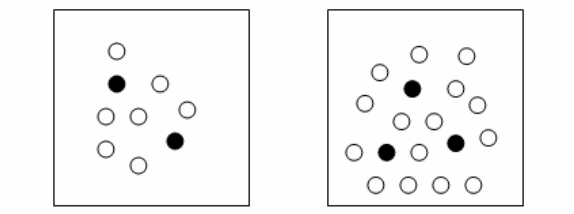

Figure2. Codebook has a number of codewords corresponding to density of vector.

Shanbehzadeh and Ogunbona [Sha97a] compared the computational complexity of the pair-wise nearest neighbor (PNN) and Linde–Buzo–Gray (LBG). They showed that for a practical codebook size and training vector sequence, the LBG algorithm was indeed more computationally efficient than the PNN algorithm.

The rest of this paper is organized as follows. In Section 2, we describe the principle of Vector Quantization and proposed how to enhance the LBG algorithm for finding number of codewords automatically in Section 3. State vectors are used to represent cartoon motions in Section 4. Finally, Section 5 shows an experiment of the restoring and retrieval cartoon motion.

2. VECTOR QUANTIZATION

The Vector Quantization algorithm is to select k representatives of vectors from vector space R^k into a set of vectors. Each y_i is a codeword, while set of codewords is a codebook. Each codeword y_i belongs to a nearest neighbor area called Voronoi region defined as follow;

$$V_i = \{x \in R^k : \| x - y_i \| \leq \| x - y_j \|, \text{for all } j \neq i\}$$

For example, two dimensional vectors in Figure 3 are in their belonging regions. Each group of vectors was represented by their codeword. A codeword must be a vector in the same region as its neighboring vectors.

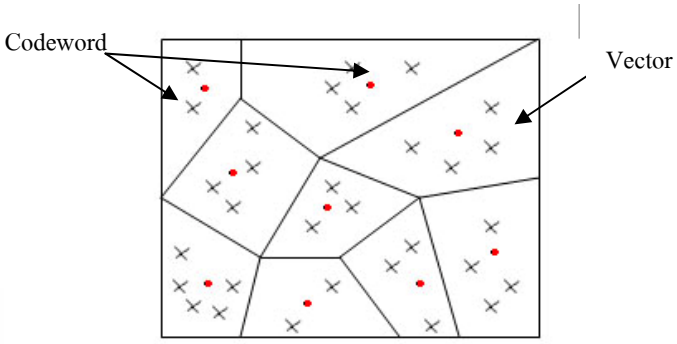

Figure 3. An example of vector quantization

The objective of vector quantization is to represent a group of vectors X with a codebook Y .When a vector $x \in X$ is approximated by a codeword y_i, an error usually occurs because $x \neq y_i$. Mean Square Error can be computed from the sum of distortion error of every group divided by total number of codeword. Efficiency of vector quantization is compared to the distortion error.

 ISBN 978-80-86943-02-2

Calculation for distance of x from y_i starts from defining a distance *d* as:

$$d : D^k \times D^k \rightarrow \Re$$

Vector quantization error can be computed using a function $d(x, y_i)$ which equals to function $d(x, q(x))$. In this research, the distance function for finding error is the Euclidean distance;

$$d(x, y) = \sqrt{\sum_{i=1}^{k} (x - y_i)^2} \quad (1)$$

Therefore the error function is Mean Square Error (MSE)

$$MSE = \frac{1}{N_p} \sum_{p=1}^{N_p} d(x_p, q(x_p)) \quad (2)$$

when x_p is the p[th] input vector and N_p is the number of codewords and $d(x_p, q(x_p))$ is Euclidean distance.

2.2 LBG Algorithm

LBG algorithm is a popular method among several codebook design methods. The algorithm is described as follows;

Algorithm 1 Finding quantized representation with LBG algorithm

Pre-condition: *N* is number of codeword, *T* is threshold distance

Post-condition: All vectors are quantized and represented by randomly choose input vector

For all number of codeword **do**

If $d(x, y_i) < d(x, y_j)$ **then**

x is subset of y_i

else x is subset of y_j

end if

end for

Calculate distortion

While ($\left| D_{prev} - D_{curr} \right| / D_{curr} > T$ **)**

Find centroid for new codeword

If $d(x, y_i) < d(x, y_j)$ **then**

x is subset of y_i

else x is subset of y_j

end if

end while

Final codebook

At the beginning, a vector in set *X* will be selected as an initial codebook by randomization and its Voronoi region is defined. By computing the Euclidean distance between each vector and the initial codebook, each vector neighboring to the codeword will be quantized to the same region. Then, the difference between MSE in present and previous iteration is computed to find the stability of the codebook. If stable condition is accepted, the algorithm is terminated. If not, the algorithm will iteratively calculate the cancroids vector of each region. The centroid is treated as a new codebook for the initial of next iteration.

3. ENHANCEMENT OF LBG ALGORITHM

We develop an enhanced LBG algorithm and use it to find number of representative vectors for each cluster automatically. In this research, the acceptance threshold and acceptance mean square error is user-defined. While comparing MSE from each iteration in the algorithm to the pre-defined threshold and pre-defined MSE that we can insert a new codeword to the codebook. If both values are less than the pre-defined value, the output result is the current codebook. If the MSE is greater than the pre-defined value but the MSE ratio less than threshold, then another codeword is inserted randomly and the LBG algorithm starts again. The enhanced algorithm is shown below and displayed as a diagram in Figure 6.

Algorithm 2 Enhanced LBG algorithm

Pre-condition: *S* is MSE, *T* is threshold, *M* =1

Post-condition: All vectors are quantized and represented by randomly choose input vector

For all number of codeword **do**

If $d(x, y_i) < d(x, y_j)$ **then**

x is subset of y_i

 ISBN 978-80-86943-02-2

x is subset of y_j

end if

end for

Calculate distortion

While ($\left| D_{prev} - D_{curr} \right| / D_{curr} > T$ **)**

Find centroid for new codeword

$M = M + 1$

If $d(x, y_i) < d(x, y_j)$ **then**

x is subset of y_i

else x is subset of y_j

end if

Calculate distortion

end while

while ($D_{curr} \geq S$ **)**

Find quantize represent by randomly choose input vector

$M = M + 1$

If $d(x, y_i) < d(x, y_j)$ **then**

x is subset of y_i

else x is subset of y_j

end if

Calculate distortion

end while

Final codebook

4. MOTION REPRESENTATION

In this Section, we will describe how to store cartoon motion into a database. First we represent cartoon motions extracted from keyframes as state vectors which is a set of parameter specifying angular values for each joint of the articulated cartoon character body at is a given time *t*. The dimension of the state vector is defined corresponding to the degrees of freedom (DOF) of the cartoon body. For example in Figure 5, the moving arm having three joints have two DOFs, one DOF and one DOF respectively, so each state vector of the arm keyframe in Figure 5 has five dimensions or five elements.

In our example will show storing the motion of an arm which has 3 joints. We use state vectors to represent keyframes of cartoon and use vector quantization by using enhance LBG algorithm as in figure 4.

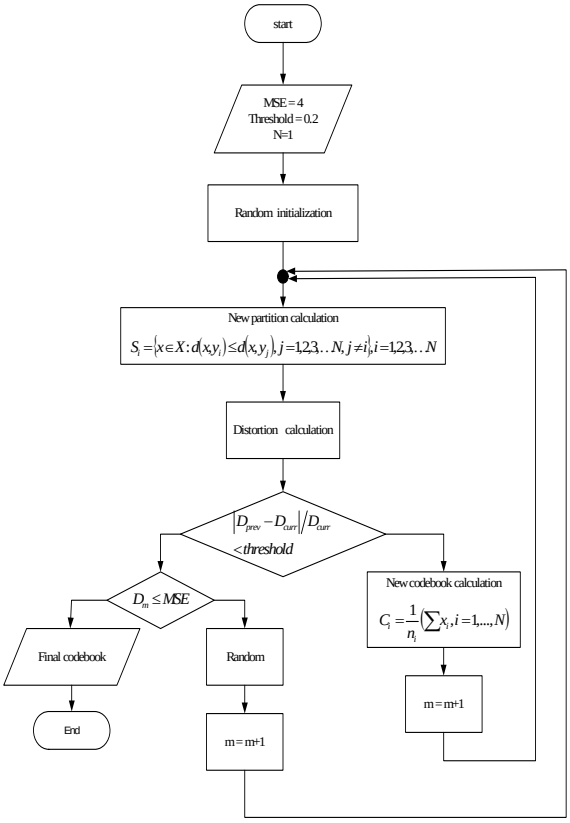

Figure 4. Procedure of enhanced LBG

Moving arm has 10 keyframes which have state vectors as follows:

q1 ((10, 20), 20, 30), q2 ((10, 20), 20, 35)

q3 ((10, 20), 25, 35), q4 ((10, 20), 25, 35)

q5 ((15, 25), 25, 30), q6 ((15, 25), 30, 30)

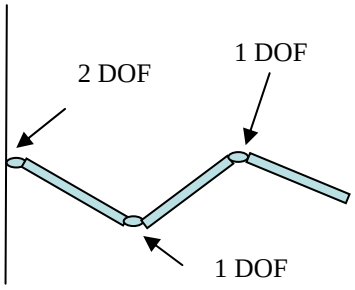

Figure 5. Arm which 1st joint has two DOFs (can move two directions). 2nd and 3rd joint have 1 DOF (can move one direction).

q7 ((15, 25), 30, 25), q8 ((20, 35), 30, 25)

q9 ((20, 35), 35, 20), q10 ((20, 35), 35, 15)

These state vectors are then represented as a codebook using enhance LBG vector quantization algorithm. By repeating this algorithm 3 times, the result of the codebook is shown in table 1. Result is optimal codewords for state vector of our motion as in figure 8 which θ_1 show state vector which is represent first joint which has two DOFs, θ_2 show state vector which represent second joint which has one DOF and θ_3 show state vector which represent third joint which has one DOF.

5. EXPERIMENTAL RESULTS

In this research, we set the threshold mentioned in Section 3 to 0.2 and MSE to 4 according to our statistical experimental results. When we got state vectors, we got one initial codeword by random. Define each vector to be same group with the nearest codeword: the nearest distance between codeword and state vector was calculated by Euclidean distance. Calculate Mean Square Error from distortion.

First, we stored skeletal cartoon motion data that was file format MDL1 which MDL1 was Apron Tutorials first 3D skeletal animation format modified by Ronny André Reierstad. The 3D Model 1 format called MDL1 is actually a converted Milkshape 3D ASCII format. You are able to load, import or create your own 3D models and animations with the popular Milkshape 3D editor and export as Milkshape 3D ASCII (*.txt) files. Skeletal cartoon motion data were stored in skeletal cartoon motion database. Those are the biped cartoon motion as working, running, jumping, and kicking and are stored into database in state vector form and number that is specified in cartoon through our program. Then we classify information into cluster and find representation of each cluster. We explain and show table that store all cartoon motion in figure 6. In the process, Information of biped cartoon in state vector format and also cartoon motion file and index of cartoon (use sequence of number to represent input cartoon motion by order) is stored into animation table in database by our program. Then we bring query cartoon motion to calculate its state vector which is characteristic of cartoon motion in that time. State vector attribute define motion of each cartoon. Number of state vector is not equal in each cartoon depend on number of keyframes of cartoon motion. State vector also is stored in animation table. So this table uses the attribute File_animation to represent cartoon motion, Number_animation represent cartoon index and State vector represent state vector. When we get state vector of each cartoon motion, we bring its to find representation of state vectors series in each motion by using vector quantization which represent out state vector by Representation table.

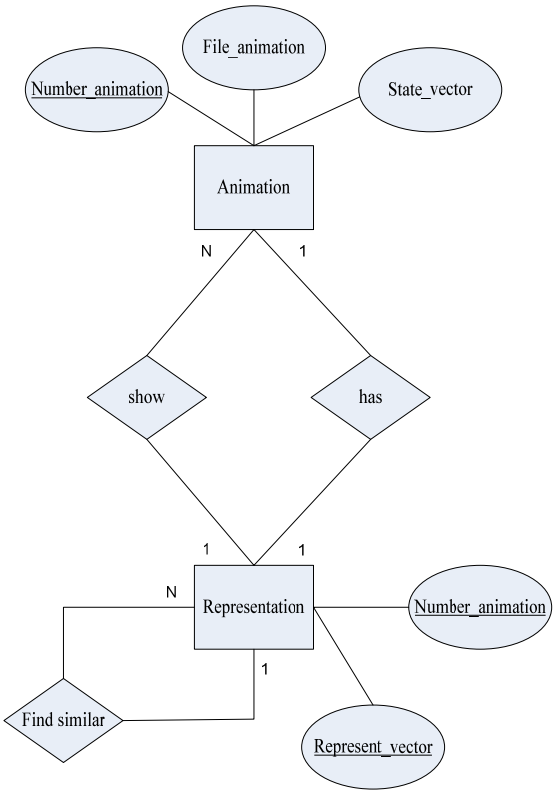

Figure 6. Show E-R model show relation of database that consists of entity "Animation" which has Primary key is Number_animation and entity "Representation" which primary key is Number_animation and Representation.

Representation of each motion different depends on number of keyframes in each cartoon motion. This number of keyframes is flexible by number of motion .Relation of each table can be explained by E-R Model in figure 7 which relation of entity "Animation" and entity "Representation" is 1:1. In finding representation of state vector of motion to store into database and 1: N in getting alike query motion to show and entity "Representation" entity has relation 1: N with itself in finding representation that alike as query motion.

The retrieval of biped cartoon motion is tested by using state vector instead of keyframe of cartoon. Then using vector quantization by LBG algorithm. We tested by storing 30 motions into database.

Database had several biped cartoon motions which were synthetic data.

In retrieval of cartoon motions, we retrieve from motion of cartoon which is stored 30 examples. We get motion information from desired cartoon. In this case we use walking to search motion information which is the same or alike as motion information in database as in figure 7. By quantizing vector of query cartoon, the result is a codebook, and then we bring it to compare with each codebook in database. We compare the different distance of each codebook in database with the motion we need, using Euclidean distance. We show these values in table 1 and these values must not exceed 15 .We show result of desired motion in figure 8.

The motion retrievals	**The distance of each codebook.**
Motion 1	12.41
Motion 2	10.52
Motion 3	8.60
Motion 4	13.35
Motion 5	11.30

Table 1. Show value of the different distance of each codebook in database with the motion we need.

Then we compare output with input. We calculated the recall that was number of items retrieved and relevant and total relevant in motion database ratio and calculated the precision that was number of items retrieved and relevant and total retrieve ratio. We repeated this 100 time and consider the entire of the precision and recall which results are satisfactory.

6. CONCLUSION

In this paper we have proposed a method for biped cartoon motion retrieval using state vectors of keyframes as indexes. Input motion give state vector is quantized by enhanced LBG Algorithm, developed by our, to calculate indexes for set of input key frame. Because of the motion of cartoons which have difference the number of keyframes, are stored in database. If we define the number of codewords are constant, the result of codewords are calculated is not flexible to number of frame. The result is the retrieval of cartoon motion that is more accurately and automatically.

In the future works, we will try to speed up the computation of the state vector and the codebooks

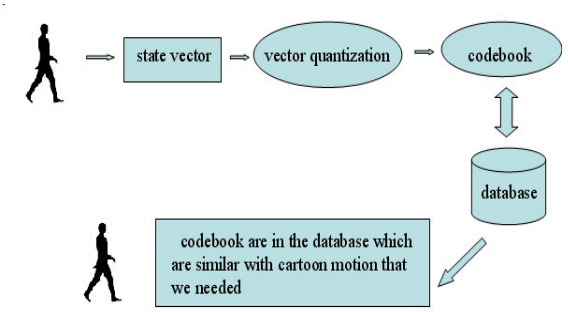

Figure 7. Show procedure of the retrieval.

If we use sorting techniques and searching techniques to sort and search the motion of cartoon, the response time to find the result of the retrievals would be decrease. We compare the response time of the retrieval that use our method, between differences searching technical as Rtree search algorithm and FastScan algorithm. The accuracy of retrieval by our method, depend on searching techniques by using same database. If the sizes of database are increase, the result could vary.

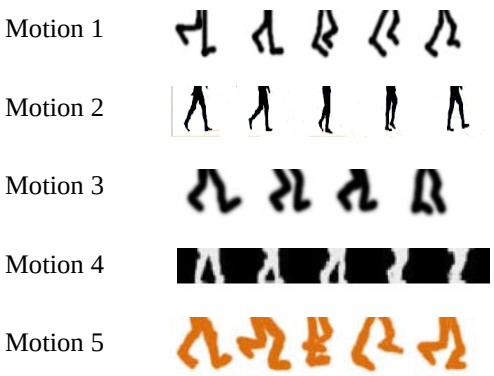

Figure 8. Show example output of walking motion.

7. REFERNCES

[Ari03a] O. Arikan, David A., F. James and F. O'Brien. Motion Synthesis from Annotations. ACM Transactions on Graphics, pp. 402 – 408, 2003.

[Bas05a] S. Basu, S. Shanbhag and S. chandran. Search and Transitioning for Motion Captured Sequences. Proceedings of the ACM symposium on Virtual reality software and technology VRST '05, pp. 220–223, 2005.

[Edu05a] T. Edumunds, S. Muthukrishnan, S. Sadhukhan and S. Sueda. MoDB: Database System for Synthesizing Human Motion. Proceedings of the 21st International Conference on Data Engineering (ICDE'05), pp. 1131–1132, 2005.

[For05a] K. Forbes and E. Fiume. An Efficient Search Algorithm for Motion Data Using Weighted PCA. Proceedings of the 2005 ACM SIGGRAPH / Eurographics symposium on Computer animation, pp. 67 – 76, 2005.

[Keo04a] E. J. Keogh, T. Palpanas, V. B. Zordan,

D. Gunopulos, and M. Cardle. Indexing large human-motion databases. In VLDB, pp. 780–791, 2004.

[Liu05a]G. Liu, J. Zhang, W. Wang, and L. McMillan. A System for Snalyzing and Indexing Human-Motion Databases. In SIGMOD '05: Proceedings of the 2005 ACM SIGMOD international conference on Management of data, pp. 924–926, 2005.

[Lin80b] Y. Linde, A. Buzo and R. Gray. An Algorithm for Vecter Quantizer Design. IEEE Transactions on [legacy, pre - 1988], Vol. 28, pp. 84-95, 1980.

[Non05a] A. Nongnuch, A. Surarerks. An Algorithm for Vector Quantization Using Density Estimation. Master's thesis, Faculty of Engineering, Chulalongkorn University, 2005.

[Pat01a] G. Patanè. Unsupervised Learning on Tradition and Distributed Systems. Master's thesis, University of Palemo, 2001.

[Sha97a] J. Shanbehzadeh and P. O. Ogunbona. On the Computational Complexity of the LBG and PNN Algorithms. IEEE Transactions on Image processing, Vol. 6, No. 4, pp. 614-616, 1997.

[Gra84a] R. Gray. Vector Quantization, IEEE ASSP Magazine, pp. 4-29, 1984.