Real-time Drums Transcription with Characteristic Bandpass Filtering

Maximos A. Kaliakatsos–Papakostas
Computational Intelligence Laboratoty (CILab), Department of Mathematics, University of Patras, GR–26110 Patras, Greece
maxk@math.upatras.gr

Andreas Floros
Department of Audio and Visual Arts, Ionian University, GR–49100 Corfu, Greece
floros@ionio.gr

Nikolaos Kanellopoulos
Department of Audio and Visual Arts, Ionian University, GR–49100 Corfu, Greece
kane@ionio.gr

Michael N. Vrahatis
Computational Intelligence Laboratoty (CILab), Department of Mathematics, University of Patras, GR–26110 Patras, Greece
vrahatis@math.upatras.gr

ABSTRACT

Real–time transcription of drum signals is an emerging area of research. Several applications for music education and commercial use can utilize such algorithms and allow for an easy-to-use way to interpret drum signals in real–time. The paper at hand proposes a system that performs real–time drums transcription. The proposed system consists of two subsystems, the real–time separation and the training module. The real–time separation module is based on the use of characteristic filters, combining simple bandpass filtering and amplification, a fact that diminishes computational cost and potentially renders it suitable for implementation on hardware. The training module employs Differential Evolution to create generations of characteristic filter combinations that optimally separate a set of given drum sources. Initial experimental results indicate that the proposed system is relatively accurate rendering it convenient for real-time hardware implementations targeted to a wide range of applications.

Categories and Subject Descriptors

J.7 [**Computer Applications**]: Computers in other Systems—*Real time*; I.2.8 [**Computing Methodologies**]: Artificial IntelligenceProblem Solving, Control Methods and Search[Heuristic Methods]

AM '12, September 26 - 28 2012, Corfu, Greece

General Terms

Algorithms,Experimentation

Keywords

automatic drums transcription, characteristic filter, differential evolution application

1. INTRODUCTION

Real–time audio analysis is becoming a subject of great scientific interest. The increasing computational power that is available by small electronic and portable devices allows the encapsulation of sophisticated algorithms to commercial and educational applications. The paper at hand introduces a novel approach for performing real–time transcription of a polyphonic single–channel drum signals. The novelty of the proposed approach is the simplicity of its architecture, while high-efficiency is achieved based on a robust training procedure. The transcription strategy proposed was implemented in terms of two submodules: the real–time separation and the training module. The first one utilizes a combination of bandpass filters and amplifiers that we hereby term as characteristic filters. These filters are trained to capture the characteristic frequencies produced by the onset of each percussive element of a specific drum set. Thus, the intensity of the signal that passes through each characteristic filter indicates the onsets of the respective percussive element. The training process is realized through a) the evolution of the characteristic filters with the Differential Evolution (DE) algorithm and b) fitness evaluation measures for determining each filter's ability to correctly detect the onset of the respective drum element.

Although several works have been already presented for the transcription of recorded drum signals, until very recently, the real–time potential of this task remained unexplored. Among the non–real–time methodologies, the early works of Schloss [12] and Blimes [3] incorporated the transcription of audio signals with one percussive element be-

ing active at a time. The work of Goto and Muraoka [7] (extended in [14]) introduced the transcription of simultaneously played drum elements by utilizing template matching. Several other methodologies are based on preprocessing a recorded file for onset detection [8]. These methodologies utilize sophisticated pattern recognition techniques like Hidden Markov Models and Support Vector Machines [6], N-grams and Gaussian Mixture Models [10], Prior Subspace Analysis and Independent Component Analysis [5], Principal Component Analysis and Clustering [4] and Non-Negative Matrix Factorization [9] among others. The real–time perspective of drums transcription has been examined in [1], where each drum beat is identified with Probabilistic Spectral Clustering Based on the Itakura-Saito Divergence.

The rest of the paper is organized as follows. Section 2 presents the proposed transcription technique by describing the two modules that comprise its implementation: the real–time separation and the training modules. The first one is analyzed in Section 2.1. A detailed analysis of the training module is provided in Section 2.2, combined with the analytic description of the required parameter representation, the continuous transformation of the training process and the segregation of the waveforms to onset and no–onset parts. Experimental results on using 3 drum signals among 12 different drum sets are provided in Section 3, which indicate that the proposed approach is promising and suitable for real-time implementation on reduced-power hardware platforms. Finally, Section 4 concludes the work and defines some points for future work.

2. THE PROPOSED METHODOLOGY

The presented approach receives a single–channel signal of drums and provides real–time indications about the onset of each percussion element. In this way, it permits the real–time transcription of drums performances using a single microphone as an input device. The architecture of the system illustrated in Figure 1 is rather simple, avoiding the hazard of software–oriented latency dependencies deriving from complicated algorithms that demand high computational cost and advanced signal processing techniques. Additionally, the complete system can be easily implemented in hardware, provided that the training process is accomplished through a typical computer. As mentioned previously, the proposed technique implementation includes two modules, both of which are for the purposes of this work developed in software: the training and the real–time separation module. These modules are described in detail in the following two Sections.

2.1 The real–time separation module

We have built and evaluated our system in a set of test–tube cases (sampled and processed drum recordings), with the utilization of 3 drum elements, the kick (K), the snare (S) and the hi-hat (H). The module under discussion utilizes correspondingly 3 filter–amplifier pairs that are able to isolate characteristic frequency bands of the respective percussive elements. As Figure 1 demonstrates, the polyphonic single–channel signal that is captured by the microphone is processed by the filter–amplifier pairs, a procedure that we hereby term *characteristic filtering*, with each filter–amplifier pair being called a *characteristic filter*.

Each characteristic filter utilizes a *bandpass filter* with frequency response as the one depicted in Figure 2. The results

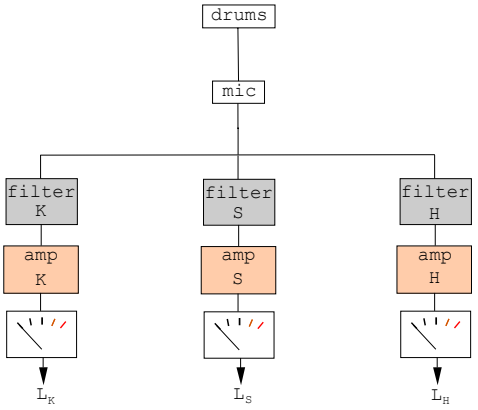


Figure 1: Block diagram of the proposed methodology. If the L_K, L_S and L_H levels exceed a predefined threshold, then the respective drum element is considered active.

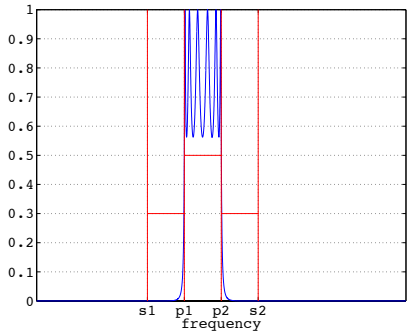


Figure 2: The frequency response of a bandpass filter and the parameters that define its characteristics.

presented in this work are implemented using the elliptic IIR filters of MATLAB. These filters are defined by the following four parameters:

1. $\texttt{s1}_I$: the edge of the stop band,
2. $\texttt{p1}_I$: the edge of the pass band,
3. $\texttt{s2}_I$: closing edge of the pass band and
4. $\texttt{p2}_I$: edge of the second stop band,

where the index $I \in \{\text{K, S, H}\}$ characterizes the filter values for the respective percussive elements. Furthermore, we denote by $\texttt{v}_I$, $I \in \{\text{K, S, H}\}$, the amount of amplification for each filtered signal. Given this formulation of the bandpass filters and the amplification values, the problem can be stated as follows: find the proper $\texttt{s1}_I$, $\texttt{p1}_I$, $\texttt{p2}_I$, $\texttt{s2}_I$ and $\texttt{v}_I$ values for $I \in \{\text{K, S, H}\}$ so that maximum *separability* between K, S and H is accomplished with the respective filters.

The term *separability* is used to convey that the respective characteristic filters suppress the frequency bands that result in cross–talk between the percussive elements and at the same time highlight the exclusive frequency band of each active drum part. With the terminology provided so far, two aspects need to be discussed for the construction of the training module: parameter tuning and separability formulation.

2.2 The training module

The training module adjusts the characteristic filter parameters (frequency borders and amplification levels) for each percussive element. The training is based on a single recorded sample by each element provided by the drummer, i.e. in our case a kick (K), a snare (S) and a hi-hat (H). These sample clips are used as the preset sound patterns for each element. They are fed into the system and are handled by the training module with a training methodology described in the following paragraphs.

2.2.1 Parameter Encoding and Filter Evolution

As mentioned in the previous Section, the bandpass filters are described by 4 values, $\mathtt{s1}_I$, $\mathtt{p1}_I$, $\mathtt{p2}_I$ and $\mathtt{s2}_I$ for $I \in \{\text{K, S, H}\}$, for which we obviously observe that $\mathtt{s1}_I < \mathtt{p1}_I < \mathtt{p2}_I < \mathtt{s2}_I$. To reduce the number of parameters and the consequent computational and algorithmic cost derived by the aforementioned inequality checks, we encode these 4 parameters using 3 variables: α_I, ρ_I and τ_I, for $I \in \{\text{K, S, H}\}$. This decoding is accomplished as follows:

$$\begin{aligned}\mathtt{s1}_I &= \alpha_I \\ \mathtt{p1}_I &= \alpha_I + \rho_I \\ \mathtt{p2}_I &= \alpha_I + \rho_I + \tau_I \\ \mathtt{s2}_I &= \alpha_I + 2\ \rho_I + \tau_I\end{aligned}$$

With this simplification we consider only symmetric bandpass filters, meaning that $\mathtt{s1}_I - \mathtt{p1}_I = \mathtt{p2}_I - \mathtt{s2}_I$. Thus, a characteristic filter is defined with four variables (α_I, ρ_I, τ_I, $\mathtt{v}_I$), with the latter variable indicating the amplification value.

Since we can make no prior assumptions about the properties of each characteristic filter, we utilize a *metaheuristic* search method to tune the 4-tuple of each filter. The search space for finding three optimal characteristic filters is thus a 12-dimensional space. The search method that we use is the Differential Evolution (DE) approach [13, 11]. DE is initialized with a set of random guesses about the optimal filters by producing an "initial population" of 12-dimensional vectors, also called individuals, that define the properties of the three filters. Then it iteratively provides optimized solutions to the problem at hand by improving the candidate solutions in each iteration, also called generation, using the "crossover" operator which combines the coordinates of individuals to produce new ones. With a selection procedure, the individuals that provide an improved solution to the problem propagate to the next generation. This improvement is measured with a *quality* or *fitness* function, the optimal points of which describe a satisfactory solution to the given problem. Using the aforementioned formulation, the DE algorithm searches for the appropriate 4-tuples that describe the 3 characteristic filters which designate the characteristic frequencies of each percussion element. To this end, the aptness of each characteristic filter combination needs to be evaluated.

2.2.2 The objective function

For the formulation of the proper *fitness* function, we previously have to define as strictly as possible the desired attributes of the system. To this end, the system should distinguish:

1. the onsets of separate percussive elements,
2. the onsets of simultaneously played elements in *all possible combinations* and
3. the parts of silence or no-onset regions.

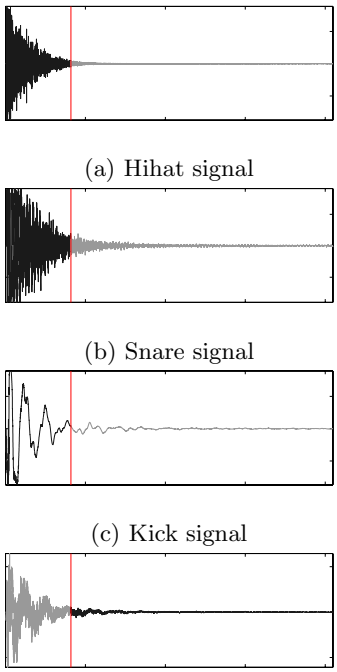
Table 1: All the possible onset scenarios that the system may encounter.

	onset combination		
scenario	K	S	H
1	1	0	0
2	0	1	0
3	0	0	1
4	1	0	1
5	1	1	0
6	0	1	1
7	1	1	1
8	0	0	0

Therefore, considering the fact that we have 3 percussive elements, we have 8 possible scenarios, as demonstrated in Table 1. Specifically, scenarios 1, 2 and 3 describe the single onset events, where a single drum element is played. Scenarios 4, 5 and 6 incorporate simultaneous activation of two elements, while scenario 7 describes the simultaneous sounding of all three considered elements. The utilization of the 8th scenario, the no-onset scenario, is an auxiliary condition that improves the accuracy of the system towards locating the "*head*" of the drum hit and discarding the "*tail*" (the "head" and "tail" terminology is borrowed by [1]), improving the detection accuracy of each percussive elements' onset.

Given the 8 scenarios, the training of the system can be realized with the utilization of a template sound clip for each drum element provided by the drummer. Having the separate sources of each percussive element we are able to construct any scenario by mixing–down the respective element waveforms. Specifically, since we are interested in capturing only the head of the waveform, we split each element's clip in two parts: the head and the tail. An example of this splitting is depicted in Figure 3. The scenarios that incorporate element activations (all scenarios except the last one), utilize only head part of the participating elements. The last scenario on the other hand, utilizes the tail of the mixed–down signal of all 3 template clips.

The training module creates all the combinations dictated by the above scenarios. Next, we describe the training process with an example on a specific scenario. Later, we will provide an analysis on the no-onset training scenario. Suppose that we are currently constructing and testing the 4th scenario, with binary representation $\{1, 0, 1\}$ which indicates that only the K and H elements are active. We mix–down the head parts of the K and H template clips, provided in the beginning of the training process by the drummer, and pass the mixed–down signal through all three characteristic filters. We then measure the amplitude responses or the *activity* of these filters. If a characteristic filter's activity exceeds a predefined *threshold*, then the respective percussive element is considered *active*, else it is considered *inactive*. When a filter is active, we conclude that the respective percussive element is played. The training for the 4th scenario would have a successful conclusion if the characteristic filters of K and H were active (their levels are above threshold) and the S characteristic filter inactive (its level is below thresh-



(a) Hihat signal

(b) Snare signal

(c) Kick signal

(d) The summed signals

Figure 3: Darker parts demonstrate the waveform parts that are used for the training scenarios. The lighter parts are discarded.

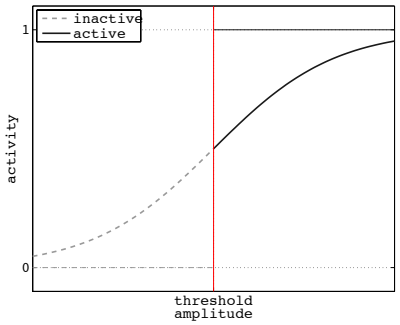


Figure 4: The sigmoid function that was utilized for the continuous transformation of the discrete objective function.

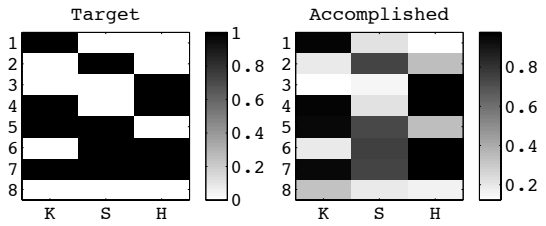


Figure 5: The binary target scenarios (left) and the continuous filter amplitude responses of a training trial with error 0.7010**.**

old).

However, there are two problems with this *binary* training approach. Firstly, it afflicts the training itself, since the search space abounds in large plateaus of local minima that provide unsatisfactory solutions. Secondly, even if an area with a satisfactory local minimizer is located, the solution it provides would most likely be a solution on the boundary of acceptable. Thus the system would be very sensitive to noise, i.e. small modification of the input signal (like dynamic variations of a drum hit) would provide misleading results during real–time separation.

Both drawbacks are avoided if we consider a continuous analogous of the aforementioned binary training scheme. The continuous scheme rewards the filter activities that converge to the correct binary solution and at the same time penalizes opposite answers in a *continuous* manner. Consider a characteristic filter activity response, r, and a threshold, t, above which this response is considered as active. The continuous analogous of the thresholding states is provided by normalizing the response according to its distance from the threshold within $[0, 1]$, by

$$c = \frac{1}{2} - \frac{1}{2}\arctan(\lambda(t - r)), \qquad (1)$$

where λ is a smoothing coefficient that controls the convergence rate to the binary states. The result of the transformation of Equation 1 is illustrated in Figure 4.

The continuous approach of training tackles the two aforementioned problems caused by the binary approach. Firstly, the flat fitness plateaus in the 12–dimensional search space become curved. This facilitates the training process by offering continuous optimization flow. From Figure 4 it is obvious that the farthest an activity response moves from the threshold value, the more it approaches the desired activation value (0 or 1). This resolves the second problem, since the borderline solutions (solutions close to the threshold) do not have high fitness rate. On the contrary, activities with considerably higher value than the threshold are closer to one and activities with lower value to zero. Thus, extreme activity differences are rewarded, leading to more robust solutions. Figure 5 illustrates the binary target values (left) and the normalized responses (right) of a trained system. The training error is measured as the Euclidean distance of the two matrices (square root of the squared differences of the respective matrix elements), which is the *fitness evaluation* of the 3 characteristic filters combination among all scenarios.

An important aspect of the training procedure is the scenario enumerated as number 8, the no-onset scenario. If we train the system without the no-onset scenario, then the optimal filters that are obtained by the training process do not detect the onset efficiently. Specifically, on the one hand they capture frequency regions that are characteristic for each drum element, but on the other hand these regions are not characteristic about their *onset*. For example, the characteristic filter of the snare or the kick drum captured their harmonic frequencies and thus remained active several milliseconds after their onset, as did the respective harmonic frequencies. The no-onset scenario excludes the filters that preserve the "harmonic tails", keeping only the ones that are characteristic about the head–onset part. The clip that is utilized for the no-onset scenario is the tail part of the mixed–down audio of all preset clips, as illustrated in Figure 3 (d). The mixed–down audio is filtered *before* the tail part is cut off, in order to maintain the remnants of the

filtered impulsive part.

3. EXPERIMENTAL RESULTS

To assess the accuracy of the presented system we measure the responses among 12 different drum sets in 2 rhythmic sequences. In order to have an accurate representation of the ground truth rhythms, they are recorded through MIDI files. These MIDI files trigger sampled percussion elements that correspond to a kick, a snare and a hi-hat, combined to form 12 different drum sets. Both rhythmic sequences through which we tested the system were recorded in a tempo of 100 beats per minute. They are 1 measure long, but they differ in their dynamics. `Rhythm1` has no dynamic variations, while `Rhythm2` has great dynamic variations expressed with MIDI velocity and more onsets. The MIDI velocity variations do not only affect the intensity level of the each drum hit, but also alter the sound characteristics. This is accomplished by activating separate drum samples of the same element with different drum hit dynamics. Furthermore, we assess the accuracy of each percussive element separately, in order to obtain indications about the limitations and improvement potential of the system. Therefore, we could say that we measure the system's ability to locate *onsets* of separate drum elements.

The experimental setup is focused on assessing the accuracy on onset detections, given a time error tolerance. Specifically, we measure the *precision*, the *recall* and their combination into the *f–measure*, for onset detections of separate drum elements that fall into certain time windows. Precision describes the percentage of the correctly detected onsets among all the identified onsets. Recall describes the correctly detected onsets, among the annotated "ground truth" onsets. Strictly speaking, if L is the set of onsets that are correctly allocated by the system and C is the set of the annotated onsets, then precision is computed by $p = \frac{|L \cap C|}{|L|}$ and recall by $r = \frac{|L \cap C|}{|C|}$, where $|X|$ denotes the number of elements in a set X. High values of precision informs us that the detected onsets are mostly correct, but we cannot not be sure about how many onsets remain to be detected. This lack of detecting enough onsets is monitored with recall. Thereby, a good result is described by combined high values of both precision and recall. This combination is provided by the f–measure [2] and is computed as $f\text{–measure} = 2pr/(p + r)$.

A drum element onset is considered correct if it is detected within a specified time interval. Following this kind of analysis, we admit that a percussive element of the ground truth rhythmic sequence may not have two onsets into the same time interval window. Moreover, our system in the present form is not capable of defining the intensity of an onset, although this could be realized with certain modifications (which is discussed in Section 4). The above comments indicate that there is no need to include an experimental procedure with numerous ground truth rhythmic sequences. On the other hand, it is important to assess the system's accuracy in several time windows of error tolerance, on two rhythms with different intensity characteristics. Thus, we are able to interpret latency issues imposed by the algorithm per se and the system's sensitivity in a variety of playing styles in terms of dynamics. The latency of the proposed system is not "software–oriented", in a sense that it is not caused by increased computational cost of the algorithmic parts. The latency has to do with the areas of the drum signals that the bandpass filters are able to isolate. Specifically, each filter would work with no latency if it could isolate the signal of a drum element at the exact time of its onset. However, there is great spectral overlapping between different percussive onset impulses, a fact that forces the filters to adapt and isolate the "tail" parts, several milliseconds after the actual onset occurs.

The training module was allowed to evolve 50 individuals of filter combinations as described in Section 2.2.1 for 100 generations for each drum set's preset clips. The characteristic filter values of the initial population had bandpass frequency borders within the audible range, and the amplification values were allowed to have a range between 0 and 100. Table 2 demonstrates the error and the characteristic filter values of the best individual for each drum set. Since the characteristic filters are symmetric, as stated in Section 2.2.1, they are described in Table 2 with their center frequency $f_c = (\mathtt{s1} + \mathtt{s2})/2$, their range $Q = (q_1 + q_2)/2$, where $q_1 = (\mathtt{s1} + \mathtt{p1})/2$ and $q_2 = (\mathtt{s2} + \mathtt{p2})/2$, and their amplification value `v`. These values are also depicted with box plots in Figure 6, where it is clear that the optimal characteristic filter values are grouped in distinguishable distributions per drum element.

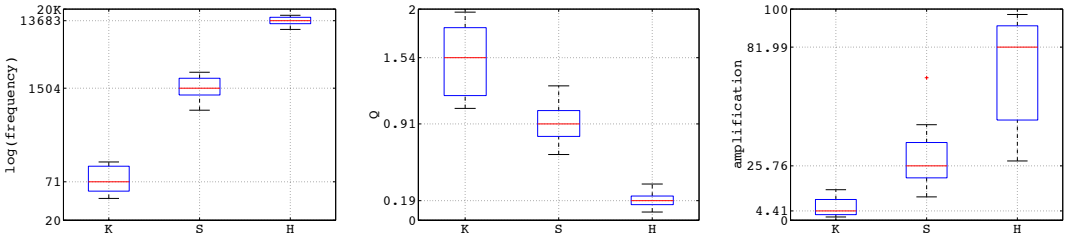
The training module created the characteristic filter combinations for each drum set. Using these filter combinations, we have applied the real–time separation framework on the two rhythms recorded by the respective drum sets. Figure 7 illustrates the spectrograms of `Rhythm1` played by a certain drum set and the signal that was produced by the characteristic filters of this drum set. It is clear that the filtered signals isolate characteristic frequencies of the respective element's onset. `Rhythm1` is also depicted in binary form in Figure 7 (a), while in Figure 7 (b) and (c) we see the activity level of each filter and the resulting binary rhythm respectively.

The mean precision, recall and f–measure values among all drum sets, for both rhythms for each percussive element are demonstrated in Table 3. In a 30ms time window the results are not satisfactory, but for a 50ms tolerance window they are improved impressively. For both rhythmic sequences the precision reaches perfection, but the recall for `Rhythm2`, remains between 0.8 and 0.9. Perfect precision means that the detected onsets are actually correctly detected. Lower recall means that a percentage of the onsets remains undetected. The hi-hat element accomplishes maximum accuracy in a smaller time window, compared to the rest. The kick drum comes second in terms of detectability accuracy, while the snare seems the hardest to locate within a window smaller than 100ms. However, a window size of 50ms to 70ms provides satisfactory results.

To examine the contribution of each drum set to the results discussed so far we present the f–measure among all the percussive elements in each drum set. These results are demonstrated in Table 4 for two error tolerance time windows, 30ms and 50ms. In the time window of 30ms, that produces the worst results, the accuracy depends on the drum set. The drum set number 6 for example achieves relatively high accuracy, on contrast to the drum set number 7. Additionally, the majority of the drum sets present an overall accuracy around 0.7. Another interesting, although expected, result is the relation of the accuracy among different drum sets with the error values during training by the

Table 2: The error and the characteristic filter values for the best individual of each drum set.

		K			S			H		
	errors	f_c	Q	v	f_c	Q	v	f_c	Q	v
1	0.49	61.26	1.68	2.69	897.08	1.27	11.07	14854.97	0.16	76.06
2	0.59	50.89	1.93	2.60	1277.61	1.08	33.30	14514.17	0.23	88.26
3	0.42	58.23	1.74	3.41	1452.29	0.78	18.21	15887.40	0.09	95.93
4	0.52	80.77	1.29	6.48	1476.09	1.04	21.94	16338.80	0.17	52.07
5	0.55	134.63	1.22	10.00	1729.53	0.93	67.50	12354.97	0.23	28.08
6	0.33	120.63	1.39	14.47	2534.41	0.95	13.12	10293.96	0.23	85.80
7	0.72	109.53	1.06	9.72	1531.13	0.73	40.34	10399.69	0.34	97.45
8	0.60	48.12	1.79	2.61	2353.63	0.81	23.67	13364.87	0.17	81.65
9	0.44	52.77	1.86	3.34	1842.81	0.89	26.82	12470.65	0.08	41.18
10	0.64	114.40	1.08	5.41	1134.96	0.62	45.24	13141.60	0.20	82.34
11	0.52	40.85	1.97	1.56	2408.93	0.85	28.79	14003.08	0.22	42.87
12	0.59	120.09	1.14	9.96	732.52	1.04	24.69	15736.22	0.13	96.54

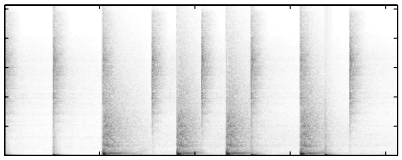


(a) center frequencies (f_c) (b) Q values (c) amplification

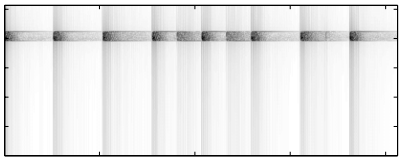
Figure 6: Box plots of the best characteristic filter values for the respective drum elements, as demonstrated in Table 2.

Table 3: The mean precision, recall and f–measure values for different error tolerance time windows, among all drum sets for the two rhythms, for each percussive element. Numbers in boldface typesetting indicate the smallest window that the maximum accuracy is accomplished.

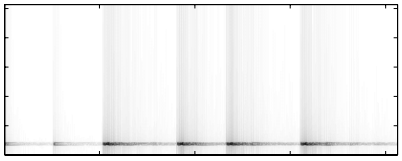
	Precision							
	Rhythm1				Rhythm2			
	30ms	50ms	70ms	100ms	30ms	50ms	70ms	100ms
H	0.9375	**1.0000**	1.0000	1.0000	0.9375	**1.0000**	1.0000	1.0000
S	0.6652	0.8671	0.9833	**1.0000**	0.6652	0.8671	0.9833	**1.0000**
K	0.3292	0.9375	**1.0000**	1.0000	0.3292	0.9375	**1.0000**	1.0000
	Recall							
	Rhythm1				Rhythm2			
	30ms	50ms	70ms	10ms	30ms	50ms	70ms	10ms
H	0.9479	**0.9792**	0.9792	0.9792	0.8426	**0.8704**	0.8704	0.8704
S	0.9375	**1.0000**	1.0000	1.0000	0.7500	**0.8000**	0.8000	0.8000
K	0.3542	0.9375	**1.0000**	1.0000	0.2833	0.7500	**0.8000**	0.8000
	F–measure							
	Rhythm1				Rhythm2			
	30ms	50ms	70ms	100ms	30ms	50ms	70ms	100ms
H	0.9378	**0.9889**	0.9889	0.9889	0.8828	**0.9301**	0.9301	0.9301
S	0.7610	0.9162	0.9907	**1.0000**	0.6890	0.8206	0.8815	**0.8889**
K	0.3380	0.9375	**1.0000**	1.0000	0.3016	0.8333	**0.8889**	0.8889



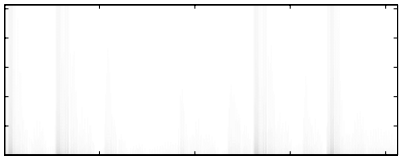
(a) The drum signal spectrogram



(b) The H characteristic filter spectrogram

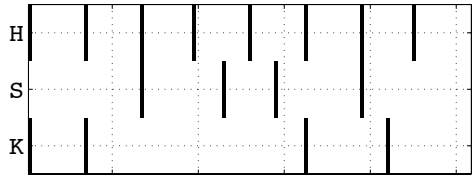


(c) The S characteristic filter spectrogram

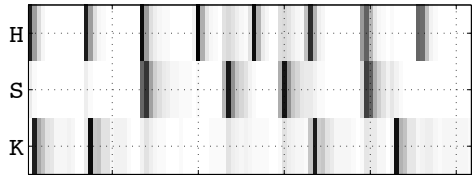


(d) The K characteristic filter spectrogram

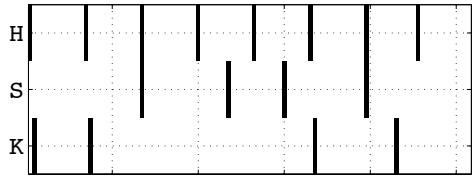
Figure 7: The spectrogram of the single–channel drum signal and the derived spectrograms after applying the characteristic filters.



(a) The ground-truth rhythm



(b) Amplitude activation of the respective filters



(c) The extracted rhythm

Figure 8: (a) The ground-truth rhythm. (b) The activity levels from each filter and (c) the extracted binary rhythm.

Table 4: Mean f–measure among all percussive elements for each rhythm, with error tolerance of 30 and 50ms. The final raw shows the correlation of the respective line with the training error demonstrated in Table 2.

	30ms		50ms	
Rhythm	`Rhythm1`	`Rhythm2`	`Rhythm1`	`Rhythm2`
1	0.7556	0.6905	0.9778	0.8843
2	0.7000	0.6449	0.9111	0.8304
3	0.7500	0.6841	1.0000	0.9063
4	0.7037	0.6471	1.0000	0.9063
5	0.7667	0.7043	1.0000	0.9063
6	**0.9167**	**0.8322**	1.0000	0.9063
7	**0.4343**	**0.4080**	0.8258	0.7582
8	0.7424	0.6841	1.0000	0.9063
9	0.7083	0.6449	1.0000	0.9063
10	0.5333	0.4955	0.9167	0.8322
11	0.5556	0.5189	0.8500	0.7784
12	0.5801	0.5391	0.8889	0.8151
error corr.	-0.7957	-0.7871	-0.6457	-0.6481

respective drum set. The linear correlation of the training errors in Table 2 with the drum set accuracy assessment in Table 4 is strong negative, which means that the smaller the error during training, the higher the accomplished precision during real–time separation.

4. CONCLUSIONS AND FUTURE ENHANCEMENTS

This paper presents a novel method for real–time drums transcription, through a single–channel polyphonic drums signal, based on a combination of bandpass filtering and amplification. These filter–amplifier pairs are called characteristic filters of each percussive element. Each characteristic filter allows a signal of considerable energy to pass if the respective drum element is played. The simplicity of the system's architecture allows efficient real–time transcription with minimal cost in terms of computational power. The system is trained with the Differential Evolution (DE) algorithm, which optimizes the filtering and amplitude parameters based on the percussive elements provided as preset templates for the specific drum set. During the training stage, filters that isolate the head part of the wave are rewarded while filters that highlight the tail part are penalized. This training procedure evolves characteristic filters that are sensitive on detecting the onset part of the respective drum element. Experimental results with multiple drum sets indicate that the proposed system is fairly accurate and detects a great percentage of the onsets of each percussive element accurately.

Future work would provide enhancements in both the training and the real–time module. The training process would be improved if the population was initialized using some statistical information about the preset template drum elements. At the present form of the system, no a priori assumptions are made for the initial un–evolved characteristic filters, which makes training slower and less robust. On the other hand, the system would be also able to detect the intensity of each onset and not only its presence. This modification would require training on non–binary scenar-

ios, that incorporate information about the intensity of each percussive element. The system should also be tested with single microphone drum recordings in several rooms in order to examine its capabilities n real–world circumstances. Finally, the system should be tested on detecting onsets from non–drum percussive sounds.

5. REFERENCES

[1] E. Battenberg, V. Huang, and D. Wessel. Toward live drum separation using probabilistic spectral clustering based on the itakura-saito divergence. In *Audio Engineering Society Conference: 45th International Conference: Applications of Time-Frequency Processing in Audio*, 3 2012.

[2] J. P. Bello, L. Daudet, S. Abdallah, C. Duxbury, M. Davies, and M. B. Sandler. A tutorial on onset detection in music signals. *IEEE Transactions on Speech and Audio Processing*, 13(5):1035– 1047, Sept. 2005.

[3] J. A. Bilmes. *Timing is of the essence : perceptual and computational techniques for representing, learning, and reproducing expressive timing in percussive rhythm*. Thesis, Massachusetts Institute of Technology, 1993. Thesis (M.S.)–Massachusetts Institute of Technology, Program in Media Arts & Sciences, 1993.

[4] C. Dittmar. Drum detection from polyphonic audio via detailed analysis of the time frequency domain. In *6th International Conference on Music Information Retrieval ISMIR'05*, London, UK, Sept. 2005.

[5] D. Fitzgerald. *Automatic Drum Transcription and Source Separation*. PhD thesis, Dublin Institute of Technology, 2004.

[6] O. Gillet and G. Richard. Automatic transcription of drum loops. In *Acoustics, Speech, and Signal Processing, 2004. Proceedings. (ICASSP '04). IEEE International Conference on*, volume 4, pages iv–269 – iv–272 vol.4, 2004.

[7] M. Goto and Y. Muraoka. A sound source separation system for percussion instruments. In *Transactions of the Institute of Electronics, Information and Communication Engineers*, volume J77-D-II, pages 901–911, 1994.

[8] A. Klapuri. Sound onset detection by applying psychoacoustic knowledge. In *Acoustics, Speech, and Signal Processing, 1999. Proceedings., 1999 IEEE International Conference on*, volume 6, pages 3089 –3092 vol.6, 1999.

[9] J. Paulus and T. Virtanen. Drum transcription with nonnegative spectrogram factorization. In *13th European Signal Processing Conference (EUSIPCO 2005)*, Antalya, Turkey, 2005. Curran Associates.

[10] J. K. Paulus and A. P. Klapuri. Conventional and periodic n-grams in the transcription of drum sequences. In *Proceedings of the 2003 International Conference on Multimedia and Expo - Volume 1*, ICME '03, pages 737–740, Washington, DC, USA, 2003. IEEE Computer Society.

[11] K. Price, R. M. Storn, and J. A. Lampinen. *Differential Evolution: A Practical Approach to Global Optimization (Natural Computing Series)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2005.

[12] W. A. Schloss. *On the Automatic Transcription of Percussive Music - From Acoustic Signal to High-Level Analysis*. PhD thesis, Stanford University, Stanford, CA, 1985.

[13] R. Storn and K. Price. Differential evolution – a simple and efficient adaptive scheme for global optimization over continuous spaces. *Journal of Global Optimization*, 11:341–359, 1997.

[14] K. Yoshii, M. Goto, and Okuno. Automatic Drum Sound Description for Real-World Music Using Template Adaptation and Matching Methods. In *Proceedings of 5th International Conference on Music Information Retrieval*, Barcelona, Spain, 2004.