

STARR: Similarity-based Transfers for Automatic Rock Rhythms

David Thue

Department of Computing Science
University of Alberta
Edmonton, Alberta, Canada, T6G 2E8
dthue@cs.ualberta.ca

Introduction

During the composition of a piece of music, composers often focus on the crafting of melody and harmonic lines rather than a percussive track that may accompany them. For amateur composers, creating a drum track by hand can be both tedious and frustrating, due to their lack of skill and experience with the task. In recent years, computer software tools have been created to help automate the creation of drum tracks, but most often they operate by simply allowing the composer to choose one of a set of drum patterns, and continually repeating that pattern for the duration of the song. As one might expect, this technique often results in aesthetically displeasing drum tracks, as their lack of variation quickly leads to listener boredom. The primary solution to this problem used by human drummers is the insertion of drum fills - percussive elements that differ from the current drum pattern, intended to add energy and excitement to the performance of a musical piece by promoting listener interest. Unfortunately, the task of choosing both *when* to play a drum fill and *what* fill to play is non-trivial; the talents of human drummers are often rated by their ability to make these choices well. In this paper, I propose a novel solution to the problem of automatically choosing where in a melody-accompanying drum track to place drum fills; this problem is henceforth referred to as the Drum Fill Placement problem. Using a database of human-crafted melodies and accompanying drum tracks, a machine learning algorithm is trained to identify and exploit a relationship between melody rhythms and the occurrence of drum fills. Given a set of test melodies, the resulting drum fill placements are evaluated by comparison to hidden, human-crafted drum tracks designed to accompany each melody. The remainder of this section introduces a set of terminology needed for discussing the Drum Fill Placement problem, gives a brief background concerning the choice of music representation used, and formally states the goals of this work. Following this, the remainder of the paper is organized as follows: first, some specific refinements to the Drum Fill Placement problem as it is treated in this work are defined; second, a set of related work is discussed, with a focus on how it might be useful in solving the Drum Fill Placement problem; third, a novel algorithm designed to solve the Drum Fill Placement problem is introduced, titled STARR (Similarity-based Transfers for Automatic Rock Rhythms); fourth, a set of experiments

designed to achieve the goals of this project are described, and results are given; fifth, the experimental results are discussed; finally, future work is suggested, and conclusions are drawn.

Terminology

The following is a list of the terminology used throughout this paper that is specific to musical drumming. A basic understanding of musical beats and bars is assumed.

- **Drum Track:** *The set of all sounds produced by percussive instruments during the course of a song.*
- **Drum Pattern:** *A set of percussive sounds, spanning one bar, that repeats frequently during a song.*
- **Drum Fill:** *A set of percussive sounds, spanning one bar, that differs from the drum patterns in a song.*
- **Drum Fill Density:** *The percentage of bars in a drum track in which drum fills are played.*
- **Drum Fill Placement:** *The task of choosing in which bars of a drum track a drum fill should be played.*
- **Melody:** *The set of all sounds produced by non-percussive instruments during the course of a song.*
- **Melody Energy:** *The number of notes played per second in a given melody.*

Project Goals

The primary goal of this project is to devise an automatic solution to the Drum Fill Placement problem, using a Machine Learning approach. Given a set of n examples consisting of melodies with human-crafted, accompanying drum tracks, n -fold cross-validation will be performed, comparing the placement and rhythmic content of drum fills in a generated drum track with those found in the original, human-crafted track. The hypothesis upon which the approach used in this paper is based is as follows:

Primary Hypothesis. *Both the position and rhythmic content of drum fills in a drum track depend on the rhythm of the melody in the bars in which they are placed.*

This hypothesis intuitively reasonable, as drum fills are often described as “filling in the holes” (silence) of the melody that they accompany. My hypotheses concerning the performance of my solution to the Drum Fill Placement

problem are given below; each of them is intended support the truth of the Primary Hypothesis, and testing them is a sub-goal of this project.

Hypothesis 1. *The placement of drum fills in the generated track will match the placement of drum fills in the original, hand-crafted track with high accuracy (greater than 80%).*

Hypothesis 2. *The rhythms of correctly placed fills will, with moderate accuracy (greater than 60%), match the rhythms of the corresponding human-crafted fills within the maximum possible distance that the rhythms of their corresponding melodies could match each other. (The concept of rhythms matching within a given distance is explained later on).*

Hypothesis 3. *The accuracy of drum fill placement will improve when the fill density of the generated track is moderated by the energy of the melody.*

Problem Refinement

This section describes and justifies the specific refinements, simplifications, and assumptions that have been made concerning the domain of the Drum Fill Placement problem, for the purposes of the work discussed in this paper.

Music Representation

Much of the work performed to-date in the field of Computer Music focuses on the analysis of analog waveform signals to perform beat detection, meter-finding, or part extraction. While the automatic production of drum tracks based on examples would normally require all three of these processes, using an alternate representation, namely, the Musical Instrument Device Interface (MIDI), allows these steps to be avoided entirely (MMA 2001). Created in 1981, the MIDI specification provides a compact, digital representation of every note, rhythm, and volume played by every instrument in a given song. Beats are expressed directly based on the time of their occurrence (in seconds) relative to the start of the song (so they need not be detected), the meter can be computed by counting beats and dividing by the elapsed number of seconds, and part extraction is reduced to a table lookup problem.

Simplifications

- As discussed previously, using a MIDI representation of music versus analog waveform signals eschews the problems of beat detection, meter-finding, and part extraction.
- The set of melodies and drum tracks used for training and testing consists entirely of Beatles rock songs. Given the huge popularity of the Beatles, high-quality MIDI files of their music are readily available on the Internet (Shears 2006).
- The training and testing set is restricted to songs with rock beats (straight 8th notes). Mixing straight & swung eighth notes occurs infrequently, and drum fills from swing beats have different rhythms than those from rock beats (this would dampen the accuracy tested in Hypothesis 2).

Assumptions

- The smallest division of beats in a bar is 1/32 of the bar length; this keeps the size of the feature vectors small (length 32), and rhythms occur at a finer accuracy (greater than 32 beats/bar) very infrequently.
- At least once in any drum track, three repeated bars of a drum pattern will be interrupted by at most two bars of drum fills. This simplifies the detection of drum fills in training examples.
- All training and testing songs are in 4/4 time (4 sets of 8 32nd beats per bar); time signatures are difficult to detect in MIDI files.

Related Work

Although I am unaware of past or current work that specifically attempts to solve the Drum Fill Placement problem, the following is a summary of relevant work in closely related areas. For each method discussed, its potential uses with respect to the approach presented later in this paper are discussed.

In 2002, Paulus and Klapuri investigated measuring the similarity of rhythmic patterns in waveform signals (Paulus & Klapuri 2002). Their goal was to devise a method that would declare rhythms to be similar regardless of both instrumentation and the time-scale at which they were played. Although their method was successful, its application in the technique that I propose in this paper is limited. The MIDI format makes finding similar rhythms independently of instrumentation trivial, and rhythms that differ only in time-scale often lead to drastically different drum fills; rarely would a human drummer play a 4-beat fill over 8-beats instead.

In 2003, Paulus and Klapuri extended their work to the task of automatically transcribing drum tracks, effectively yielding a method for converting from waveform signals to a MIDI representation of a given song (Paulus & Klapuri 2003). This method could potentially be applied as a preprocessing step to extend the work presented here to the domain of waveform signal drum tracks, but the reported accuracy of the transcriptions generated was deemed too low to be of use (~54% correct). Another relevant aspect of the same work by Paulus and Klapuri is their use of N -Grams for predicting which set of percussive instruments would be playing at any given instant, in an attempt to capture the sequential dependency that exists from one beat to the next in a drum track. Although an N -Gram approach might be useful in terms of predicting when a drum fill should be played based on the sequence of fills (or no fills) that had been played in the previous $N - 1$ bars, the hypothesis that this paper explores is that fill placement depends on *current melody rhythm*, not previous fill placements.

Also in 2003, Parry and Essa continued the study of assessing the similarity of rhythmic patterns, based on Tanguiane's concept of *elaboration* (Parry & Essa 2003; Tanguiane 1993). Treating sequences of note onsets (rhythms) as bit vectors, rhythm R_i is defined to be an elaboration of rhythm R_j if $R_i \cdot R_j = R_j \cdot R_j$, that is, if it contains all of the

same onsets as R_j , and potentially more. While this measure of similarity could potentially be useful in distinguishing drum fills from drum patterns in a given drum track, in this work I consider a drum fill to be any rhythm that is *at all* different from the drum patterns in the track.

Two recent papers by Aucouturier and Pachet seek to generate drum tracks based on real-time interactive exchanges with a human player, using on Concatenative Sound Synthesis (CSS) (Aucouturier & Pachet 2005a; 2005b; Schwarz 2000; 2005). CSS is a synthesis technique in which sound is generated by choosing and concatenating entries from a large database of sound samples; the database is constructed by extracting samples from a set of source sound files, and samples are chosen for concatenation based on how well they match a set of given targets in terms of both a distance function and a concatenative quality function. Although the goals of this project differ from those of Aucouturier and Pachet, Concatenative Sound Synthesis appears to lend itself well to the task of solving the Drum Fill Placement problem using a Machine Learning approach.

Proposed Approach

As a potential solution to the Drum Fill Placement problem, I propose a novel algorithm, titled STARR (Similarity-based Transfers for Automatic Rock Rhythms). Based largely on Concatenative Sound Synthesis, STARR performs each of the 5 CSS tasks described by Schwarz (Schwarz 2005):

- *Analysis* - the segmentation of a song into discrete units
- *Database* - the collection of elements to be used for concatenation
- *Target* - the description of a way to compare elements during Selection
- *Selection* - the choosing of an element from the database that best matches each target
- *Synthesis* - the concatenation of all selected elements into a single song

The rationale for choosing a CSS-style approach over other potential alternatives was discussed in the previous section. The following is a description of how STARR implements each of the CSS tasks; the points at which its implementation differs from traditional CSS will be made clear.

Analysis

In STARR, the Analysis phase of CSS consists of processing a large set of MIDI music files that contain both melody and an accompanying drum track. Using a simple Unsupervised Learning technique, every bar of each song is automatically labelled as “fill” or “pattern”, referring to whether the bar contains a drum fill or a drum pattern. The fill-detection technique used requires two passes through the MIDI data; the first leverages the assumption that at least once in any drum track, three repetitions of a drum pattern will be interrupted by at most two bars of drum fills. The second pass corrects for any bars of drum patterns having been mistakenly labelled as fills before the pattern was detected. Although no empirical data has yet been gathered concerning

the performance of this classifier, a human expert’s inspection of the labels while listening to the MIDI files gave a favourable evaluation. Given that the stated assumption is fairly weak, verifying the accuracy of this classifier is left as future work.

Database

Given a set of melodies and drum tracks with each bar labelled as a fill or a pattern, STARR builds a database consisting of every melody/fill pair encountered, bar by bar; bars containing patterns are discarded. For example, given three bars, the first labelled “fill” and the last two labelled “pattern”, the first bar’s melody and accompanying drum track would be entered as a pair in the database, and the last two bars would be ignored. The resulting database contains a complete description of every bar of drum fill encountered in the training set.

Target

Given a new MIDI melody without an accompanying drum track, STARR forms targets for the Selection phase by computing the rhythm of each bar of the melody as a non-negative integer vector. Under the assumption of the smallest beat division being 32 beats per bar, each vector has length 32, and the value of each entry indicates the number of note onsets that occur on that beat (multiple percussive instruments could be playing simultaneously).

Selection

Given the rhythm of each bar of the new MIDI melody, the Selection process consists of both choosing whether a drum fill should be played, and if so, which drum fill should be played. In STARR, both of these choices are made by the same mechanism: a Nearest Neighbour search of the database based on measuring the Euclidean distance between the rhythm of the current bar of input melody and the rhythms of every bar of melody in the database. The size of the neighbourhood returned is controlled by a threshold on the allowed distance between rhythms, and when multiple melody rhythms have equal distance, one is chosen randomly with equal probability of choosing each one. Once a melody has been selected from the database that best matches the current bar of input melody, its accompanying drum fill is marked for processing by the Synthesis phase. If the Nearest Neighbour search returns no matches, STARR assumes that a drum fill should *not* accompany the input melody in that particular bar. This work diverges from the traditional implementation of CSS in that no concept of “concatenative quality” is measured; an investigation of such measures remains as future work.

Synthesis

Once fill/not-fill selections have been made for every bar of the new MIDI melody, STARR generates an accompanying drum track by concatenating together a sequence of either the drum fill bars that it selected to best fit the melody, or silent bars where it determined that no fill should be played.

It is from the Selection and Synthesis steps that STARR obtains its name; based on the *similarity* of the input melody’s rhythms to those found in the database, the corresponding drum fills are *transferred* into a new drum track. For aesthetic reasons, a repeating drum pattern is added to every bar of the generated track.

Experimental Design and Results

To test the hypotheses given previously concerning STARR’s performance, a large set of MIDI songs with drum tracks was required. To this end, MIDI files of 89 Beatles songs were obtained from the Internet; the fact that they were all freely available from a single website simplified this task greatly (Shears 2006). As mentioned previously, only songs in 4/4 time with exclusively straight (vs. swung) eighth notes were used, due to the fact that detecting time signature using the MIDI format is difficult, and that drum fills from straight and swung music are rarely combined in a single song. The following sections describe how each of Hypotheses 1, 2, and 3 were tested, with the goal of establishing support for the Primary Hypothesis.

Fill Placement Accuracy

To test Hypothesis 1, STARR was run using 89-fold cross-validation on the 89 Beatles MIDI files; that is, for each of the 89 songs, the other 88 were used to build a CSS database, and the melody from the remaining song was separated from its accompanying drum track and passed into STARR as a new input. STARR then generated a drum track to accompany the new input, consisting of a sequence of bars, some of which contained drum fills. This result was then compared to the drum track that was originally obtained in the MIDI file for that melody, by first detecting the drum fills in the original drum track, and then examining where STARR chose to place its drum fills. For every song, three values were recorded concerning STARR’s performance: the percentage of correct fill/non-fill placements, the percentage of erroneous fill additions (STARR placed a fill where the original track had none), and the percentage of erroneous fill omissions (the original track had a fill where STARR placed none). The results of this experiment, averaged over all songs and for several different fixed similarity-neighbourhood sizes (maximum allowed distances between rhythms), are given in Table 1.

Maximum Distance	% Correct Placements	% Erroneous Additions	% Erroneous Omissions
0.0	62 ± 2	1.0 ± 0.2	36 ± 2
1.0	62 ± 2	1.6 ± 0.3	36 ± 2
2.0	60 ± 2	8 ± 1	32 ± 2
3.0	52 ± 2	25 ± 2	23 ± 2
4.0	47 ± 2	38 ± 3	15 ± 2
5.0	42 ± 2	49 ± 3	9 ± 2
6.0	41 ± 2	53 ± 2	6 ± 1

Table 1: Fill Placement Accuracy for various allowed distances between rhythms.

Matching Fill Content

To test Hypothesis 2, the rhythms of all fills that were correctly placed in the test for Hypothesis 1 were compared with the rhythms of their corresponding fills in the original drum track. This comparison was made by measuring the Euclidean distance between the rhythms as length 32 integer vectors, in the same way that melody rhythms were compared in the Selection phase of STARR. The generated and original drum fills were declared to “match” if the distance between them was less than or equal to the distance determining the size of the similarity-neighbourhood used in STARR. Table 2 shows the results of this experiment, averaged over all songs and for several different fixed similarity-neighbourhood sizes.

Maximum Distance	% Matched Fill Rhythms
0.0	9 ± 3
1.0	13 ± 3
2.0	17 ± 3
3.0	19 ± 2
4.0	22 ± 2
5.0	40 ± 2
6.0	77 ± 1

Table 2: Matching Fill Content. Shown here are the percentages of correctly placed drum fills whose rhythms matched those of the original drum track within the maximum Euclidean distance given, when STARR was run with a similarity-neighbourhood size determined by that distance.

Improving Accuracy by Moderating Fill Density

As a preliminary step toward testing Hypothesis 3, an experiment was performed to determine whether fill placement accuracy could be improved at all by adjusting the drum fill density of the generated tracks on a per-track basis. It so happens that adjusting the size of the similarity-neighbourhood used by STARR is a direct way to perform this adjustment; larger similarity-neighbourhoods yield more matches for a given bar of melody, and STARR will always place a fill in that bar if the number of matches is greater than zero. Taking advantage of this fact, the data collected in the experiment that tested Hypothesis 1 was re-processed, choosing the distance value that gave the highest fill placement accuracy for each song, as though an oracle had indicated which one to use beforehand. By taking the mean of the maximum fill placement accuracies obtained at each song, an upper bound on the expected accuracy resulting from moderating fill density was obtained: (70 ± 2)%.

To test Hypothesis 3, STARR was modified to calculate and include the energy of each training melody in its database. During the Selection phase, the energy of the input melody was calculated, and the minimum and maximum energies encountered during training were retrieved. These values were then used to compute a distance for STARR’s similarity-neighbourhood calculations by linearly scaling the input melody’s energy into a range of distances,

where the minimum energy mapped to the start of this range, and the maximum energy mapped to the end. The ranges tested all begin at 0.0, and their sizes range from 6.0 to 9.0; these sizes were determined based on an observation of the results obtained while testing Hypothesis 1: the percentage of erroneous additions is extremely small with a distance of 0.0, while the percentage of erroneous omissions is very small with a distance of 6.0. Table 3 shows the results of this experiment for several distance ranges, in terms of the percentage of correct fill placements made, the percentage of erroneous additions, and the percentage of erroneous omissions.

Distance Range	% Correct Placements	% Erroneous Additions	% Erroneous Omissions
[0.0, 6.0]	61 ± 2	4 ± 1	34 ± 2
[0.0, 6.5]	61 ± 2	5 ± 1	34 ± 2
[0.0, 7.0]	60 ± 2	6 ± 1	33 ± 2
[0.0, 7.5]	60 ± 2	7 ± 1	32 ± 2
[0.0, 8.0]	60 ± 2	8 ± 2	32 ± 2
[0.0, 8.5]	60 ± 2	9 ± 2	31 ± 2
[0.0, 9.0]	59 ± 2	10 ± 2	31 ± 2

Table 3: Fill Placement Accuracy with Fill Density moderated by Melody Energy. The distance ranges shown control fill density, with the distance used for a given melody being determined by its energy.

Discussion

In general, the results of this analysis of STARR were disappointing. The following is a discussion of the results of each experiment individually, followed by a set of more general comments.

Fill Placement Accuracy

Hypothesis 1 supports the Primary Hypothesis by stating that the positions of drum fills in a drum track can be inferred from the rhythm of the melody in each bar with high accuracy. Unfortunately, the best fill placement accuracy observed during testing was merely 62%, far below the 80% minimum hoped for in Hypothesis 1. While this implies that the results given in Table 1 fail to support Hypothesis 1, they still support the Primary Hypothesis to some degree, since any accuracy greater than 50% shows at least some benefit to placing drum fills based on the rhythm of the melody (a random assignment of “fill”/“pattern” labels to every bar would yield a placement accuracy of 50%). Additionally, the best accuracy was obtained only for small allowed rhythm distances: 0.0, 1.0, and 2.0, with a trend showing a decrease in accuracy as the distances increased. An examination of the two error rates given offers some explanation for this trend. With small distances, the percentage of erroneous additions is small and the percentage of erroneous omissions is very high; STARR is too conservative when placing fills, rarely adding any by mistake, but missing many of those that are present in the original drum track. With large distances, erroneous additions occur much more frequently than erroneous omissions; STARR is too liberal

when placing fills. Many of the fills from the original drum track are placed correctly, but many extra fills are added by mistake. While one might expect intermediate distance values to affect a balanced tradeoff between these effects, Table 1 shows that this is not the case; as distance increases, the frequency of erroneous fill additions increases more rapidly than the frequency of erroneous fill omissions decreases, leading to a decrease in correct fill placements. One possible cause for this phenomenon is that although increasing the allowed distance between similar rhythms makes STARR more likely to place a drum fill in any given bar, these likelihoods do not increase uniformly over all bars. Considering bars in which fills should be played, as distance increases, bars whose melody rhythms occur only infrequently in the database will remain at zero likelihood for longer than those whose melody rhythms occur more frequently. Simultaneously, the rhythms of melodies in bars that should not contain drum fills are quickly found to be similar to the rhythms of melodies in the database, and fills are placed by mistake. Increasing distance causes STARR to add more fills, but the quality of their placement decreases.

Matching Fill Content

Hypothesis 2 supports the Primary Hypothesis by stating that the rhythmic content of STARR’s correctly placed drum fills would match the content of the corresponding human-crafted drum fills to a similar degree as the rhythms of their melodies matched. The results of the experiment that tested Hypothesis 2, given in Table 2, show an interesting contrast from those in Table 1; in this case, the *largest* distance tested yielded the highest accuracy for matching rhythmic content between drum fills, with 77% of the fills matching in content at a distance of 6.0. However, while a score of 77% is greater than the minimum of 60% required to support Hypothesis 2, Hypothesis 2 claims that the rhythm content match-rate should be greater than 60% for *all* distances tested, and this is clearly not the case; for every distance tested other than 6.0, the percentage of matches was significantly lower than 60%. The results in Table 2 therefore fail to support Hypothesis 2, leaving the Primary Hypothesis without additional support. One plausible explanation for this failure is that the number of melody-rhythm/drum-fill samples in the database may have been too small. For small databases, the likelihood of finding two songs containing closely-matching (with small distance) melody-rhythm/drum-fill pairs is very low. While the Nearest Neighbour search performed by STARR may still find matches for bars of melody rhythms when only small distances are allowed, the likelihood of also having the rhythm of the selected drum fill match that of the fill in the original track within that small distance is could be extremely low. For large allowed distances, STARR would find a larger set of matches for bars of melody rhythms, but those that were detected for small distances would still be chosen as the best matches. The original and STARR-selected drum fills, on the other hand, would now be compared with a larger allowed distance between them, improving the likelihood of them being declared to match. Testing with a larger set of MIDI files would help to verify this claim.

Improving Accuracy by Moderating Fill Density

Hypothesis 3 supports the Primary Hypothesis by stating that the accuracy of STARR's drum fill placement can be improved by using the tempo and rhythm of each input melody to moderate the density of fills in each generated drum track. It is interesting to see that the trend observed in the first experiment also appears to be present in Table 3; the frequency of erroneous fill additions increases as the range of allowed distances increases, while the frequency of erroneous fill omissions decreases at a slower rate. Although this effect is somewhat obscured by experimental error for the distance ranges tested, it appears as though allowing rhythms to differ by large amounts is generally detrimental to the accuracy of drum fill placements. While Hypothesis 3 requires that fill placement accuracy improve, Table 3 shows that this was not the case in the experiment that was run; the best accuracy found with fixed fill density (fixed distance in the first experiment) was $(62 \pm 2)\%$, and every distance range tested yielded accuracies that were equal to this value within experimental error. Although these results fail to support Hypothesis 3, the fact that adding additional melody-rhythm information did not *decrease* placement accuracy is promising to some degree; perhaps converting melody energy ranges to similarity distance ranges non-linearly would yield a better result. The lack of improvement using this method may be due to the fact that many songs are often described as having both energetic and non-energetic sections, and that this information is lost when energy is measured as the number of notes played per second over the entire song. A better approach may be to measure melody energy bar by bar, and increase or decrease the size of STARR's melody-rhythm search neighbourhood based on whether the bar's energy is high or low.

General Discussion

The problem of trading off placing too many fills versus missing fill placements altogether has been prevalent throughout the course of this project, and the similarity-distance parameter of STARR's Nearest Neighbour search has been identified as a means of controlling this tradeoff. Although Hypothesis 3 suggests a means for setting this parameter automatically, in a practical application, it may be most sensible to give control over the similarity-distance parameter to the user, and allow them to obtain the density of drum fills that they desire. For small databases, the quality of fills would suffer when the desired fill density was high, as fills would be retrieved based on very dissimilar melody rhythms. For large databases, however, it should be possible to generate drum tracks with both high fill density and predominantly high quality fills.

Many lessons were learned during the course of this project, with perhaps the most prominent being that seeking to have the drum tracks generated by STARR match those of the original MIDI files is of little use in practice. In practical use, it would be the aesthetic value assigned to the generated tracks by listeners that would determine STARR's success at having learned to appropriately place drum fills throughout a song; unfortunately insufficient time was available to obtain

a statistically significant, unbiased sample of listener opinions concerning STARR's performance.

Future Work

Several items of future work have been suggested throughout this paper, including verifying the accuracy of the drum fill detector used in the Analysis phase of STARR, increasing the number of MIDI files used, converting melody energy ranges to similarity distance ranges non-linearly, and measuring melody energy bar by bar instead of song by song. Other items of future work include investigating STARR's performance with other genres of music (as well as with a mixture of genres), and investigating whether any distance metrics other than those discussed in the Related Work section and Euclidean distance would be worthwhile.

Conclusion

In this paper, I identified the Drum Fill Placement problem, in which a choice must be made concerning both *when* drum fills should be played to accompany a given melody, and *which* drum fills should be played. I surveyed a set of related work, and discussed its potential applications in solving the Drum Fill Placement problem. I then proposed a new approach, titled STARR (Similarity-based Transfers for Automatic Rock Rhythms), which attempts to solve the Drum Fill Placement problem by combining Concatenative Sound Synthesis with a Machine Learning classifier. I stated a set of hypotheses concerning STARR's performance, and although all of the experiments that were run failed to support Hypotheses 1, 2, and 3, STARR's fill placement accuracy of 62% (when compared with original, human-crafted drum tracks) does lend support my Primary Hypothesis, which states that *both the position and rhythmic content of drum fills in a drum track depend on the rhythm of the melody in the bars in which they are placed.*

Acknowledgements

Julian Macdonald is deserving of my many thanks, both for sharing his knowledge of drumming and withstanding the constant buzz of MIDI music pouring from my workstation.

References

- Aucouturier, J.-J., and Pachet, F. 2005a. Jamming with plunderphonics: Interactive concatenative synthesis of music. *Journal of New Music Research*.
- Aucouturier, J.-J., and Pachet, F. 2005b. Ringomatic: A real-time interactive drummer using constraint-satisfaction and drum sound descriptors. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR'05)*.
- MMA. 2001. *The Midi Specification 1.0*. Midi Manufacturers Association.
- Parry, M., and Essa, I. 2003. Rhythmic similarity through elaboration. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR'03)*, 251–252. Johns Hopkins University.

Paulus, J. K., and Klapuri, A. P. 2002. Measuring the similarity of rhythmic patterns. In *Proceedings of the International Conference on Music Information Retrieval (ISMIR'02)*, 150–156.

Paulus, J. K., and Klapuri, A. P. 2003. Conventional and periodic n-grams in the transcription of drum sequences. *IEEE International Conference on Multimedia and Expo (ICME03)* 737–740.

Schwarz, D. 2000. A system for data-driven concatenative sound synthesis. In *Digital Audio Effects (DAFx)*.

Schwarz, D. 2005. Current research in concatenative sound synthesis. In *Proceedings of the International Computer Music Conference (ICMC)*.

Shears, B. 2006. Beatles midi albums. On the Internet: <http://www.geocities.com/SunsetStrip/Studio/7779/>.

Tanguiane, A. S. 1993. *Artificial Perception and Music Recognition*. Springer-Verlag.