Automatic Composition of Motion Capture Animation for Music Synchronization

Jianfeng Xu, Koichi Takagi and Ryoichi Kawada

Media Solutions Laboratory, KDDI R&D Laboratories Inc., Japan

Abstract

To enrich the music experience, automatic generation of a CG music visualizer is attracting more and more attention, where 3D animation is composed to synchronize with the music using motion capture data. In this paper, we present a novel approach for the above purpose, where both beat and intensity are employed to synchronize the motion with the music. We extend the conventional unstructured motion graphs to structured motion graphs (called weighted motion graphs*) using motion beat and intensity, where a best path is searched by dynamic programming to obtain beat-level synchronization. Our objective function is designed for the following three aspects: motion quality, cost from beat synchronization, and cost from intensity synchronization. Our experiments with a user study demonstrate that the proposed approach can effectively generate attractive animations for music synchronization with much less computational cost than the state-of-the-art alternative.*

Categories and Subject Descriptors (according to ACM CCS): I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Animation J.5 [Arts and Humanities]: Arts and Humanities—Performing Arts

1. Introduction

Music and dance are two kinds of main entertainment in our daily life. Moreover, the fact of people's dancing to the music infers the possibility of synchronizing the human motion with the music. Recently, some researchers [KPS03, SNI06] have developed the techniques to generate a dancing motion from a motion capture (MoCap, a sequence of human poses in an articulated skeleton) database, which is synchronized with a piece of music. These techniques can be used in a music visualizer to enrich the music experience. Similarly, the goal of this paper is to generate a CG music visualizer by synchronizing the motion with the music from a MoCap database, where your favorite character can dance to your favorite music automatically.

2. Related Works & Problems

To synchronize the human motion with the music, Kim et al. [KPS03] only use beat information while Shiratori et al. [SNI06] employ both beat and intensity information in their system to improve the performance further, which, therefore, is compared in this paper. Basically, *motion beats*, whose concept is borrowed from music beats, are defined as the regular moments when the movement is changed significantly in direction or magnitude [KPS03]. And *motion intensity* expresses the excitement of motion [SNI06], e.g., which may be calculated as the kinetic energy [OFH08].

In [SNI06], the input music is divided into segments (with a range from several seconds to about one minute) by the repeating patterns, and then candidate motion segments are searched whose rhythm features are matched to those of each music segment by a threshold, and finally the motion is found whose intensity is similar to that of music segments. However, there are two main problems in [SNI06].

1. In such a multi-stage approach, the computational cost increases greatly when generating a longer motion, i.e., in the case of more music segments.
2. Furthermore, the granularity of synchronization is obviously in the segment level, which has the difficulty in controlling the connection quality in the generated motion.

In this paper, we propose an approach whose computational complexity is linear to the music length. Moreover, our approach reaches beat-level synchronization, which improves the control granularity further.

3. Proposed Method

In this section, we describe our system for synchronizing the human motion with the music using a MoCap database. Note that full automatic processing is supported. As shown in Figure 1, our system is composed of two parts. In the first part, which is performed off-line, we re-organize the MoCap database into weighted motion graphs by the motion beat and intensity. In the second part, a human motion is generated on-the-fly from the graph to synchronize with the input music and then a CG character is rigged for rendering.

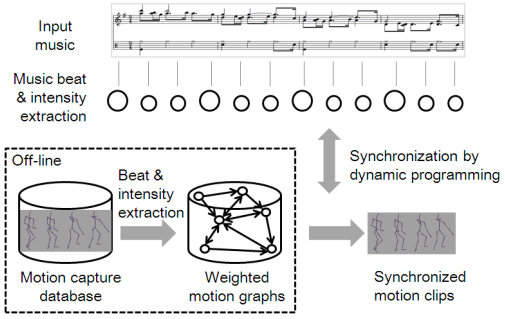

Figure 1: *System framework of generating a motion that is synchronized with the input music. Weighted motion graphs are constructed off-line and a best path is searched in the graph to generate a synchronized motion.*

3.1. Construction of Weighted Motion Graphs

The basic idea for music synchronization is re-use of MoCap data, where motion graphs [KGP02] are a powerful data structure. Many related works have been reported as briefly surveyed in [Zha09]. Basically, a motion graph is a directed graph structure of possible connections between the motions, turning the set of initial clips from a list of sequences into a connected graph of movements. However, a well-known problem for motion graphs is that there may not be any control over the structure of the graph, leading to destroying rhythmic structure in motions and a hard searching issue [GSKJ03]. Furthermore, the conventional concept of motion graphs cannot be directly adopted to synchronize the motion with the music due to the absence of necessary features such as beat and intensity. Therefore, we propose a novel data structure of weighted motion graphs using beat and intensity information, solving both the above problems.

Figure 2 shows the procedure of constructing our weighted motion graphs. Firstly, we separate the entire database into sub-groups by the motion genres (from the annotations in the database, see [CMU03]) and motion tempos (from beat induction, see details in [XTY10]). Namely, a weighted motion graph G is constructed in a sub-group where the motions are in the same motion genre and share a range of tempos. Here G is represented as $\{V,E,W\}$ for the set of nodes, edges, and edge weights, respectively. Secondly, we extract the frames at beat instants (called

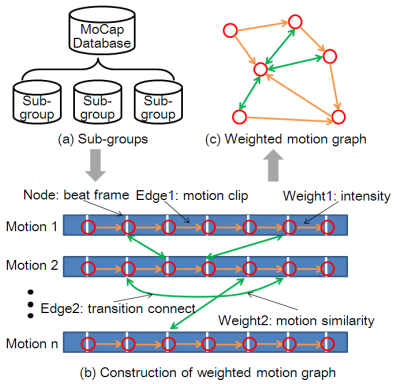

Figure 2: *Construction of weighted motion graphs, which keeps the rhythmic structure and reduces the computational costs. Only beat frames are checked for transition possibility. Edge weights are assigned for music synchronization.*

beat frames) in each motion (see details in our previous work [XTY10]), all of which are regarded as the set V.

Thirdly, we connect two successive beat frames F_B^i and F_B^{i+1} by a mono-directional edge $e^1(F_B^i,F_B^{i+1})$ and assign the motion intensity between the beat frames as the edge weight $w^1(F_B^i,F_B^{i+1})$, which can be calculated by the total kinetic energy between the beat frames [OFH08]. For those beat frames F_B^i and F_B^j with similar poses that satisfies Equation 1, we connect them by a bi-directional edge $e^2(F_B^i,F_B^j)$ and calculate the edge weight $w^2(F_B^i,F_B^j)$ as Equation 2. Obviously, the bi-directional edge takes no time and is only used with the following mono-directional edge $e^1(F_B^j,F_B^{j+1})$.

$$rd \equiv \frac{d(t_B^i,t_B^j)}{d(t_B^i-1,t_B^i+1)+d(t_B^j-1,t_B^j+1)} < TH \qquad (1)$$

$$w^2(F_B^i,F_B^j) = \begin{cases} rd & \text{if } rd > 2 \\ 2 & \text{others} \end{cases} \qquad (2)$$

where t_B^i denotes the frame index of beat frame F_B^i, the frame distance $d(t_B^i,t_B^j)$ can be calculated by such definitions as [WB03] and TH is a threshold, which controls the number of bi-directional edges. Here we compare the frame distance of two beat frames with those of their neighboring frames to decide if the beat frames are similar or not.

Now, a weighted motion graph is constructed for music synchronization, where V consists of all the beat frames, E consists of two subsets including the sets of mono-directional edges E^1 and bi-directional edges E^2, and W has two corresponding subsets, i.e., W^1 for E^1 and W^2 for E^2.

Although we only compare the beat frames, most of the meaningful connections are preserved. For example, in two walking motions as shown in Figure 3, the corresponding poses (e.g. hitting ground) in different cycles are connected as desired. The greatest benefit of our weighted motion graphs is to obtain a structured graph instead of an un-

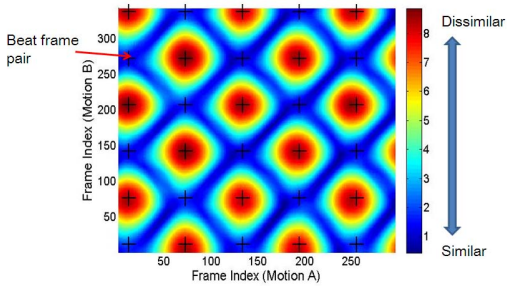

Figure 3: *Distances between frame pairs in two walking motions. Crosses denote the beat frame pairs, which locate the peaks and nadirs regularly, inferring that the corresponding poses in different cycles are properly found.*

structured graph in the conventional method, which keeps the rhythmic structure and reduces the computational costs in both construction and searching stages. This characteristics provides the ability of music synchronization. Moreover, since we construct the weighted graph in a sub-group, our system is flexible for additional motions in the database.

3.2. Motion Synthesis

Given a piece of music, we firstly extract music beats using existing techniques such as [Dix01] and music intensity which may be the sound pressure level between two beat instants. Then, we select a weighted motion graph according to the music genre and music tempo. Lastly, a best path is searched in the graph by dynamic programming beat by beat. Here it is essential to properly define an objective function.

Basically, three requirements should be met in the generated motion. Firstly, beat instants in the generated motion should be synchronized with those in the music. Because the beat frames are and only are in the node set V, we can exactly synchronize the beat instants in any path with those in the music by modifying the frame rate between two nodes in the path. In other words, there is no time difference between the corresponding beat instants in the music and motion. Therefore, the cost of beat synchronization is controlled to zero, i.e., the granularity of synchronization is in the beat level. Secondly, the motion intensity should be matched to that in the music, which is minimized by dynamic programming. Thirdly, the motion should be as natural as possible, where the source of motion artifacts is the bi-directional edges. As a result, the cost function of an edge $e(F_B^i,F_B^j)$, which is a mono-directional edge $e^1(F_B^i,F_B^j)$ or a bi-directional edge $e^2(F_B^i,F_B^j)$ with its following mono-directional edge $e^1(F_B^j,F_B^{j+1})$, is defined as:

$$edgeCost(e(F_B^i,F_B^j),k) = \begin{cases} \| \overline{w^1}(F_B^i,F_B^j) - \bar{I}(k) \| & \text{if } e(F_B^i,F_B^j) \in E^1 \\ w^2(F_B^i,F_B^j) \| \overline{w^1}(F_B^j,F_B^{j+1}) - \bar{I}(k) \| & \text{if } e(F_B^i,F_B^j) \in E^2 \\ \infty & \text{others} \end{cases} \quad (3)$$

where $\bar{I}(k)$ denotes the music intensity from the k-th beat to $(k+1)$-th beat or the k-th beat interval, which is normalized in the entire music, $\overline{w^1}$ denotes the normalized edge weight in the set of W^1, and w^2 denotes the edge weight in the set of W^2. Note that standard scores are calculated for the normalization and w^2 is a penalty for selecting a bi-directional edge to avoid worsening the motion quality.

Table 1: *Comparison of computational cost (seconds). The time of the off-line process is excluded.*

		Song004		Song091	
		30s	60s	30s	60s
Break	Conventional	735	2103	6039	15027
	Proposed	27	56	40	80
Indian	Conventional	284	757	2025	4888
	Proposed	3	7	5	9

Then, assuming we have K beat intervals in the music, our objective function is $\min \sum_{k=1}^{K} edgeCost(e(F_B^i,F_B^j),k)$, which can be optimized by dynamic programming as shown in Equations (4)-(6). At least one initial node is required in dynamic programming. We use the first beat frames of all the motions in the graph as multiple initial nodes $InitS$.

$$P(F_B^v,0) = \begin{cases} 0 & if \quad F_B^v \in InitS \\ \infty & others \end{cases} \quad (4)$$

$$P(F_B^v,k) = \min_{F_B^i \in V} \{P(F_B^i,k-1) + edgeCost(e(F_B^i,F_B^v),k)\} \quad (5)$$

$$P(K) = \min_{F_B^v \in V} \{P(F_B^v,K)\} \quad (6)$$

where $P(F_B^v,k)$ denotes the cost of a best path for the first k beat intervals with the last node of F_B^v, and $P(K)$ is the cost of a best path for the entire music. Equation (5) shows that the current best path with the last node of F_B^v is searched based on the previous best path whose last node is any node F_B^i. In the standard dynamic programming, the cost function is the edge weight that is constant. In our task, the cost function is dynamic, which is updated with the music intensity for the k-th beat interval as Equation 3. Suppose the average indegree is D and the node number is N in the graph, we only need to compare DNK times to search the best path, which is linear to the number of beat intervals K in the music. In the worst case of full graph, the complexity is $O(N^2K)$. However, the constructed graphs are rather sparse because only beat frames are permitted to connect together.

4. Experimental Results

Experimental Conditions: Two pieces of music are tested from RWC music database [GHNO02] with the motions from CMU MoCap database [CMU03] including the Break and Indian dances. All the experiments are performed on the same PC with a Core2® Duo 2.2 GHz CPU and a 4 GB RAM memory. The proposed method is compared with the conventional method [SNI06], where the beat and intensity information are shared in the two methods to focus on the

Table 2: *Mean Opinion Scores (MOSs) from the participants for all the three questions. The differences of MOSs from the conventional method are in the parentheses. Font colors show the results of t-test, where $p < 0.05$ for red, $p < 0.10$ for blue, and $p \geq 0.10$ for black.*

		Q1	Q2	Q3
Song004	Break	3.7(+0.5)	3.4(+0.5)	3.5(+0.5)
	Indian	3.4(+0.2)	3.4(+0.2)	3.6(+0.6)
Song091	Break	4.3(+1.1)	4.0(+0.8)	3.8(+1.1)
	Indian	3.0(-0.4)	3.4(+0.2)	3.4(+0.5)

motion synthesis. In our method, the threshold in Equation 1 is set as 3.0. In [SNI06], the length of music segments is set as 10s and the threshold for candidate motion segments is set as 1.0 according to our preliminary investigation.

Computational Cost: (see details in Table 1) It takes less than 10 seconds to generate a 60s animation in our method for the Indian dance. The average speedup is 169.5 times for 30s contents and 219.2 times for 60s contents, which shows our method is much faster than [SNI06]. When the music doubles in length, computing time of our method is averagely 2.05 times and that of Shiratori et al. [SNI06] is averagely 2.61 times, inferring that the computational cost increases faster in [SNI06] than ours. Moreover, the computational cost in our method is predictable given the music length and motion genre (see the analysis in Section 3.2) while the time varies much in [SNI06] due to the number of candidate motion segments depends on the music.

User Study: Totally 11 participants have evaluated the above 30s contents by scoring the three questions about beat synchronization (Q1), intensity synchronization (Q2), and motion quality (Q3), respectively, in the range of 1 to 5. By randomly displaying the results, participants do not know which is generated by which method. Table 2 shows the Mean Opinion Scores (MOSs, average scores) [Win05]) from all the participants, which demonstrates our method can generate plausible animations that are well synchronized with the music. A t-test is performed between the two methods, where p-value is less than 0.05 for those with red fonts and less than 0.10 for those with blue fonts in Table 2. The results show that our method, while reducing the computational cost greatly, still maintains the quality. In addition, in the case of Song091 and Break dance, our method is significantly better than the conventional method [SNI06]. This is an effect of the weighted motion graph. For further reference, please see the attached video, where not only are beats synchronized but intensity is also matched well in our method (especially Song091 and Break dance).

5. Conclusions and Discussions

In this paper, our main contribution is as follows.

- We propose a novel scheme for motion synchronization with music using a MoCap database, which is (1) much faster (speedup of 27 $\sim$ 543 times) and (2) controlled in finer granularity than [SNI06]. Three aspects are considered including motion quality, the costs from beat and intensity synchronization respectively. Our experimental results show the promising performance of our scheme.
- We extend the conventional concept of motion graphs into weighted motion graphs, which makes it possible to adopt dynamic programming in the searching stage. Such a weighted motion graph provides a structured graph, which keeps the rhythmic structure of motions.

Although automatic generation is supported in our system, user interaction is also provided to give the users the opportunity of selection motion genre, CG character, etc. to further satisfy the users. In addition, it is observed in our user study that not all the beat instants are of the same importance for human perception. In the future, we plan to estimate the importance of beat frames in both music and motion to match the human perception better.

References

[CMU03] CMU MoCap Database: http://mocap.cs.cmu.edu/, 2003.

[Dix01] Dixon S.: Automatic extraction of tempo and beat from expressive performances. *Journal of New Music Research 30*, 1 (Mar. 2001), 39–58.

[GHNO02] Goto M., Hashiguchi H., Nishimura T., Oka R.: Rwc music database: Popular, classical, and jazz music databases. In *ISMIR '02: Proc. of the 3rd International Conference on Music Information Retrieval* (2002), pp. 287–288.

[GSKJ03] Gleicher M., Shin H. J., Kovar L., Jepsen A.: Snap-together motion: assembling run-time animations. In *I3D '03: Proc. of the 2003 symposium on Interactive 3D graphics* (2003), pp. 181–188.

[KGP02] Kovar L., Gleicher M., Pighin F.: Motion graphs. *ACM TOG 21*, 3 (July 2002), 473–482.

[KPS03] Kim T. H., Park S. I., Shin S. Y.: Rhythmic-motion synthesis based on motion-beat analysis. *ACM TOG 22*, 3 (July 2003), 392–401.

[OFH08] Onuma K., Faloutsos C., Hodgins J. K.: FMDistance: A fast and effective distance function for motion capture data. In *Proc. of EUROGRAPHICS 2008, Short Papers* (2008).

[SNI06] Shiratori T., Nakazawa A., Ikeuchi K.: Dancing-to-music character animation. *Computer Graphics Forum 25*, 3 (July 2006), 449–458.

[WB03] Wang J., Bodenheimer B.: An evaluation of a cost metric for selecting transitions between motion segments. In *SCA '03: Proc. of the ACMSIGGRAPH/Eurographics Symposium on Computer Animation* (2003), pp. 232–238.

[Win05] Winkler S.: *Digital video quality: vision models and metrics*. John Wiley and Sons, 2005.

[XTY10] Xu J., Takagi K., Yoneyama A.: Beat induction from motion capture data using short-term principal component analysis. *Journal of the Institute of Image Information and Television Engineers 64*, 4 (Apr. 2010), conditionally accepted.

[Zha09] Zhao L.: *Constructing Good Quality Motion Graphs for Realistic Human Animation*. Ph.D. Dissertation in the University of Pennsylvania, 2009.