The HandSolo: A Hand Drum Controller for Natural Rhythm Entry and Production

Kunal Jathal
Department of Music and Performing Arts Professions
New York University
New York, NY 10012
kkj220@nyu.edu

Tae-Hong Park
Department of Music and Performing Arts Professions
New York University
New York, NY 10012
thp1@nyu.edu

ABSTRACT

The majority of electronic percussion controllers on the market today are based on location-oriented striking techniques, resulting in a "finger drumming" interaction paradigm, that is both fundamentally eclectic as well as imposingly "controllerist". The few controllers that allow hand-drumming techniques also invariably conform to region-based triggering design, or, in trade-off for expressivity, end up excluding hardware connectivity options that are vital to the context of the modern electronic rhythm producer. The *HandSolo* is a timbre-based drum controller that allows the use of natural, hand-drumming strokes, whilst offering the same end-goal functionality that percussion controller users have come to expect over the past decade.

Author Keywords

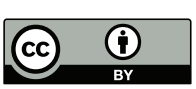
Hand Drumming, MIDI Controllers, Timbre Recognition

ACM Classification

H.5.5 [Information Interfaces and Presentation] Sound and Music Computing, H.5.2 [Information Interfaces and Presentation] User Interfaces—User-Centered Design

1. INTRODUCTION

Electronic percussion is one of the most well established electronic music interfaces after the piano keyboard; yet, these interfaces are often crude and traditionally only capture striking velocity at the moment of impact [20]. While the visual dimension of a percussionist's performance is rich with detail and subtlety [15], the current hardware controller model for rhythm production, however, is based on finger drumming techniques, and hence largely esoteric. The majority of commercial devices simply extend on the "piano" MIDI-keyboard controller design to try and offer interfaces and experiences only marginally catered to "beat" production. This traditional finger-drumming paradigm is problematic because it creates an unintuitive learning environment for drummers and percussionists, and additionally requires new skill sets that other proficient rhythmists need to acquire in order to produce beats. More importantly, however, it doesn't expand upon or allow percussionists to use skills they have honed over years of playing experience.



NIME'16, July 11-15, 2016, Griffith University, Brisbane, Australia.
.

Instead of the "location" and velocity based approach that exemplifies current drum and hand percussion controllers, models that enable "force-sensing" could offer more intuitive control, especially for the salient qualities associated with hand drumming [9]. Users find physical manipulation of digital sound well geared toward performance settings involving improvisational styles [12]. History has already demonstrated that through designing a controller that considers and utilizes percussionists' pre-existing gestural vocabulary, the possibilities for gestural sound control can be extended. Furthermore, new control paradigms could also enable elaboration in underexplored computer music and percussion performance areas [15]. The *KORG WaveDrum* is an example of such a device; it has thus far been the only commercial system that has been robust to hand-drumming playing gestures [1]. The *WaveDrum* is far more responsive and predictable than other percussion controllers, yet it's success was short-lived due to the absence of integration with MIDI and computer music production practices.

The *HandSolo* is an interface that aims to allow the entry and production of rhythm using existing natural hand drumming based techniques and physical nuances, to allow easier and more intuitive beat production for the purposes of both computer music production and performance.

2. COMMERCIAL CONTROLLERS

This section takes a brief look at some common contemporary commercial drum controllers on the market today, outlining their ideology, design, and trade-offs.

2.1 Location-Based Design

2.1.1 Finger Drumming Controllers

At the quintessential level, the large majority of commercial drum controllers on the market today contain the same layout and the same underlying technology. There is essentially a matrix or grid of rubber-based "pads", that tend to be color coded and velocity sensitive. Functionally speaking, the pads are generally multi-purpose; in addition to performing and producing drum patterns, they may be used to trigger musical events such as digital samples. Figure 1 shows the *Akai MPD 18* and *Native Instruments' Maschine*, the latter of which, though advertised as a complete music production system, essentially encapsulates the standard form that nearly all MIDI drum controllers on the market adhere to today. As mentioned above, the fundamental problem with finger drumming is that it introduces a paradigm of drumming that does not incorporate existing hand-drumming techniques, and instead emphasizes controllerist gestures and visual flair. In this way, the intuition and natural flow of hand strokes is not accommodated for.

(a) *NI Maschine* (b) *Akai MPD18*

Figure 1: Current Commercial Drum Controller Design

2.1.2 Hand-Drumming Controllers

The *Roland Handsonic* hand-drumming percussion device computes surface strike location by cross-referencing the force-sensing resistor (FSR) and piezo sensor outputs [19]. Certain zones are pressure sensitive as well, and patches built in the device use this ability to offer supplementary musical functionality. Despite its sensitivity, as all of the stikes are "discretized", it is inadequate in capturing subtle interactions: gestures like stirring with brushes or sliding a hand on the pad will not produce reliable output [10]. More importantly, however, the *HandSonic* is still a location-based trigger device and is thus functionally identical to the *Maschine*, with the only notable exception being the layout change in an attempt to match hand-drumming orientations. The problem, however, is that while drumhead timbre in general is a function of the stroke location, it is *also* a function of the stroke technique itself. That is, hand drumming strokes are unique as the technique and its resulting timbre are defined by both where one strikes the surface and *how* the surface is struck.

Figure 2: The *Roland Handsonic*, a hand-drum controller

2.2 Timbre-Based Design

2.2.1 Hardware Controllers

The *KORG WaveDrum* was one of the first commercial devices that actively based its sound synthesis engine on physical modeling, but it's main claim to fame was its novel approach of actually incorporating the acoustic response of the drumhead itself, in order to directly drive the synthesis algorithm. The tactic of utilizing a portion of the actual sound present in the drumhead resulted in the *WaveDrum* being arguably the most responsive and predictable percussion system in its league in the market [17]. The limited MIDI functionality and connectivity options of the *WaveDrum* with Digital Audio Workstations (DAWs) were shortcomings [3]. The *WaveDrum* can send and receive MIDI data to and from other MIDI hardware, allowing it to be controlled by, or be in control of, a MIDI device. However, it cannot send the actual sound of the drumhead over MIDI, so nearly all it's expressionality is lost in this way. The *WaveDrum* was sold for a relatively short period of time, at a prohibitively high cost; a likely contributing factor to its demise.

Figure 3: The *KORG WaveDrum* pioneered the commercial use of timbre

2.2.2 Software Controllers

Modern smartphones, equipped with various sensor technologies, are powerful interactive tools capable of a variety of complex media and data processing. *TableDrum* is a simple yet novel Apple iOS application that allows users to 'play' drum samples by 'playing' physically generated, distinguishable sounds, in order to trigger the samples themselves. The app essentially analyzes the incoming audio signal via the Discrete Fourier Transform (DFT) and then 'fingerprints' it to determine which sample to trigger [2]. The classification is quite robust with distinctive sounds, but playing different strokes on the same surface does not work effectively. Even between separate surfaces, overlapping strokes quickly and easily ambiguate the frequency spectrum, consequently degrading identification accuracy [2]. In addition, the lack of MIDI connectivity gears the software away from its utility as a controller for more serious musicians and is perhaps designed to address the needs of general users for entertainment purposes.

Similar to *WaveDrum*, *Impaktor* is an app that analyzes the spectrum and dynamics of the actual hand drumming audio signals to drive its synthesis engine, primarily based on physical modeling techniques [4]. However, because *Impaktor* is driven by incoming audio, its operation generally requires an acoustically isolated environment, thus making it difficult to use in environments such as ensemble performance settings. It is therefore recommended to use headphones instead of the phone's built-in speaker in order for the generated sound to stand out better, as well as to avoid creating a feedback loop and re-triggering unwanted sounds [4]. Like *TableDrum*, *Impaktor* also lacks MIDI connectivity, in trade-off for its high expressionality. For both these apps, the lack of MIDI connectivity and audio-driven modus operandi cause them to be largely impractical in a performance and production context. There are a handful of other esoteric controllers currently on the market, but even commercial solutions that are "force" based use only velocity, and/or surface location, to demarcate samples. By providing a stroke-based paradigm, the *HandSolo* aims to alleviate some of these shortcomings whilst concurrently allowing for a more natural manner of rhythm entry.

(a) *TableDrum* (b) *Impaktor*

Figure 4: Operational Modes for iOS Timbre-based Drum Apps

3. *HANDSOLO*: DESIGN

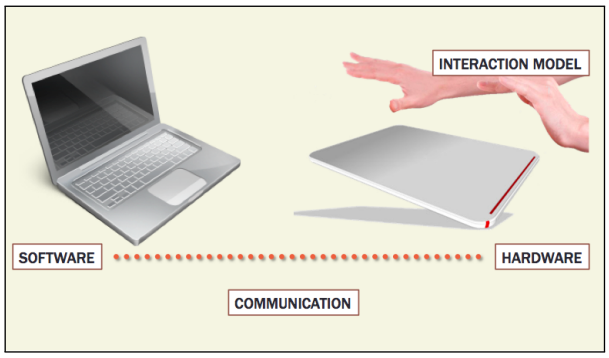


Figure 5: *HandSolo* High Level Concept

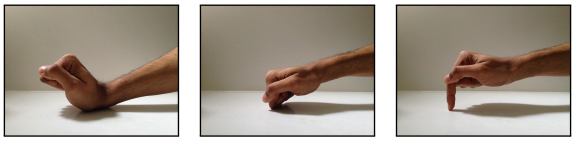
Figure 5 shows a simple overview of the *HandSolo* controller concept: (1) the user produces strokes on a surface, (2) the physical gestures are simultaneously transduced into an acoustic signal, and (3) the stroke's acoustic representation is then analyzed on a computer and appropriate percussion audio is synthesized. A typical, wooden, tabletop is the quintessential example of a surface in this context. A contact microphone attached to the table serves as the only communication link from the table to the computer. In this way, the acoustic vibration data itself is captured and relayed to a real-time software analysis system running on the computer. The timbre captured through stroke analysis is then processed by the system and automatically classified with a small amount of latency. The classification result can be output as a MIDI message, which can then be mapped to any software instrument; because this paper focuses on hand-drumming, percussive programs will be selected. Through this design, various hand strokes are mapped to different drum sounds. MIDI output in lieu of the actual classification index allows the system to be connected to any Drum Machine, VST, or Sampler, to allow for a plethora of drumkit choices as per user discretion. The "hardware" can be *any* surface, and the hardware sensor is simply a standard contact microphone.

3.1 Interaction Model

During the course of design evaluation, the decision was made to keep the number of stroke classifications low. Strokes were empirically analyzed for their spectral content, and the driving incentive was to select a palette that would provide reasonably varied spectral content: that is, the strokes decided upon were based on those with the widest range of frequency content that would be easiest to differentiate algorithimically. The final motivation was to keep the stroke detection accuracy high and the algorithm design efficient. After empirical analysis of different table drumming techniques and possibilities [21], three different gestures were chosen for detection: the Heel Stroke, the Touch Stroke, and a modified version of the Open Stroke (using the knuckles instead of the pads of the fingers). Figure 6 shows the strokes. In the context of *HandSolo* design and its eventual MIDI map, the Heel Stroke represents a bass drum, the Open Stroke a snare, and the Touch Stroke a hi-hat.

3.2 Software Design

Timbre recognition-based instruments use timbre as a control parameter. Such systems utilize digital signal processing and machine learning techniques to classify the timbre of the instrument, outputting user-defined labels for this purpose. The labels correspond to different playing techniques



(a) Heel Stroke (b) Open Stroke (c) Touch Stroke

Figure 6: *HandSolo* Strokes

that have been identified to the system from human labelled instances. Since different playing techniques produce different timbres, it is a matter of collecting training instances of the desired technique and retraining the classifier [20].

Timbre classification systems can be broken down into three main stages:

- Onset Detection
- Feature Extraction
- Classification

3.2.1 *Onset Detection*

The *HandSolo* onset detection system is based on the *high frequency content* (HFC) function to strategically focus on the part of the audio signal that provides important onset information: energy increases linked to transients in the spectral domain tend to appear as broadband events. As much of the signal's timbral energy is concentrated at low frequencies, changes due to transients are more noticeable at high frequencies [18]. To exploit this phenomenon, the input signal is subject to pre-emphasis and weighted toward high frequencies before summing to obtain a weighted energy measure:

$$\tilde{E}(n) = \frac{1}{N} \sum_{k=-\frac{N}{2}}^{\frac{N}{2}-1} W_k \left| X_k(n) \right|^2 \quad (1)$$

where $X_k(n)$ is the Short-time Fourier Transform (STFT) of the time series $x(n)$, and W_k is the frequency dependent weighting factor. Using Parseval's theorem, if, $W_k = 1 \forall k$, then $\tilde{E}(n)$ is equivalent to the local energy. Additionally, $W_k = k^2 \forall k$ then produces the local energy of the derivative of the signal [5]. The high frequency content (HFC) function, proposed by Masri in [11], with $W_k = |k|$ linearly weights each frequency bin's contribution in proportion to its frequency. This produces sharp peaks during attack transients, providing notably better performance with percussive onsets, where transients are well modeled as bursts of white noise [5].

3.2.2 *Feature Extraction*

Brent in [6] outlines the feature set of most commonly used temporal and spectral analysis tools that are relevant and useful for percussive timbre identification. Several features were tested for the *HandSolo*. Because the complexity of the feature vector increases computation time (and hence latency), the goal was to pick a minimal feature set that would provide a good balance of accuracy and speed. In the end, a Bark-Frequency Spectrum proved to be the most reliable and performant feature. A Bark-Frequency spectrum is a warping of the normal magnitude (or power) spectrum to the Bark scale. This attenuates some of the high frequency detail while maintaining resolution on the low end. A triangular filter-bank spaced at half-bark spacing is used in the *HandSolo*, and as the strokes mostly contain low frequency content, only the first five bands are extracted. Equation 2 shows the definition of a bark with respect to frequency, f.

$$Bark = \left[26.81 \times \frac{f}{1960 + f} \right] \quad (2)$$

3.2.3 Classification

There are three separate classifiers (one for each stroke) in the *HandSolo* and each employs the k-Nearest Neighbor (kNN) algorithm. Each classifier labels the instance as one of two possible classes: either corresponding to the stroke, or not. The Open Stroke classifier will thus label an instance as either 'an open stroke' or 'not an open stroke', the Heel Stroke classifier labels an instance as either 'a heel stroke' or 'not a heel stroke', and the Touch Stroke classifier will label an instance as either 'a touch stroke' or 'not a touch stroke'. The *HandSolo* uses user-defined input to force manual clustering of the training data and thus strategically implements a supervised learning system. The advantage of supervised learning is that, for simple systems with similar training and 'real-world' testing data, reliability and accuracy of classification is boosted. The testing mode of the classifier is essentially the modus operandi ('performance mode') of the *HandSolo*. In testing mode, after the feature vector data from all training instances has been appropriately clustered, the classifier receives new feature vector data from test instances (i.e. bark data being generated from actual strokes in the performance) and attempts to categorize the test instance with the correct class label. The *HandSolo* currently uses Euclidean distance measurements for kNN, although the system can easily be configured to use the Manhattan distance or the Pearson Correlation Coefficient.

Table 1: Configurable metrics of the *HandSolo*

Metric	Formula
Euclidean Distance	$\|a-b\|_2 = \sqrt{\sum_i (a_i - b_i)^2}$
Manhattan Distance	$\|a-b\|_1 = \sum_i \|a_i - b_i\|$
Pearson Correlation	$\rho_{X,Y} = \frac{cov(X,Y)}{\sigma_X \sigma_Y}$

Each classifier is also able to compute a confidence measure for its final classification decision. In the event of a tied classification between classifiers, the classification with the highest confidence measure is output. The final output label is sent to the MIDI patch for triggering the appropriate MIDI sample. By default, the *HandSolo* increments the class index to map a Heel Stroke to a MIDI bass drum, to map an Open Stroke to a MIDI snare drum, and to map a Touch Stroke to a MIDI hi-hat.

4. *HANDSOLO*: IMPLEMENTATION

4.1 Hardware

We tested our system on a standard Macbook Air running on a 1.3 GHz Intel Core i5 processor, 4 GB 1600 MHz DDR3 RAM, and 250 GB of SSD Storage. A Focusrite Saffire 6 USB audio interface was used to connect the contact microphone, a KORG CM-200, to the MacBook. The CM-200 is fashioned in a clip form factor, ideal for attachment to tables, counter tops, or any relatively uniform board-like surface. A standard wooden table surface was used, with dimensions of 35" x 36" x 0.75".

4.2 Software

The system running on the computer was entirely written in *Pure Data* (Pd). In addition, several open source libraries were used including [6], [7], [13], and [16].

(a) KORG CM-200 (b) Attachment to tabletop

Figure 7: *HandSolo* contact microphone placement

Figure 8: *HandSolo* Sample Setup

The *aubioOnset*[1] library was used for onset detection. There are three separate stroke channels (one for each stroke), that all have identical structure. Each channel has it's own feature extraction module; *barkSpec*. The *barkSpec* object from the *timbreID*[2] library performs bark frequency magnitude spectrum coefficient generation on the captured frame of audio. The list output by *barkSpec* is then fed to the *timbreID* object, the central classifier object of the HandSolo. The *timbreID* object is the "brains" of the *HandSolo*. It performs both the machine learning function during training, as well as the classification algorithm during performance mode. Once all training instances for all strokes have been entered, the system is put out of training mode, and appropriate cluster messages are sent to all *timbreID* objects to effectively pair instances (input values to the system) with strokes (output values of the system).

In performance mode, the first *timbreID* outlet reports the index of the nearest match instance, or, because clustering has been performed, the index of its associated cluster. Because the classifiers are binary classifiers, they output a class index of either 0 (for the stroke being detected) or 1 (for the 'non-stroke' being detected). The class label and its corresponding confidence measure are sent to a patch that compares all classification results with their confidence measures and outputs a single class label to the MIDI patch, for conversion to the appropriate drum sound. MIDI output is simplistically implemented in Pd by formatting the class index into a MIDI note, which is then routed to the desired channel through a bus. The bus may be connected to a DAW, routed through a MIDI track, and/or sent to a Sampler or VST of choice.

[1] `http://aubio.org/`
[2] `https://puredata.info/downloads/timbreid/`

5. RESULTS

5.1 Device Testing

Device Testing covers testing done on the system to measure technical metrics, such as stroke classification success rates, surface tolerance, and latency. Unless stated otherwise, testing was carried out on a total of 150 instances; 50 instances for the Open, Heel, and Touch strokes each. The single nearest neighbor was searched for, only Bark Spectrum feature data was used, and the surface utilized was a standard, wooden table top with dimensions 36" x 35" x 0.75". The WEKA [14] suite of machine learning software was used to run data analysis. The instances were recorded in *HandSolo's* training mode, which allows for saved feature vector data to be exported in Attribute-Relation File Format (ARFF) format. The ARFF database was then edited to annotate the stroke types for each row of feature vector data, and subsequently fed into the WEKA preprocessor, setting the stroke type as the class to pivot around. The attributes were then piped to the kNN classifier, using 10-fold cross validation to generate the data set. Success rates presented are the lowest common denominator of success amongst all classifiers, and calculated as the percentage of correctly classified instances.

Table 2: kNN Success Rates vs. k

k	1	3	5	9	15
Success	98.67%	98.67%	98.67%	98.67%	98.67%

Table 3: kNN Accuracy vs. # of Instances per Stroke

Training Instances per Stroke	10	25	50
kNN Success	96.67%	97.33%	98.67%

Table 4: Classification Algorithm Accuracy

Algorithm	kNN	SVM	k Means	Neural Network
Success	98.67%	95.33%	92.67%	98.00%

Table 5: Features vs Classification Success Rate

Feature	BFCC	MFCC	Centroid	Bark Spectrum
Success	97.33%	96.67%	70.00%	98.67%

Table 6: Classification Success vs Surface

Surface	Wood	Glass	Ceramic	Cloth
Success	98.67%	99.33%	97.33%	96.67%

5.2 User Evaluation

The two primary metrics assessed through user experimentation were the accuracy of the system (measured in terms of stroke classification), and the general usability of the device. 12 participants were recruited for experimentation. After being instructed about the various strokes and training the system, participants were asked to reproduce two rhythms at a fixed tempo (100 BPM). Each rhythm was performed twice over the course of 16 measures.

Table 7: Individual and Total System End-to-End Latency

Stage	Audio Interface	Onset Detector	Classifier (incl. FE)	MIDI Out	Total
Latency	10 ms	12 ms	5 ms	6 ms	**33 ms**

During this time, the system recorded the number of user strokes that were classified incorrectly. Results of user experimentation are outlined in Table 8 below, portraying stroke classification accuracy as a percentage of the total number of strokes played per participant. Participants were grouped and arranged by rhythmic skill, in ascending order (i.e. 1-4: beginner, 5-8: intermediate, 9-12: advanced).

Table 8: Stroke classification accuracy by participant

User	1	2	3	4	5	6	7	8	9	10	11	12	All
Hit Strokes	58	60	59	56	64	62	66	68	68	67	68	68	**764**
Missed Strokes	10	8	9	12	4	6	2	0	0	1	0	0	**52**
Total Strokes	68	68	68	68	68	68	68	68	68	68	68	68	**816**
Success (%)	85	88	87	82	94	91	97	100	100	99	100	100	**94**

6. ANALYSIS

6.1 Device Testing

kNN proved to be a considerably efficient choice: the algorithm worked with high success rates for low values of k; in addition, these success rates stayed constant with increasing values of k. The number of training instances provided to the system showed direct proportionality with kNN accuracy, although with quickly diminishing returns (doubling the number of instances from 25 to 50 only produces a 1.34% increase in accuracy). Bark-based features produced the most accurate classification results, with both the Bark Magnitude Spectrum (98.67%) and BFCC (97.33%) ranking above MFCC (96.67%). These results are consistent with literature review comparing Bark and Mel features [6], [8], [13]. Any relatively firm plane with distinct timbre and no external surface vibration was found to be the best setting for the *HandSolo* system. It worked best on surfaces that had distinct timbres: glass proved to have the highest success rate, likely because of the Heel and Open strokes that produce more distinct timbres (than on wood). Latency was low, allowing meaningful real-time performance.

6.2 User Evaluation

While the data is statistically insignificant, User Evaluation shows promising results for the *HandSolo*. In a questionnaire filled by participants, ease of use and the feel of the device were both highly rated, receiving 83% and 92% of positive answers ("Easy/Somewhat Easy", and "Satisfied/Somewhat Satisfied" respectively). In addition, 75% of participants answered "Yes" to considering adopting the *HandSolo* as their primary drum controller. One category whose implications are interesting to analyze, is user skill level. For instance, all participants that indicated they played a hand-drum / percussion instrument (i.e. were percussionists) also answered "Easy" for ease of use. This bodes well for the *HandSolo's* aim of providing an experience that utilizes existing percussionist repertoire to allow natural rhythm entry.

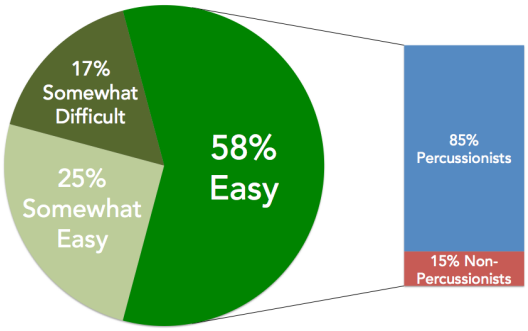


Figure 9: Link between ease of use and hand-drum skill: percussionists found the *HandSolo* easy to use

In addition, Figure 10 shows that, with increasing user skill, *HandSolo* classification accuracy increases as well. One hypothesis in this regard has to do with the difference in the training and testing portion of experimentation. During training, strokes are individually performed, but during testing, they are performed as part of a rhythm: our assertion is that experienced percussionists are more likely to reproduce strokes more consistently (in both isolated and rhythm contexts), while inexperienced players will exhibit greater variance in differing contexts.

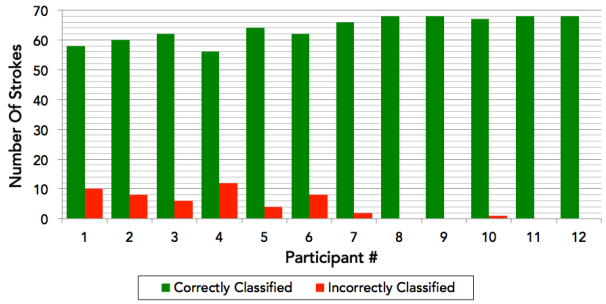


Figure 10: Stroke classification accuracy by participant

7. CONCLUSIONS

This paper presented a novel design for an electronic drum controller based on hand drumming techniques. A robust timbre recognition system was developed in *Pure Data*, capable of recognizing three distinct strokes with high reliability and low latency, allowing it to be used in a real-time performance setting. MIDI output was also incorporated into the interface, permitting use with DAWs and hence extending its utility for the premise of industry-standard computer music production. The system may be improved for greater accuracy through the implementation of a weighted feature set, using search heuristics. A non-supervised classification scheme may be explored in an attempt to further boost reliability. Options for increased expressive potential may be explored through the incorporation of velocity sensitivity, an increased number of recognized strokes, and deeper timbral analysis. Eventually, the system may also be ported to a mobile platform, such as iOS.

8. REFERENCES

[1] R. M. Aimi. *Hybrid percussion: Extending physical instruments using sampled acoustics*. PhD thesis, Massachusetts Institute of Technology, 2007.

[2] K. Audio. Kvr forum: Tabletop drumming to midi tracks (production techniques), 2015.

[3] AudioNewsRoom'. Korg wavedrum wd-x review, 2009.

[4] BeepStreet. *Impaktor User Manual*. Impaktor, 2015.

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

[6] W. Brent. Cepstral analysis tools for percussive timbre identification. In *Proceedings of the 3rd International Pure Data Convention, Sao Paulo, Brazil*, 2009.

[7] W. Brent. Physical and perceptual aspects of percussive timbre. 2010.

[8] P. Herrera, A. Dehamel, and F. Gouyon. Automatic labeling of unpitched percussion sounds. In *Audio Engineering Society Convention 114*. Audio Engineering Society, 2003.

[9] R. Jones, P. Driessen, A. Schloss, and G. Tzanetakis. A force-sensitive surface for intimate control. In *Proceedings of the International Conference on New Interfaces for Musical Expression (NIME)*, pages 236–241. Citeseer, 2009.

[10] J. Krogh. Keyboard reports: Roland hpd-15 handsonic: Electronic hand percussion instrument. *Keyboard Magazine*, 27(2):108–114, 2001.

[11] P. Masri. *Computer modelling of sound for transformation and synthesis of musical signals*. PhD thesis, University of Bristol, 1996.

[12] D. Merrill, H. Raffle, and R. Aimi. The sound of touch: physical manipulation of digital sound. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 739–742. ACM, 2008.

[13] M. Miron, M. E. Davies, and F. Gouyon. An open-source drum transcription system for pure data and max msp. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pages 221–225. IEEE, 2013.

[14] T. Ngo. Data mining: practical machine learning tools and technique, by ian h. witten, eibe frank, mark a. hell. *ACM SIGSOFT Software Engineering Notes*, 36(5):51–52, 2011.

[15] J. Oliver and M. Jenkins. The silent drum controller: A new percussive gestural interface. In *Proceedings of the International Computer Music Conference*, pages 651–654, 2008.

[16] M. S. Puckette, M. S. P. Ucsd, T. Apel, et al. Real-time audio analysis tools for pd and msp. 1998.

[17] G. Reid. 40 years of gear: The history of korg, 2002.

[18] X. Rodet and F. Jaillet. Detection and modeling of fast attack transients. In *Proceedings of the International Computer Music Conference*, pages 30–33, 2001.

[19] K. Senda and Y. Miyamoto. Apparatus and method for detecting and processing impacts to an electronic percussion instrument. *Acoustical Society of America Journal*, 115:461–461, 2004.

[20] A. R. Tindale, A. Kapur, G. Tzanetakis, P. Driessen, and A. Schloss. A comparison of sensor strategies for capturing percussive gestures. In *Proceedings of the 2005 conference on New interfaces for musical expression*, pages 200–203. University of British Columbia, 2005.

[21] V. Välimäki. *Sonically Augmented Table and Rhythmic Interaction*. PhD thesis, Aalto University, 2011.