Using Musical Beats to Segment Videos of *Bharatanatyam Adavu*'s

Tanwi Mallick*, Aakash Anuj, Partha Pratim Das, and Arun Kumar Majumdar

Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur 721302, India
tanwimallick@gmail.com,aakashanuj.iitkgp@gmail.com
ppd@cse.iitkgp.ernet.in,akmj@cse.iitkgp.ernet.in

Abstract. We present an algorithm for audio-guided segmentation of the Kinect videos of *Adavu*'s in *Bharatanatyam* dance. *Adavu*'s are basic choreographic units of a dance sequence in *Bharatanatyam*. An *Adavu* is accompanied by percussion instruments (Tatta Palahai (wooden stick) - Tatta Kozhi (wooden block), Mridangam, Nagaswaram, Flute, Violin, or Veena) and vocal music. It is a combination of *events* that are either postures or small movements synchronized with rhythmic pattern of beats or *Taal*s. We segment the videos of *Adavu*'s according to the percussion beats to determine the events for recognition of *Adavu*'s later.

We use *Blind Source Separation* to isolate the instrumental sound from the vocal. Beats are tracked by onset-detection to determine the instants in the video where the dancer assumes key-postures. We also build a visualizer for test. From over 13000 input frames of 15 *Adavu*'s, 74 of the 131 key-frames actually present get detected. Every detected key-frame is correct. Hence the system has 100% precision, but only about 56% recall.

Keywords: Music driven dance video segmentation, multimodal Indian classical dance data captured by Kinect, Onset detection on Indian music, music-to-dance video synchronization

1 Introduction

India has a rich tradition of classical dance. *Bharatanatyam* is one of the eight Indian classical dance forms. *Adavu*'s are basic choreographic units that are combined to form a dance sequence in *Bharatanatyam*. These *Adavu*'s are performed in synchronization with rhythmic pattern of beats known as *Taal*. The *Adavu*'s are classified according to the style of footwork employed and the *Taal* on which they are based (synchronized).

* Please note that the LNCS Editorial assumes that all authors have used the western naming convention, with given names preceding surnames. This determines the structure of the names in the running heads and the author index.

Every *Adavu* of *Bharatanatyam* dance is a combination of events, which are either *Key Postures* or *Short yet Discrete Movements*. These events are synchronized with the *Taal*. Our objective here is to find the beat pattern or *Taal* from the audio of the musical instrument and locate the corresponding events of the *Adavu*'s. The beat detection and *Taal* identification from an *Adavu* leads to meaningful segmentation of *Bharatanatyam* dance. We propose to adapt an algorithm for onset detection to achieve effective segmentation of videos of *Adavu*'s into events. We also build a visualizer to validate segmentation results.

After an overview of related work in Section 2, we explain the concept of *Taal* in Section 3. The methodology of our work is outlined in Section 4 followed by the elucidation of data capture and data sets in Section 5. *Blind Source Separation (BSS)* to segregate the instrumental (typically, percussion) sound from the vocal music is discussed in Section 6. The beat tracking / onset detection for the audio to locate the events in the corresponding video are elaborated in Section 7 followed by video segmentation and visualization in Section 8. We talk of the results in Section 9 and conclude in Section 10.

2 Related Work

Indian classical music, as used in Indian classical dance like Bharatanatyam, is based on a sophisticated rhythmic framework, where the rhythmic pattern or *Taal* describes the time-scale. Beat detection and *Taal* recognition are challenging problems as Indian music is a combination of instrumental audio and vocal speech. Several attempts [8], [6] have been made to separate the audio streams into independent audio sources without any prior information of the audio signal. Further, several researchers have worked [3], [4], [5], [9], and [11] to extract the rhythmic description in music through various *Beat Tracking algorithms* to extract the long-duration as well as the short-duration rhythmic structures. *Onset Detection* is a dominant and effective approach for *Beat Tracking*. In [1], Bello et. al. present a nice tutorial on *Onset Detection in Music Signals*.

There is, however, no work that use the rhythms of music to identify key body postures in videos of Indian Classical Dance.

3 Concept of *Taal* in *Bharatanatyam*

Adavu's are performed along with the rhythmic syllables played in a particular *Taal* or rhythmic pattern of beats that continues to repeat in cycles. Rhythm performs the role of a timer. Between the interval of beats, the dancer changes her posture. We define these as *Key Postures*. The sequence of frames between two Key Postures corresponding to two consecutive beats is defined as an *Event* that depicts a primitive Audio-Visual correspondence in an *Adavu* (Figure 1).

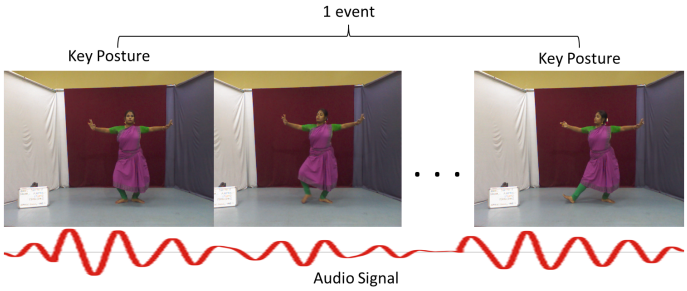


Fig. 1. An Event in a *Bharatanatyam Adavu*

4 Methodology

We intend to track beats in the audio stream to determine the time-instant of the beat and then to extract the RGB frame corresponding to the same instant to determine the events of an *Adavu* video. The steps are (Figure 2):

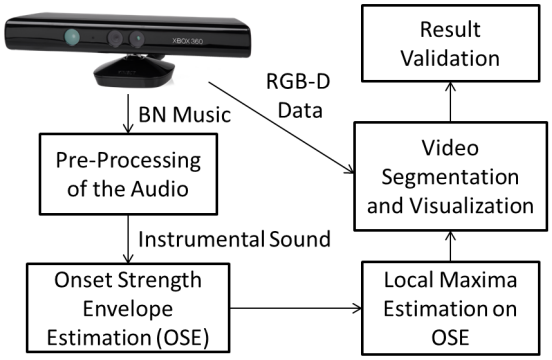


Fig. 2. Flowchart of *Bharatanatyam* Video Segmentation

1. Use *Non-diagonal Audio Denoising* [12] through adaptive time-frequency block thresholding to denoise the audio stream.
2. Extract different sources from the audio stream by *Blind Source Separation (BSS)* [8], [10]. Select the instrumental sound for further analysis.
3. Estimate the *Onset Strength Envelope (OSE)* [3].
4. *Onset Detection* is done on the OSE to estimate the time-instant of a beat. Before using *Onset Detection*, dynamic programming from [3] was tried to compute the time-instant of a beat. This often detected more beats than were actually there. Hence, we improved the algorithm from [3] by finding local maxima in OSE to estimate onset.

5. Extract the video frame at *Onset* or the estimated time-instant of a beat. This gives the Key Postures and segments the video. A tool is built to visualize segments.
6. Match the results with the segmentation by experts.

5 Capturing Data Sets

We recorded *Adavu*'s using *nuiCapture*[1] at 30 fps. RGB, skeleton, audio and depth streams were captured for 15 *Adavu*'s using Kinect for Windows. These 15 *Adavu*'s – *Tatta*, *Natta*, *Utsanga*, *Tirmana*, *Tei Tei Dhatta*, *Sarika*, *Pakka*, *Paikkal*, *Joining*, *Katti/Kartari*, *Kuditta Nattal*, *Mandi*, *Kuditta Mettu*, *Kuditta Tattal*, and *Sarrikkal* – together cover all constituent postures and movements of *Bharatanatyam*. All 15 *Adavu*'s are used in our experiments.

Each *Adavu* was recorded separately by 3 dancers to study individual variability.

6 Blind Source Separation

The recorded audio streams are often noisy. So we first need to denoise the stream. Audio denoising aims at attenuating environment and equipment noise while retaining the underlying signals. We use *Non-diagonal Audio Denoising* through adaptive time-frequency block thresholding by Cai and Silverman [12]. We find that this is effective in reduction of noise in musical streams. Next we perform source separation.

The musical (beating) instrument used for an *Adavu* is a *Tatta Palahai* (wooden block) and a *Tatta Kozhi* (wooden stick). This is played alongside the vocal sound and is mixed. We separate the sound of the instrument (has *beats*) from the vocal music using *Flexible Audio Source Separation Toolbox* (*FAAST*) [10], [8]. It was able to segment the audio stream into 4 parts – *Melody*, *Bass*, *Drums*, and *Other sources*. We selected the *Drums* as we need the beating instrument. Experiments with our *Adavu* videos show good separation for the beating sound even in the presence of multiple instruments.

7 Beat Tracking

We attempt to track the beats from the denoised audio stream using two methods as discussed below.

7.1 Method 1. Beat Tracking by Dynamic Programming

We first explore the beat tracking algorithm by Ellis [3]. It starts with an estimation of a global tempo to construct a transition cost function, and then uses dynamic programming to find the best-scoring set of instants for beats that reflect the tempo as well as correspond to moments of high *onset strength* derived from the audio. This goes as follows:

[1] nuiCapture [2] on Windows records and analyzes Kinect data.

Onset Strength Envelope (OSE) is calculated as:

- Audio is re-sampled at 8KHz, and then STFT[2] (spectrogram) is calculated using 32 ms windows and 4 ms advance between frames.
- This is then converted to an approximate auditory representation by mapping to *40 Mel bands* via a weighted sum of the spectrogram values.
- The Mel spectrogram is converted to dB, and the first order difference along time is calculated in each band. Negative values are set to zero (half wave rectification), then the remaining, positive differences are summed up across all frequency bands.
- This signal is passed through a high-pass filter with a cut-off around 0.4Hz to make it locally zero-mean, and is smoothed by convolving with a Gaussian envelope of about 20ms width. This gives a *1D OSE* as a function of time that responds to proportional increase in energy summed across approximately auditory frequency bands.

Tempo Period (TP) is the inter-beat interval, τ_p. Auto-correlation of the OSE $O(t)$ is computed to reveal the regular, periodic structure of the *Tempo Period Strength* (TPS) by: $TPS(\tau) = W(\tau)\sum_t O(t)(t-\tau)$, where $W(t)$ is a *Gaussian Weighting Function* on a log-time axis. The τ for which $TPS(\tau)$ is largest, is then the estimate for τ_p.

Dynamic Programming (DP) Given OSE and TP, we can find the sequence of time instants for beats that correspond to both the perceived onsets in the audio signal and also constitute a regular, rhythmic pattern in them. The objective function $C(t_i) = \sum_{i=1}^{N} O(t_i) + \alpha \sum_{i=2}^{N} F(t_i - t_{i-1}, \tau_p)$ combines both these goals, where t_i is the sequence of N beat instants found out by the beat tracker, $O(t)$ is the OSE, τ_p is the TP, α is a weight to balance the relative importance, and $F(.,.)$ is a function that measures the consistency between the inter-beat interval and the ideal spacing τ_p defined by the target tempo. We use a simple squared-error function $F(\Delta t, \tau) = -(\log \frac{\Delta t}{\tau})^2$ applied to the log-ratio of actual and ideal time spacing.

For the objective function above the best scoring time sequence can be assembled recursively to calculate the best possible score $C^*(t) = O(t) + \max_{r=0\ldots t}\{\alpha F(t-\tau, \tau_p) + C^*(t)\}$, of all sequences that end at time t.

This follows from the fact that the best score for time t is the local onset strength, plus the best score to the preceding beat time τ that maximizes the sum of that best score and the transition cost from that time. In the process, the actual preceding beat that gave the best score is also recorded as: $P^*(t) = O(t) + \arg\max_{\tau=0\ldots t} \alpha F(t-\tau, \tau_p) + C^*(t)$.

To find the set of optimal beat times for an OSE, C^* and P^* are computed for every time starting from zero. The largest C^* forms the largest beat instant.

[2] Short-Time Fourier Transform

Next we backtrack via P^*, find the beat time $t_{N-1} = P^*(t_N)$, and continue backwards till the beginning to get the entire beat sequence $\{t_i\}^*$.

The DP performs well only for a limited set of *Taal*s as used in *Bharatanatyam*. This is because it assumes that the beats reflect a locally constant inter-beat interval. This is not true for all *Bharatanatyam Taal*s, and any two consecutive onsets might have variable time gaps between them. Figure 3 shows a *Taal*, where the beats/onsets are not equally separated.

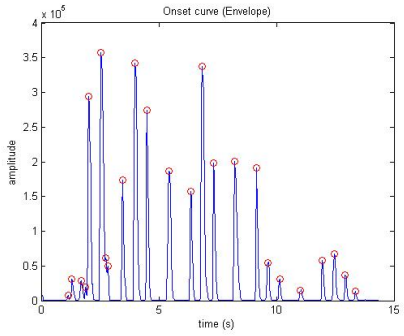


Fig. 3. Unequal separation of the onsets

The DP solutions leads to the over-detection of beats. This is not acceptable, since we only want good onsets corresponding to salient body postures in the dance. Hence, we propose the method of local maxima detection.

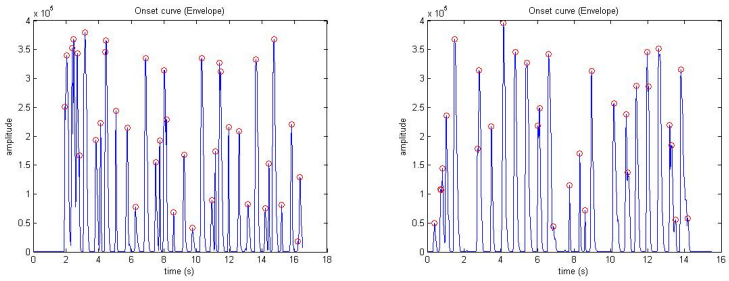
7.2 Method 2. Detection of Local Maxima in OSE

Our proposed method uses the OSE found earlier. We detect the local maxima in the envelope. The local maxima would correspond to the key postures. Figure 4 shows the detection on onsets for the *Utsanga* and *Tirmana Adavu*s.

Avoiding Over-detection of Local Maxima Naive detection of local maxima usually leads to over-detection. To avoid this, a given local maximum is considered as a peak if the difference of amplitude with respect to both the previous and successive local minima (when they exist) is higher than a threshold *cthr* (0.1, by default). This distance is expressed with respect to the total amplitude of the input signal. A distance of 1, for instance, is equivalent to the distance between the maximum and the minimum of the input signal.

This is implemented from MIRtoolbox [7] and illustrated in Figure 5.

Retaining Good Onsets It is important that we represent an *Adavu* by a minimal set of body key postures. If two local maxima are very close to each



(a) (b)

Fig. 4. Onset Detection in *Adavu*'s (a) *Utsanga* (b) *Tirmana*

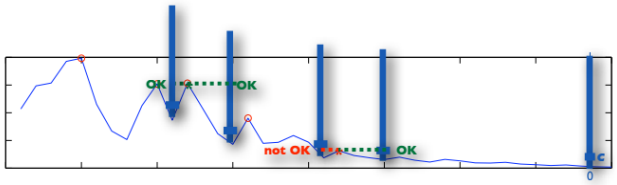


Fig. 5. Avoiding overdetection of local maxima

other in time (the difference being less than a threshold $tthr = 0.15$s), then there would be almost no change in the posture at the corresponding onsets. In such cases, we retain the maxima with the higher peak. A maxima with a higher peak corresponds to an onset with higher confidence. Figure 6(b) and 6(d) show the removal of unwanted local maxima for the *Utsanaga* and *Tirmana Adavu*.

8 Video Segmentation and Visualization

Next we use the detected beat instants to segment the videos into events and visualize the key postures in them.

8.1 Segmentation into Events

Since the recording has been done at 30 fps, we know the time stamp for each frame in the RGB, skeletal or depth stream by the frame number. Hence, given the onset times of beats in the audio stream we can find the corresponding frames (frame numbers) at the onset times by simple temporal reasoning. The frame number corresponding to an onset time t would be $(30 \times t)$, where t is in seconds. Since $(30 \times t)$ might be a floating point value, we round it off to the nearest integer and obtain the corresponding frames for RGB, depth and skeleton.

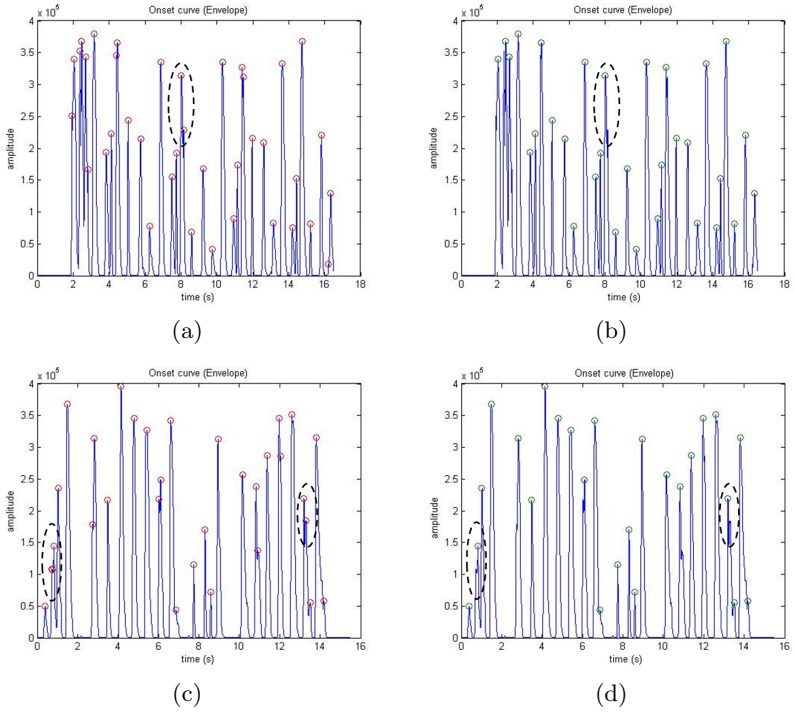


Fig. 6. Retaining Good Onsets (a) Detection in *Utsanga* (b) Retention (c) Detection in *Tirmana* (d) Retention

8.2 Visualization of Key Postures

A visualization tool has been built to view the correspondence between the onsets and the key posture frames. This helps us to validate if the postures selected are actually those at the onsets of the audio signal. Using this tool we select any of the onset points as given by local maxima detection. It then displays the corresponding RGB frame. Figure 7 shows a snapshot of the tool.

9 Results and Discussion

Using the visualizer we have tested the method for videos of 15 *Adavu*'s. In total, 74 key posture frames were detected by the system based on the onsets from a total of over 13000 frames in 15 videos. *Bharatanatyam* experts reviewed and verified that every detected key posture was indeed correct.

Independently, the experts were asked to identify key postures in the 15 videos. They manually inspected the frames and extracted 131 key posture frames from the 15 videos including the 74 key postures as detected above. So our system has 100% precision, but only about 56% recall.

Fig. 7. Visualizing Correpondence between Onset (audio) and RGB frame – *selecting an onset by mouse automatically takes one to the corresponding frame*

10 Conclusions

Here we have attempted segmentation of videos of *Adavu*'s in *Bharatanatyam* dance using beat tracking. We engaged a dynamic programming approach [3] using the global tempo period (uniform inter-beat interval) estimate and the onset strength envelope. It performed well only on some *Adavu*'s, while on the others, it over-detected beat instants due to the non-uniformity of inter-beat intervals for a number of *Taal*s.

We have adapted an algorithm for OSE with detection of local maxima to estimate beats. This does not need the assumption of global tempo period (uniform inter-beat interval) as in [3]. Further, we propose heuristics to avoid over-detection of onsets and retain only the good peaks to get a minimal sequence of key postures to represent an *Adavu*. From a set of onset times, we find the corresponding RGB (skeleton / depth) frames. We have also developed a visualization tool for validation.

We have tested the method for 15 *Adavu*'s. We find that our system has 100% precision, but only about 56% recall. So we need to strike a balance between the over-detection of the DP approach and the over-precision of the local maxima method. We also need to use the domain knowledge of the structure of *Bharatanatyam* to aid the segmentation.

Acknowledgment

The work of the first author is supported by TCS Research Scholar Program of Tata Consultancy Services of India.

References

1. Juan Pablo Bello, Laurent Daudet, Samer Abdallah, Chris Duxbury, Mike Davies, and Mark B. Sandler. A tutorial on onset detection in music signals. *Speech and Audio Processing, IEEE Transactions on*, 13, 2005.

2. Cadavid Concepts. nuiCapture Analyze. http://nuicapture.com/ Last accessed on 10-Jan-2016, 2016.
3. Daniel P.W. Ellis. Beat tracking by dynamic programming. *Journal of New Music Research*, 36, 2007.
4. Jonathan T. Foote and Matthew L. Cooper. Visualizing musical structure and rhythm via self-similarity. In *MULTIMEDIA '99 Proceedings of the seventh ACM international conference on Multimedia*, pages 77–80, 1999.
5. Jonathan T. Foote and Matthew L. Cooper. Media segmentation using self-similarity decomposition. In *Proc. SPIE Storage and Retrieval for Media Databases*, 2003.
6. Yun Li, K. C. Ho, and Mihail Popescu. A general flexible framework for the handling of prior information in audio source separation. *Biomedical Engineering, IEEE Transactions on*, 61, 2014.
7. Mathworks. Mirtoolbox: An innovative environment for music and audio analysis. http://www.mathworks.in/matlabcentral/fileexchange/24583-mirtoolbox Last accessed on 10-Jan-2016, 2016.
8. Alexey Ozerov, Emmanuel Vincent, and Frdric Bimbot. A general flexible framework for the handling of prior information in audio source separation. *Audio, Speech, and Language Processing, IEEE Transactions on*, 20, 2012.
9. Zafar Rafii and Bryan Pardo. Repeating pattern extraction technique (repet): A simple method for music/voice separation. *Audio, Speech, and Language Processing, IEEE Transactions on*, 21, 2012.
10. Yann Salaun. Flexible audio source separation toolbox(faast). http://bass-db.gforge.inria.fr/fasst/ Last accessed on 10-Jan-2016, 2016.
11. Ajay Srinivasamurthy, Gregoire Tronel, Sidharth Subramanian, and Parag Chordia. A beat tracking approach to complete description of rhythm in indian classical music. In *2nd CompMusic Workshop*, 2012.
12. Guoshen Yu, Stephane Mallat, and Emmanuel Bacry. Audio denoising by time-frequency block thresholding. *Signal Processing, IEEE Transactions on*, 56, 2008.