2013 IEEE/RSJ International Conference on
Intelligent Robots and Systems (IROS)
September 14-18, 2014. Chicago, IL, USA,

Rapidly Learning Musical Beats in the Presence of Environmental and Robot Ego Noise

David K. Grunberg[1] and Youngmoo E. Kim[2]

Abstract— **Humans can often learn high-level features of a piece of music, such as beats, from only a few seconds of audio. If robots could obtain this information just as rapidly, they would be more capable of musical interaction without needing long lead times to learn the music. The presence of robot ego noise, however, makes accurately analyzing music more difficult. In this paper, we focus on the task of learning musical beats, which are often identifiable to humans even in noisy environments such as bars. Learning beats would not only help robots to synchronize their responses to music, but could lead to learning other aspects of musical audio, such as other repeated events, timbrel aspects, and more. We introduce a novel algorithm utilizing stacked spectrograms, in which each column contains frequency bins from multiple instances in time, as well as Probabilistic Latent Component Analysis (PLCA) to learn beats in noisy audio. The stacked spectrograms are exploited to find time-varying spectral characteristics of acoustic components, and PLCA is used to learn and separate the components and find those containing beats. We demonstrate that this system can learn musical beats even when only provided with a few seconds of noisy audio.**

I. INTRODUCTION

When exposed to a novel piece of music, humans are often able to learn a lot about it after only a few seconds of audio. Even in very noisy environments such as dance clubs and bars, humans can often learn aspects of the music such as beats within a few moments, and can then use that information to influence their responses. Knowledge of beat locations can allow humans to synchronize dance motions with the music, while knowing what the beats sound like can allow humans to identify higher-level rhythmic structures within the piece and incorporate that information into their responses. For instance, if every other beat of a musical work sounds different (such as when alternating beats are played on different drums), humans may be able to identify the beats as belonging to meaningful categories such as 'on-beats' and 'off-beats'. Thus, the ability of humans to rapidly learn musical aspects such as beats is very helpful in allowing them to respond to music.

It would be useful for musical robots to also be able to learn aspects of music after only a few seconds of audio. For this paper, we are particularly focused on enabling them to learn musical beats. While algorithms enabling robots to find beat locations have been developed, few allow robots to learn the spectral characteristics that determine what a beat sounds like [1], [2]. The ability to learn how beats sound, though, could help robots in understanding higher-level rhythmic structure, just as it can help humans. This knowledge could be exploited to enable more sophisticated responses, such as by allowing the robot to differentiate between on-beats and off-beats and react accordingly. Algorithms for learning beats could also potentially be extended to learning other aspects of music, such as learning other repeating events and environmental characteristics. Finally, knowledge of how beats sound could potentially be fed back into a beat tracker to help the system learn what it should be listening for. As such, we are interested in teaching robots to learn musical beats in short time frames.

For optimal performance, a system designed to solve this problem must operate under several constraints. First, it must be robust to the nonstationary ego noise produced by a robot's motors [3]. Second, the system should be able to learn beats with a few seconds of audio. The faster a robot can determine this information, the faster it can start using these features to inform its own responses to music, with a correspondingly more responsive and therefore better performance. Humans are often able to perform this task in just a few seconds, and we would like for robots to do the same. Finally, the system should obtain both beat locations and time-varying spectral characteristics simultaneously. Chaining the tasks is possible–for instance, by first running a beat tracker, and then using a source separation algorithm on the beat frames–but can lead to propagation of error, where a mistake in the first step compounds in the second. Performing the two steps together reduces this risk.

We propose a system to solve this problem and simultaneously estimate beat locations and time-varying beat spectral characteristics. We use stacked spectrograms, spectrograms in which each column contains frequency bins from multiple instances in time, as well as a Probabilistic Latent Component Analysis (PLCA)-based decomposition technique for extracting the different components of a musical signal. The stacked spectrogram can be exploited to find the time-varying spectral characteristics of different parts of the signal, and the PLCA portion of the system can separate the beat component from the rest of the music as well as the robot's ego noise. As long as the beat components are relatively consistent, PLCA should be able to identify them despite the presence of ego noise. The proposed system is then evaluated on audio contaminated by noise from a state-of-the-art humanoid robot known as Hubo [4].

*This work was supported by NSF CNS-0960061 MRI-R2: Development of a Common Platform for Unifying Humanoids Research.

[1]David K. Grunberg is in the Department of Electrical and Computer Engineering, Drexel University, 3141 Market Street, Philadelphia, Pennsylvania, USA. dgrunberg@drexel.edu

[2]Youngmoo E. Kim is with the Faculty of the Department of Electrical and Computer Engineering, Drexel University, 3141 Market Street, Philadelphia, Pennsylvania, USA. ykim@drexel.edu

II. Literature Review

Numerous beat trackers have been developed for robots. Yoshii et al devised a system that utilizes a multiple-agent architecture to find beat locations, allowing a robot to step in time with music [1]. Kozima et al enabled a small, toy-like robot called Keepon to listen to beats so that it can perceive rhythm and then dance with children [5]. While systems such as these allowed for robots to react to beat locations, they did not allow for the robot to learn the spectral characteristics of the beats. The robots could therefore not use knowledge of how the beats sounded in their responses.

Weinberg et al enabled a robot to analyze a provided drum sequence and perform an appropriate response [6]. This system involved turn-taking between the robot and human performers, and so the robot did not need to deal with the effects of its own noise while listening to the human.

Murata et al developed a system to perform beat tracking in the presence of a robot [2]. Their algorithm uses semi-blind Independent Component Analysis to separate music from the scatting and singing sounds of the robot. This algorithm requires the noise signal to be known in advance, so while it is useful for digital noises such as a robot's voice, it is not as useful for a robot's motor noise, which is produced mechanically and varies from performance to performance.

One technique for dealing with ego noise is to try to remove it from the audio. Ince et al demonstrated that ego noise could be modeled and masked given prior knowledge of the acoustics of the room [3]. Our prior work showed that an adaptive filter for subtracting out noisy frequency subbands also improved beat tracking accuracy [7]. Oliveira et al used a variety of noise removal algorithms, including beamforming, to improve beat tracking performance [8]. These systems, however, are not only unlikely to remove all of the noise, but also risk removing some of the signal as well, which can hurt accuracy. Thus, rather than attempting to design an ideal *noise-removal* system, we instead focus our efforts on creating a *noise-robust* system, which can perform accurately even in the presence of noise.

III. Robot Platform

As it is important for the final system to be able to function in audio contaminated by robot ego noise, we have incorporated the Hubo robot (Figure 1) into our experiments. Hubo is a humanoid robot that has been enabled to perform several music tasks, including moving its arms on a beat, using motion-capture data to dance with a troupe, and actuating pitched pipes by striking them with its arms [7], [9]. All of these gestures, however, generate large amounts of motor noise. It would be useful for Hubo to be able to learn high-level musical features, such as beats, from audio contaminated with ego noise.

Hubo was initially not equipped with any auditory sensors. We mounted two lapel microphones on a head that we printed in a 3D-printer in order to allow the robot to hear. We also added a 2-channel preamplifier and audio interface called the USB Dual Pre to the system. As humans are often able to hear beats with two ears in noisy audio, we determined that no more than two channels would be needed for this task.

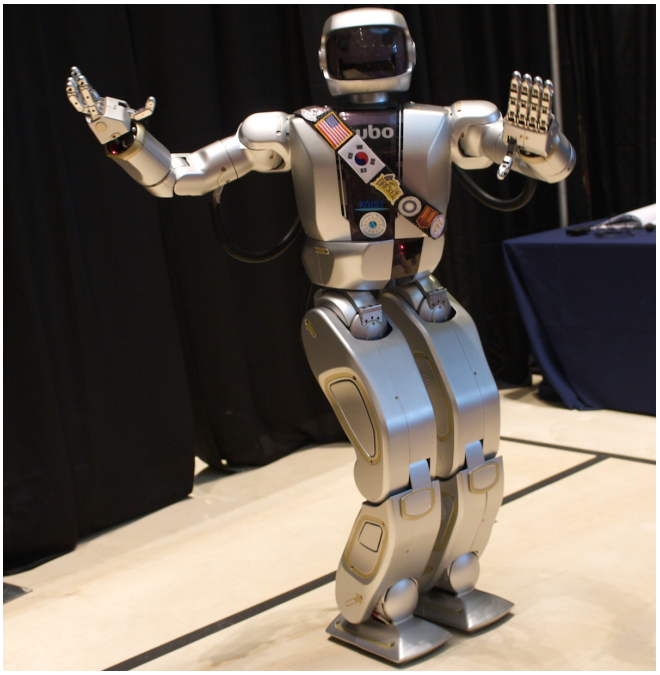

Fig. 1. A Hubo robot.

IV. Method

Our method includes four significant steps. First, the audio is converted into a 'stacked', spectrogram with each column representing multiple temporal windows. Second, the stacked spectrograms are used to determine the time-varying frequency components that make up the audio, as well as the activation probabilities of those components. Third, the system selects the component that most likely includes the beat. Lastly, the system estimates beat locations.

A. Calculating the stacked spectrogram

Audio is sampled by Hubo's microphones at 44.1 kHz and is averaged over both channels to form a monaural signal. An initial magnitude spectrogram is then calculated using a 256-point Fourier Transform with 50% overlap. We determined empirically that this resolution is high enough for useful identification of beat times and spectral characteristics, while also low enough to make the problem computationally tractable. Because the final system may involve the robot doing many other things while it listens to the audio, we wish to minimize the computation required for this task.

The spectrogram's elements are then averaged over time to produce a reduced spectrogram in which each column represents 46.3ms of audio and is spaced 23.2ms from the preceding column. The first three of these columns are then vectorized, as are columns 2-4, and so on, and the vectors are aggregated into a 'stacked' spectrogram (Figure 2). This structure allows for frequency characteristics that last for more than 46.3ms to be represented in a single column. The

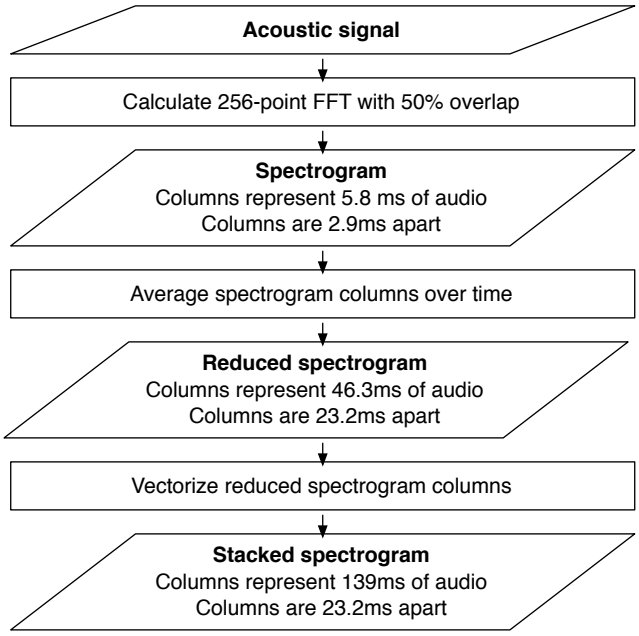

Fig. 2. Flowchart of the stacked spectrogram calculation.

system can thus learn time-varying spectral characteristics without having to resort to computationally expensive 2-dimensional or convolutional methods.

B. Identifying the latent components with PLCA

We next decompose the stacked spectrogram into its component elements. There are many methods for decomposing a matrix, including Principal Components Analysis (PCA) and Independent Component Analysis (ICA), but both of these can produce components with negative values, which are not meaningful for a spectrogram. Probabilistic Latent Components Analysis (PLCA) and Non-Negative Matrix Factorization (NMF), however, decompose matrices into non-negative values (and are numerically equivalent when the latter minimizes the Kullbach-Leibler divergence), with PLCA additionally providing a probabilistic framework that lets us model the stacked spectrogram as a histogram drawn from a set of latent components [10], [11]. This model allows the use of an efficient Expectation-Maximization algorithm to determine those components [10]. We therefore implement a version of PLCA to decompose the stacked spectrograms.

Given a stacked spectrogram S that has T total time indices, F frequency indices, and is composed of Z components, the system first calculates:

$$P_t(z|f) = \frac{P_t(z)P(f|z)}{\sum_{z'=1}^{Z} P_t(z')P(f|z')} \tag{1}$$

This is the a-posteriori probabilities of the components z at time t given observed frequencies f. After Equation 1 is calculated, the Maximization step is performed to update both the activation probabilities $P_t(z)$ and the components themselves $P(f|z)$:

$$P_t(z) = \frac{\sum_{f=1}^{F} P_t(z|f)S_t(f)}{\sum_{z'=1}^{Z}\sum_{f=1}^{F} P_t(z'|f)S_t(f)} \tag{2}$$

$$P(f|z) = \frac{\sum_{t=1}^{T} P_t(z|f)S_t(f)}{\sum_{f'=1}^{F}\sum_{t=1}^{T} P_t(z|f')S_t(f)} \tag{3}$$

The Expectation and Maximization steps alternate until convergence or for a certain number of iterations; in practice, we found that 40 iterations was sufficient. The system then records both the activation probabilities $P_t(z)$ and the components $P(f|z)$ for the provided musical signal. If needed, components can then be unstacked to show the spectral characteristics of those components over time.

An example is shown in Figure 3. Clean and noisy audio spectrograms of two audio excerpts are displayed, as are the activation probabilities and the spectral characteristics of a component that contains beat information for each excerpt. Peaks in the activation probabilities are aligned with the beat structure of the audio, even though that structure is obscured in the noisy audio. Additionally, the differences between the spectral characteristics of the beats in each excerpt are visible. For example, the spectrograms indicate that the beat spectral characteristics for the first example stop short of 20 kHz, while those of the second example extend further up the spectrum. This is reflected in the spectral characteristics plots; 'King' has a very sharp cutoff in its frequency spectrum at about 17 kHz, while 'Canned' has a much smoother rolloff after that point, indicating that it has some energy in higher frequencies.

C. Choosing the correct component

The system must next select the component that most likely contains the beat. In order to be flexible, it should not make assumptions on what beat spectral characteristics look like. Depending on the instrument used to perform the beats, the genre of music, and many other factors, the spectral characteristics of a beat component can vary dramatically. As such, instead of looking at the frequency components themselves to determine which component is likely to contain the beat, the system examines the activations.

If the beat is relatively constant over the audio, the activations of the correct beat component will likely spike at regular intervals corresponding to beats and will be relatively small elsewhere. In other words, they are likely to be somewhat periodic, and more periodic components, are more likely to contain beat information. By estimating the periodicity of each activation signal the system can therefore determine which components are likely to contain the beat.

This system estimates periodicity using an autocorrelation-based algorithm. It first calculates the autocorrelation of each component's activations, and then looks for the global maximum value in a range that corresponds to tempos of between 40 and 160 beats per minute (BPM). The system is biased in favor of tempos between 80 and 160 BPM, as most pop music falls within that range [12]. Global maximums between 40 and 80 BPM are only used if the autocorrelation

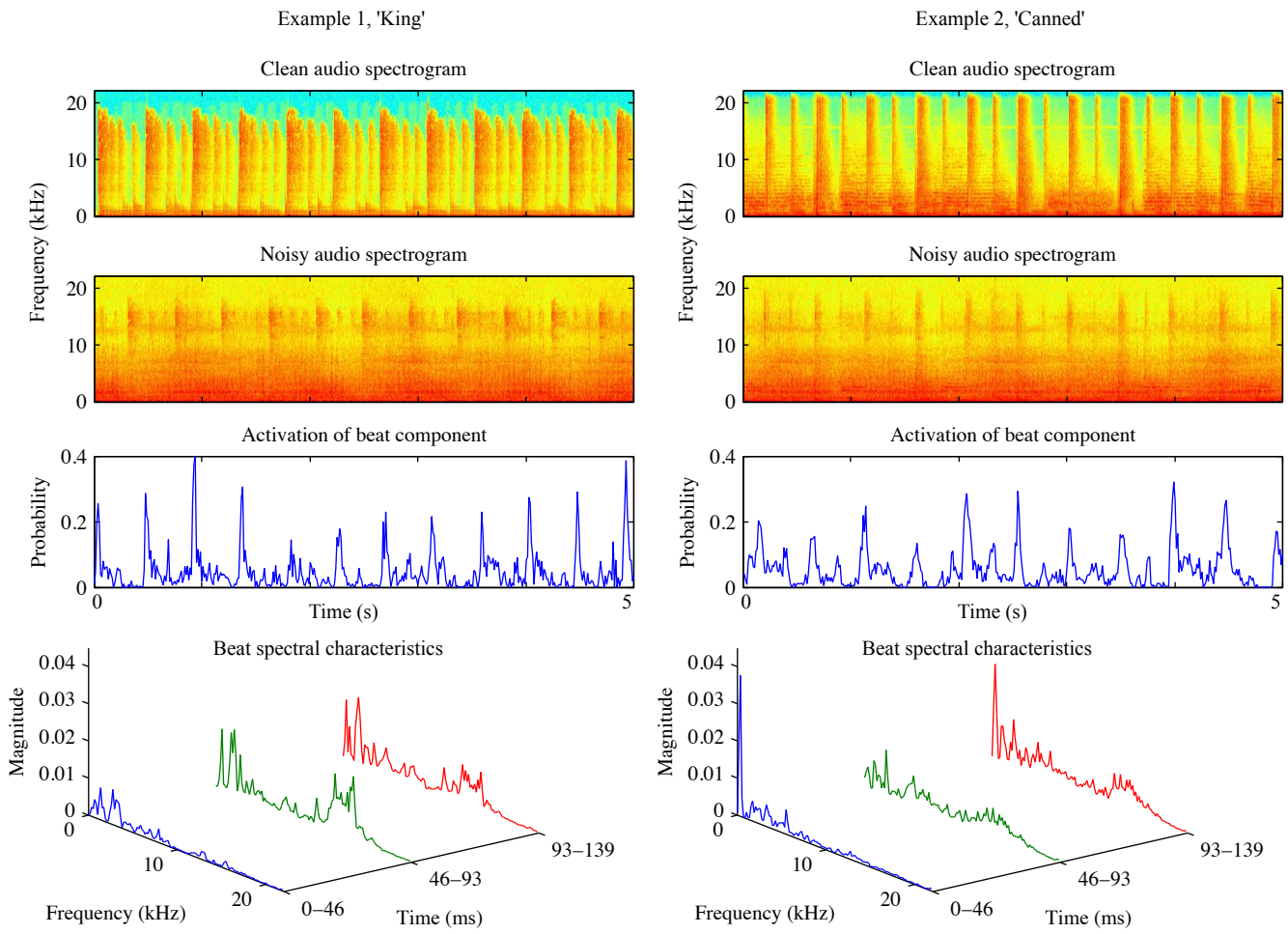

Fig. 3. Spectrograms of excerpts of clean (top) and noisy (second from top) audio, along with the activation probabilities of their beat component (second from bottom) and the spectral characteristics of that beat component (bottom). Example 1 is an excerpt from 'King of the Fairies/Western Junk' by Blood or Whiskey, denoted 'King', and Example 2 is an excerpt from 'Canned Heat' by Jamiroquai, denoted 'Canned'.

also has a local maximum at twice the tempo of the global maximum; this bias was found to rule out spurious slow periodicities corresponding to noise. The chosen maximum value is then divided by the autocorrelated signal's peak, which is a representation of the total energy in the signal. The component whose activation signal has the maximum ratio is thus marked as the beat component.

D. Marking beat positions

Once the activation signal for the true beat component is known, dynamic programming is used to mark beats. The system first estimates the music's period by finding the lag of the local maximum of the activation signals's autocorrelation function. This value represents the period at which the beat component activation repeats itself most strongly, and is thus an estimate of the signal's period. Periods corresponding to tempos of less than 80 BPM are halved to double the tempo estimate, as most pop music is faster than 80 BPM [12].

Once the period of the music is estimated, the system next selects a moment in time (such as the very beginning or ending of the piece) and records the value of the activation probability for the beat component at that time. It then shifts by one period and looks for a local maximum within three frames of the shifted position, recording both the value of the activation at that maximum as well as the position, and this repeats until the entire piece has been processed. A window of three frames around the period is used to account for minor tempo fluctuations in the musical performance. This process is then repeated for all possible starting positions in a given range, such as points within one period from the end of the piece. The list of beat locations corresponding to the activation values with the highest sum is determined to be the most likely list of beat locations for the musical piece.

V. Experiments and Results

A set of 18 songs, drawn from the pop music genre and with tempi ranging from 80 to 160 beats per minute, was collected, and the beat positions in each song were annotated by the lead author. All 18 songs have steady, heavy beats and were previously found to be easy for conventional beat trackers to analyze (with our previous system obtaining an average F-Measure score of .98), ensuring that poor

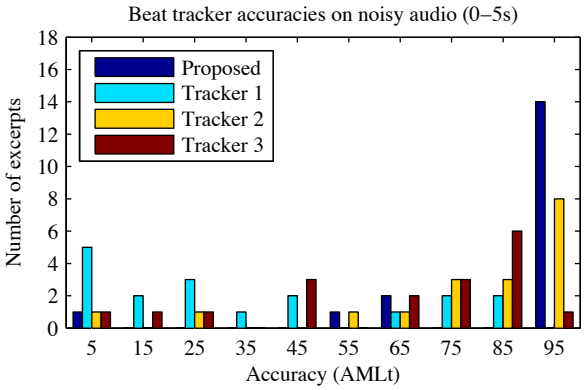

Fig. 4. Histogram of beat tracking accuracies, in scaled AMLt, on audio excerpts contaminated with robot noise (0-5s).

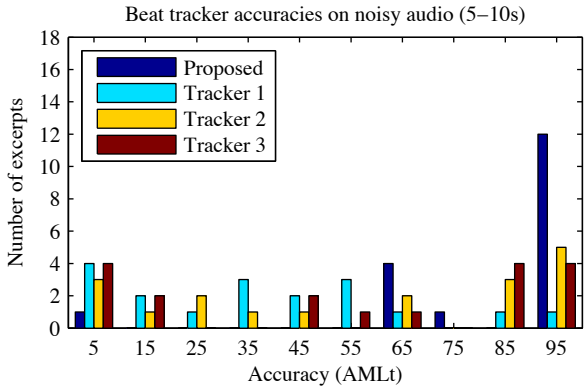

Fig. 5. Histogram of beat tracking accuracies, in scaled AMLt, on audio excerpts contaminated with robot noise (5-10s).

performance is due to the noise and not the music itself [7]. The songs were then played through a speaker positioned 9 feet from the Hubo robot, which recorded the audio using its lapel microphones. At the same time, the Hubo moved its arms up and down, with the shoulder motor moving from 0 to 1.35 rads and back again every 1.9 seconds. This motion had a sound pressure level ranging from 2-6 dB, with the exact value at a given point in time depending on the velocity of the robot's motors at that time.

Excerpts of each audio clip were extracted for processing. In order to satisfy the 'short-time' constraint we imposed, we took two consecutive five-second clips from each song. The full system was initially run on the first excerpt from each song, in order to learn both beat locations and the acoustic components of the music. The components were then held constant and the rest of the system was run on the second excerpt from each song. This helped determine if the components learned in one section of a song were universal enough to be useful on new audio from the same song.

The excerpts were then processed by the system, and beat locations and spectral characteristics were extracted. For comparison, we also ran three off-the-shelf beat trackers on the same audio. The most directly comparable of these is the beat tracker designed by Ellis (labeled *Tracker 1*), as it also uses a dynamic programming approach [13]. The other two beat trackers are Dixon's program 'Beatroot', labeled *Tracker 2*, and Oliveira et al's program 'IBT', labeled *Tracker 3* [14], [15]. In a recent analysis of 16 beat trackers by Holzapfel et al, these three beat trackers were found to be accurate and useful enough to merit inclusion in a multiple-tracker system that was restricted to using only five trackers [16]. Accuracy was evaluated using the standard beat tracking AMLt metric, in which a beat estimate is marked as correct if the distance between itself and the nearest beat is less than 17.5% of the inter-beat interval, and permits estimates at double- or half-the ground truth tempo level [17].

The mean result of running our proposed trackers and the three comparison trackers are shown in Table I, and histograms of these results are shown in Figures 4 and 5.

TABLE I

AVERAGE BEAT TRACKING ACCURACY, IN SCALED AMLT, ON AUDIO EXCERPTS CONTAMINATED WITH ROBOT NOISE .

Excerpt	Proposed	Tracker 1	Tracker 2	Tracker 3
0-5s	88.0	35.7	79.8	65.5
5-10s	85.6	38.4	58.0	54.0

As Table I and Figures 4 and 5 show, the proposed system surpasses the other systems on this noisy audio. When compared to the other dynamic programming algorithm, Tracker 1, the proposed system's superiority is clear, as it consistently outperforms the off-the-shelf algorithm. Instead of optimizing over activation probabilities, Tracker 1 uses the audio's subband spectral energy to determine beat salience, but this feature can easily develop spurious peaks due to ego noise, and so the tracker's ability is reduced. The proposed system also outperforms the other two trackers.

Of note is that three of the trackers have a drop in accuracy between the first and second excerpts, though the drop of the proposed system is less than 3 points. This is likely attributable to two main factors. First, because these excerpts are later in the piece than the first ones, additional instruments and effects can obscure the beat that are not present earlier in the music. Second, because the segments are short, the effects of spurious peaks in beat salience functions can have a disproportionate effect. One or two peaks caused by ego noise or by those other instruments can throw off the trackers, and the systems may have trouble recovering in the short time span. The proposed system, however, reduces the chances of this by extracting the beat component from the rest of the music and noise. The use of dynamic programming also allows the system to learn beats in simpler sections of the audio and apply that knowledge to more complex sections (and Tracker 1, which improved slightly between excerpts, also used dynamic programming).

While the beat spectral characteristics cannot be evaluated quantitatively, since without multitrack audio we cannot know exactly what the 'pure' beat components should be, the strong beat tracking results above imply that they are accurate. If the spectral characteristics did *not* correspond

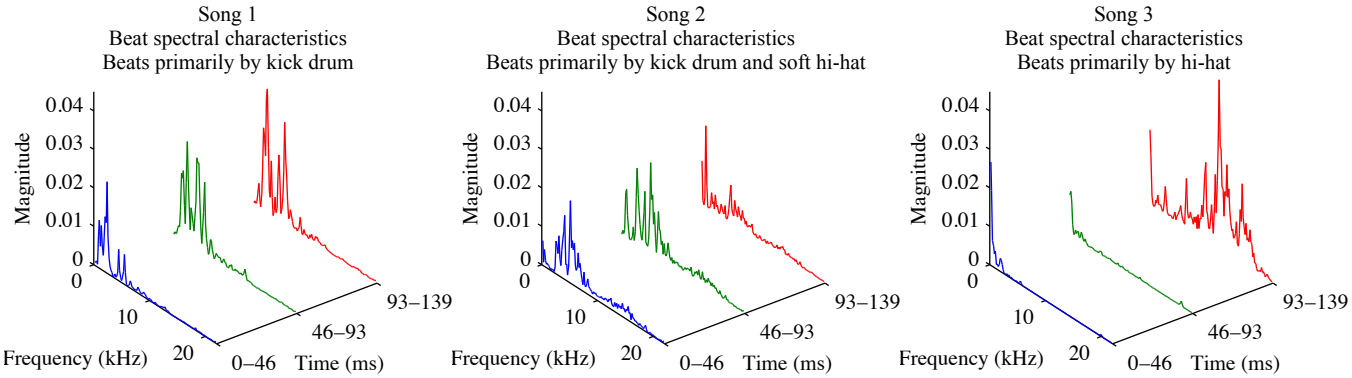

Fig. 6. Beat spectral characteristics. Song 1 is 'Moskau', by Deschinghis Khan, Song 2 is 'King and Queen of America', by Eurythmics, and Song 3 is 'I Fought the Law', by Bobby Fuller Four.

to the beats, it is improbable that the system, trying to maximize the activation of the estimated beat component would find them. Additionally, the spectral characteristics were often visibly different based on the type of drum used to produce the beat. Three examples are shown in Figure 6. 'Song 1' uses primarily kick drums, which have most of their energy at low frequencies, for its beats, and the corresponding spectrums indeed have most of their energy at low frequencies. The second song uses a kick drum as well as soft hi-hats, which spread their energy up to very high frequencies, for its beats. The spectral characteristics for this song thus show large values in lower frequencies and small raises at about 13 kHz. Finally, the third song is very hi-hat heavy, and its strongest frequencies are in the middle-to-high range of the spectrum, especially after 93 ms.

VI. Conclusion and Future Work

We have developed a system that can learn beats from noisy audio given only five seconds of music. In the future, we aim to expand this algorithm for greater use in both robotic musical performances and more general topics in robotic audition. Relating to music, we aim to develop an updating procedure that allows for the system to listen to longer excerpts of music and update its knowledge of the beats without needing to completely retrain on each new segment of audio. This will allow the robot to react appropriately to the music, even if the tempo or rhythm changes, in a computationally efficient manner. We also aim to exploit the knowledge of beats to allow the robots to determine higher-level structure, such as by classifying beats into categories such as off-beats, on-beats, and downbeats. This information could help the robots produce more sophisticated responses.

The ability to learn elements such as beats in musical audio could also be extended to other domains within robot audition. Self-driving cars, for instance, could learn what cars around them sound like, even in the presence of acoustic noise (such as a rainstorm). Being able to detect other cars could be useful in collision avoidance. Another useful domain for this system is with service robots, which could learn various commands in noisy environments and could therefore be more useful in real-world situations. In general, the ability of a robot to learn acoustic components quickly even in noise could be useful for a wide variety of tasks.

References

[1] K. Yoshii *et al.*, "A biped robot that keeps steps in time with musical beats while listening to music with its own ears," in *Proc. of the International Conference on Intelligent Robots and Systems*, 2007.

[2] K. Murata *et al.*, "A beat-tracking robot for human-robot interaction and evaluation," in *Proc. of the International Conference on Humanoid Robotics*, 2008.

[3] G. Ince *et al.*, "Robust ego noise suppression of a robot," *Lecture Notes in Computer Science*, vol. 6096, pp. 62–71, 2010.

[4] I.-W. Park *et al.*, "Mechanical design of humanoid robot platform khr-3 mechanical design of humanoid robot platform khr-3 (kaist humanoid robot - 3: Hubo)," in *Proc. of the International Conference on Humanoid Robots*, pp. 321–326, 2005.

[5] H. Kozima, M. P. Michalowski, and C. Nakagawa, "Keepon," *International Journal of Social Robotics*, vol. 1, pp. 3–18, January 2009.

[6] G. Weinberg, A. Raman, and T. Mallikarjuna, "Interactive jamming with shimon: A social robotic musician," in *Proc. of the 4th International Conference on Human Robot Interaction*, 2009.

[7] D. K. Grunberg *et al.*, "Robot audition and beat identification in noisy environments," in *Proc. of the International Conference on Intelligent Robots and Systems*, pp. 2916–2921, 2011.

[8] J. L. Oliveira *et al.*, "Live assessment of beat tracking for robot audition," in *Proc. of the International Conference on Intelligent Robots and Systems*, pp. 992–997, 2012.

[9] A. M. Batula *et al.*, "Using audio and haptic feedback to detect errors in humanoid musical performances," in *Proc. of the International Conference on New Interfaces and Musical Expression*, 2013.

[10] P. Smaragdis, B. Raj, and M. Shashanka, "Missing data imputation for spectral audio signals," in *Proc. of the IEEE International Workshop on Machine Learning for Signal Processing*, 2009.

[11] M. Shashanka, B. Raj, and P. Smaragdis, "Probabilistic latent variable models as non-negative factorizations," *Computational Intelligent and Neuroscience Journal*, 2008.

[12] D. Moelants, "Dance music, movement and tempo preferences," in *Proc. of the 5th Triennal ESCOM Conference*, 2003.

[13] D. P. W. Ellis, "Beat tracking by dynamic programming," *Journal of New Music Research*, vol. 36, no. 1, pp. 51–60, 2007.

[14] S. Dixon, "Evaluation of the audio beat tracking system beatroot," *Journal of New Music Research*, vol. 36, pp. 39–50, March 2007.

[15] J. L. Oliveira *et al.*, "IBT: A real-time tempo and beat tracking system," in *Proc. of the 2010 International Society on Music Information Retrieval*, pp. 291–296, 2010.

[16] A. Holzapfel *et al.*, "Selective sampling for beat tracking evaluation," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 20, pp. 2539–2548, November 2012.

[17] M. Davies, N. Degara, and M. Plumbley, "Evaluation methods for musical audio beat tracking algorithms," Tech. Rep. C4DM-TR-09-06, Queen Mary University of London: Center for Digital Music, 2009.