Timbral Hauntings: An Interactive System Re-Interpreting the Present in Echoes of the Past

Michael Musick
Music and Audio Research Lab (MARL)
New York University
New York, NY 10012 USA
michael@michaelmusick.com

Tae Hong Park
Music and Audio Research Lab (MARL)
New York University
New York, NY 10012 USA
thp1@nyu.edu

ABSTRACT

Timbral Hauntings (2014) is an interactive installation system created by Michael Musick that considers the impact of echoes from the past on the perception of the present and their capacity to induce future sonic expectancy. This paper discusses details in producing *Timbral Hauntings* including motivation, core concepts, and technical particulars. It specifically discusses the compositional processes using music information retrieval (MIR) and feature extraction techniques to classify phrases and pull information from the 'past' to re-shape the 'present'. Concepts of temporal dynamics will be discussed by examining the compositional process during analysis/feature extraction, classification and re-structuring, and synthesis phases.

1. INTRODUCTION

Timbral Hauntings (2014) is an interactive installation work that borrows ideas from soundscape analysis and the convergence of how "echoes and ethos" reshape the present and future. This paper focuses on the exploitation of feature extraction and automatic sound classification techniques common in the field of music information retrieval (MIR) to the creation of Michael Musick's interactive music system installation *Timbral Hauntings*. Musick has been involved with the composition and performance of *sonic ecosystems* [1] for four years within the *Sonic Spaces Project*. This composition was approached from a desire to address specific problems found in past works from the *Sonic Spaces Project*, specifically, the need for controlled decision making, based on larger collections of data. This is accomplished by applying analysis and organizational techniques common in the research domain of MIR to an ecosystemic-like [2] interactive performance system that builds on past work from the *Sonic Spaces Project*. To accomplish this, modules representing the specific tasks of the system were composed, with considerations of how MIR analysis and classification could benefit these processes.

As is not atypical with many electroacoustic works, this piece comes from a concern for the manipulation of timbre and space within music [3]. This led to an immediate connection to the timbre-based instrument and sound source classification research common in MIR [4]. These tools were examined for their potential use in real-time timbral-based compositions. Ultimately, this led to the development of a system, which analyzes the timbral properties of a physical space (in this case a room in which it is installed), picks the most frequently occurring classification output, then applies these timbral properties to the incoming signals captured by microphones. The processed input signal is then projected into the space, while using classifications results to predict likely future acoustic events.

This paper presents an overview of the system creation, the technical and aesthetic choices that were made, and a discussion of the participant experience.

2. THEORETICAL BACKGROUND AND RELATED WORK

Interactive music systems refer to systems that exhibit changed states in accordance to input data at their interface [5]. These systems are typically thought of as machines that 'listen' to a performer via a microphone or controllers such as digital keyboards and then analyze the incoming signals rhythmic, harmonic, and melodic qualities in order to accompany or follow the human musician. There are numerous examples of systems that exhibit high-level pitch-driven decision-making processes based on user musical input. Prominent examples include George Lewis' *Voyager* [6], John Biles' *GenJam* [7], and Robert Rowe's *Cypher* [5]. Both interactive systems, as well as more general music generating systems are increasingly reliant on MIR-based techniques, including machine learning, to achieve the human-machine interactivity sought by composers, such as in [8]. Even with the high-level decision-making processes that are being incorporated, and the increasing complexity that these types of systems exhibit, the primary interaction for the machine, is that of listening to and reacting to a human performer who inputs data directly into the interface.

Within interactive music systems there are a subset of systems which are composed with the capability of listening to themselves in order to affect their own state [9]. These systems may provide data to the interface themselves, which is then potentially re-introduced back to the system, essentially creating various levels of feedback loops. These can exist as control signals, internal audio signals, and audio signals mitigated via the room through speakers and microphones. This has the potential of creating a complex relationship between all of the various components and agents within an interactive system, whereby changes in any one may cause global system changes. In the case that the interface a physical room where the system is installed, participants and external agents that enter the space, equally become part of this complex relationship between all parts. This interplay between technology, physical space, and external participants/community is one of the defining characteristics of ecosystemic compositions [10]–[12].

This project's focus on the ecosystemtic compositional domain with an emphasis on timbre and interactive performance concepts was significantly influenced by the work of Agostino Di Scipio. Di Scipio's *Audible Ecosystemics* project [5] as well as other works from his later *Modes of Interference* [13] project have been especially influential in part due to the notion of creating music emergent from the complex relationships between all of the agents within these systems. Di Scipio's system compositions are able to achieve this emergent quality by his explicit shift away from "creating wanted sounds via interactive means, towards creating wanted interactions having audible traces" [2]. In a sense, he creates systems where the compositional focus is on the relationships and possible interactions between the various components of the 'ecosystem'. This idea has been an important metaphor in the *Sonic Spaces Project*[1], just as it has had an important role in influencing the work of others including [14].

The goal for the composition of *Timbral Hauntings* and the study around it has been to apply analysis and decision making techniques from MIR to individual agents within the *Sonic Spaces Project*. This project also came from a desire of exploring potential creative application around the data available from and the infrastructure of the *Citygram* project, which can be used to stream soundscape feature vectors through its cyber-physical sensor network. A complete explanation of the *Citygram* project, including goals, technologies, musical applications, and infrastructure can be found in [15]–[18]. Future iterations of *Timbral Hauntings* that leverage the Citygram infrastructure are currently being developed. This will ultimately lead to the inclusion of multiple physical spaces each hosting its own sonic ecosystems, where each system is fully interconnected and reliant upon each other.

3. APPROACH

The basic approach for the development of *Timbral Hauntings* was to identify an appropriate programming environment, the various timbre features that could be used to drive the sonic re-interpretation of the present, a way of classifying and "picking" the feature sets that would accomplish this, and then fine-tuning the system to optimize performance for specific tasks. Although a significant amount of preliminary planning was involved, as the project quickly grew, it diverged away from this original formalization.

[1] For more about the *Sonic Spaces Project* please visit: http://michaelmusick.com/category/the-sonic-spaces-project/

3.1 Program Modules

It was clear from the beginning that this piece would require three major sections.

1. Data Acquisition and Feature Extraction
2. Machine Learning and Classification
3. Sound Processing and Performance

This model closely follows the ideas laid out by Blackwell and Young [19] and built on by Bown et al. [20] in which they propose to work within a *PfQ* modular composition system for the direct development of interactive systems, such as this one, that leverage extensive analysis techniques. In this framework, *P* are the analysis modules, *f* pattern processing (in this case classification and hierarchical organization), and *Q* modules for the sound synthesis (or digital signal processing of sound in this case). *Q,* the final module constitutes the main part of the ecosystem, as it is here that the agents must use the data acquired form *P* and *f* to create a self-regulating, interconnected sonic ecosystem.

Within each of these components, separate processes were developed to handle the specifics of the task. Conceptually speaking, to design these sections, the decision was made to work backwards in order to determine what features and/or control signals were needed to produce the desired interactions for the sounding agents in the final *Q* stage. The next part of this paper discusses the desired interactions in the *Q* section, followed by what was conceived of for *P* and *f* in order to facilitate these interactions. Following this conceptualization, the system modules were built in tandem to ensure the data that was being passed around would work the way it was intended.

3.1.1 Signal Processing - (Q, The Final Stage)

One of the main driving ideas was the creation of a system where interactions of the sounding agents heavily utilized the information of the past to reshape the present and predict the future. The resulting music that emerges from the system is then always directly tied to the echoes, events, and happenings of the past. The qualities of the past are thereby embedded in the nature of agents as they are created within the system to interact with the sonic energy present in the room.

The major challenges for this problem included: (1) determining how to extract meaningful features that would allow the system to choose a frequently occurring past event and (2) how to embed these features into agents to transform the current audio events in the space. The transformations that were explored included filtering and shifting of current "live" sonic events through the application of past sonic event characteristics represented by extracted features. This project also worked to explore ways the system could try and predict the next sounding events in the space by utilizing frequently occurring phrases that would be the 'same' sonic event likely produced in the future.

In addition to testing and prototyping the features that could be used as timbral re-interpreters, a need to obtain control signals that could be used for the further re-interpreting of current audio events was also found. Specifically, determining a control signal that could be used to alter the temporal speed and direction of the timbre events. Finally, from the identified interaction goals it was known that a control signal needed to be found that could be used to efficiently evaluate moments of the present against near-matches of the past.

3.1.2 Analysis and Feature Extraction - (P)

After determining a need for both timbral and gestural features for the synthesis stage, the project focused on outlining possible ways of extracting this data. The problems that had to be solved in the analysis section included: (1) finding suitable features to describe the timbral and gestural properties of audio events and (2) implementing ways to extract and store those features in a real-time process for later use. Ultimately, as will be discussed below, it was determined that the identification of timbre classes and the application of those timbre classes' characteristics through a filter would need to be handled by separate signal processing techniques.

It became clear that a secondary decision-based feature set, describing the past, was also needed to provide further control variables for agents in the *Q* module. The decision to pursue the analysis of gesture came from research in the area of music theory that uses common gestural motifs to help analyze moments in classical music compositions [21]. This was the inspiration for trying to represent and classify frequently occurring gestures.

3.1.3 Decision Making and Classification - (f)

After determining the features that would yield the necessary control signals and feature sets for the *Q* stage, a final step to consider was to explore an appropriate algorithm for classification that could choose which feature sets to pass to agents at their time of creation in the *Q* stage. Multiple means of machine learning approaches were thus considered. However, the aesthetic goals of this piece were to create a system, which could be used for public installations or with improvising ensembles. The nature of the typical sonic events of either would be difficult to "know" prior to performance. Additionally, this piece is concerned with allowing for any events in a space to be considered as a potentially valid historical sonic event that could affect future agents and interactions. This eliminates most techniques, because supervised machine learning algorithms require labeled training data prior to deployment. Therefore it seemed obvious that unsupervised, real-time learning techniques would be more appropriate for providing this flexibility.

3.2 Environment

The *Sonic Spaces Project,* and as such, *Timbral Hauntings* are live systems that are intended to be installed in spaces where external agents and participants can interact with, and listen to the compositions as they emerge and fill a room. Much of the work involving MIR has utilized offline approaches to analysis, training, and modeling using tools such as MATLAB and Python. Unfortunately, it is difficult to do real-time audio in MATLAB, which has been a standard for MIR research [22], especially when using it on the OS X platform. Although the MATLAB environment was used for early prototyping of feature extraction techniques and composition of potential interactions, this composition was moved to SuperCollider in order to facilitate real-time interaction. This programming environment allows for real-time audio processing, and is well equipped to handle the types of analysis and processing that were under consideration for the project.

Much of the *Sonic Spaces Project's* work, including system implementation has been accomplished in the SuperCollider environment. SuperCollider allows for considerable amounts of flexibility in coding practice and it is a very efficient audio processing and synthesis environment. For custom algorithms that are unavailable in SuperCollider, it is not exceedingly difficult to implement in native C++. It is a proven client for using machine listening applications and includes a growing base of researchers writing third party unit generators (UGens) and also using SuperCollider's analysis and MIR capabilities for composition purposes [23].

Ultimately, the final implementation in SuperCollider utilized external machine listening and MIR libraries. However, the development and testing process included working between MATLAB and SuperCollider in order to continue prototyping interactions for each stage of the system and to insure complete understanding of how each analysis algorithm was being put to use.

3.3 Deployment Tweaking and Testing

The plan for this system was to build each module up piece by piece, ensuring that communication between the modules was considered throughout the development process. The system was to be built by verifying ideas and techniques through a combination of work in MATLAB and SuperCollider. This way, implementations of MIR specific techniques could be explored to ensure understanding of the expected outcomes before using the equivalent functions in SuperCollider.

4. APPLIED TO A SYSTEM

This section discusses the implementation of the system and more fully explores the development flow of the composition. It also gives specifics as to how the system works, and how compositional choices were made. Note that even though the system was not developed sequentially from start to finish but was rather conceptualized backwards, the presentation of various components will be described and detailed here in the order that it is represented within the computer code.

4.1 Initial Considerations

In order to consider the entire physical space as the interface for a system, it is important to provide a sufficient number of audio inputs to cover the space. This is especially important when trying to capture the spatial timbral characteristics. The compositional focus in this project was not on instrument-specific timbre but rather on the collective timbre of physical spaces. For this reason, except during stage *Q*, all feature extraction stages take a mix of an array of microphones covering the space. During the initial stages of development as well as early presentations of the work, all of the spaces where the *Timbral Hauntings* system has been installed have been relatively confined; therefore two microphones are currently used. The number of microphones is flexible, and for installation of the system in larger spaces, more microphones can easily be incorporated.

One key aspect of MIR is to develop and select appropriate low-level acoustic descriptors as a way of improving results – a classic garbage-in-garbage-out (GIGO) system. This concept has been applied at every stage of the signal processing chain and is one of the reasons for utilizing high-quality hardware and transducers. Of the highest importance is the use of high-quality microphones, at least in regards to the *Sonic Spaces Project*. Consequently feature sets, when used for analysis or other digital signal processing applications, tend to produce more robust results when using these high-quality input signals.

4.2 Development of the Timbral Analysis and Classification System

The system starts by taking in the mixed audio signal from the room's microphone array. This signal is then routed to various components that require audio input for analysis. In order to classify the room's various timbre events, Mel-Frequency Cepstrum Coefficients (MFCC's) [24] were used. MFCC's have found wide use for automatic musical instrument recognition and have been found to robustly reflect timbre in general. For this project, a DFT with a hann window size of 1024 samples, hop size of 50%, and sampling rate of 44.1kHz proved to provide acceptable results when prototyping the system in MATLAB. These parameters did not change after evaluating and testing other parametric values in the final SuperCollider implementation. In SuperCollider, the frames from the DFT are passed to the MFCC *UGen*. For the current implementation, it has been found that using 13 MFCCs resulted in efficient classification results. The MFCC UGen returns a control rate signal, which is then passed through a one-second long moving averaging filter, which serves to remove high-frequency irregularities caused and creates a feature signal that appropriately describes the room's timbral characteristics. This resulting control rate signal from this MFCC *SynthDef* is then passed to the classification *SynthDef*.

In order to do classification of salient acoustic events, a frequency-domain based onset detector was used for acoustic event detection [25]. This onset detector allows for the filtering of novel acoustic events, which are then passed through, a frequency-bin based whitening process [26] before threshold-based triggering occurs. For the type of input expected in this system, a weighted phase detection algorithm worked well in tracking both changes in amplitude, pitch, and novel sound sources. The trigger threshold was also assigned to an adaptive process, which scaled the threshold down, over the course of 16 seconds, after a new timbre classification occurs. (This is a user tunable parameter, and different values work better in different spaces.) This was found to limit re-triggering of a single sonic event.

When an event is identified from the onset detection UGen, a trigger is passed to the classifier UGen causing it to analyze the extracted features. As mentioned above, an unsupervised learning algorithm was determined to provide the desired flexibility for this project. For this reason, a real-time k-means clustering algorithm was implemented for the classifier [27]. This classifier works by re-computing the *k centroids* every time a new acoustic event is detected. It was found that determining the optimal number of *k* to use is still an active area of research lacking common practices [28]. For that reason, multiple *k* values were auditioned. Currently, six seems to provide a suitable solution, although this too is a user adjustable parameter where varying values may produce more appropriate results depending on the characteristics of the installation space. Originally, the number for centroids was determined and equal to the number of expected timbres. However, it was found that choosing a number slightly larger than the expected number of timbres results in better accuracy for the system.

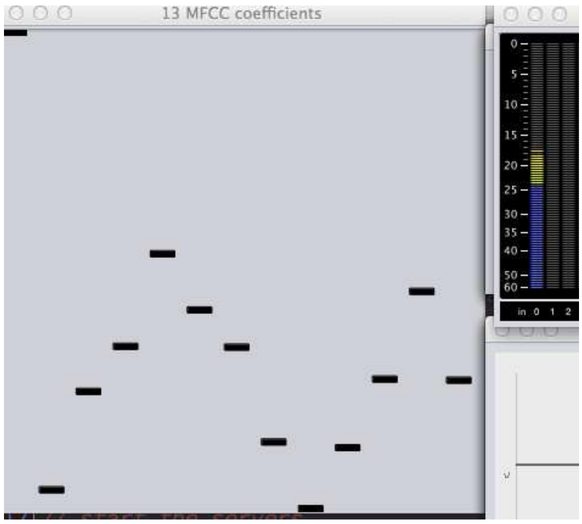

Figure 1. Relative MFCC Coefficient output for a whistle. These values are passed to the K-Means classifier as an array of floats.

Other options considered during initial prototyping in MATLAB, included; Linear Predicative Coding (LPC) coefficients, and single value spectral measures, such as spectral centroid and spectral flatness. MFCCs seemed to provide a large enough feature set to create acceptable results with minimum samples in the k-means algorithm. Figure 1, shows a typical k-means representation of a whistle; these 13 coefficients are then passed to the classifier as an array of floating point numbers. Alternative machine learning approaches have also been considered. However, as mentioned earlier, a willingness to accept all possible sonic events is important to the theoretical goals of this piece. Therefore, machine-learning techniques that require pre-labeled, supervised training data were not an option.

4.3 Development of Gesture Feature Extraction

During prototyping, the resulting audio from the *Q* module did exhibit convincing characteristics of the analyzed audio. However, the resulting sonic material was too similar to the original acoustic events, and more abstraction in the composed interactions seemed necessary. For this reason, it was decided to explore the use of additional control signals that could be embedded within each of the agents, along with the timbre characteristic descriptors in the *Q* module. This focused on the identification of temporal gesture features extracted from audio as a way of further transforming the end result. Fundamental frequency estimations from the identified timbre class were originally considered. However, it was decided that this parameter should be a separate descriptor of past spatial sonic events that are not necessarily the same as the timbre classifications. This provided a decoupling from the timbre features used, as they were being applied to the synthesis module, resulting in a more interesting interaction on the part of the agents.

The motivation for exploring gestures came from a music theory study on using gestures of pitches to describe common anticipations in classical era compositions [21]. This led to another discussion in the same source, which examined the mechanics of gesture as they relate to human physical motion and how motion changes gradually, "ramping in and out of the motion." To track changes in musical gesture, the system computes the first derivate of an autocorrelation-based fundamental frequency estimator. The pitch detector was limited to frequencies between 80 and 3 kHz and also requires a minimum amplitude threshold of 0.01 Root-Mean-Squared (RMS) as well as a minimum peak threshold, which helps eliminate octave errors.

To normalize these features to a linear scale they are converted to MIDI note numbers, which in SuperCollider are represented as floating point numbers, and not limited to integer representations. These values are then passed through a moving average filter of 512 samples. Finally, the slope between these floating-point MIDI values is measured. Figure 2 shows an output signal captured by this process. These events are recorded, and after the classifier chooses a class, the most recent event, which matches the most frequently occurring class, is passed to the *Q* module where it is embedded in the newest agent as a control signal.

In order to classify these events, it was necessary to reduce feature dimensions to a manageable size to an approximately 6-18 feature size range, which matches the size range used for MFCC classification. This is accomplished by passing the signal through a curve-fitting algorithm and using the computed coefficients (except for the intercept value) as the features to describe an acoustic event. No such solution exists in SuperCollider at this moment, and attempts to create a non-optimized process through SuperCollider's client language severely slowed down system performance. A solution that was developed entailed sending gesture signals to Python via a Unix terminal process. This allows access to Python's *Numpy polyfit* method, and the unloading of processing from SuperCollider: after Python has computed the result, the

Figure 2. A 3" gesture generated from rotating tingsha bells.

coefficients are returned to SuperCollider. The output of the Python process is shown in Figure 3. These features are then passed to a separate classifier, which classifies and eventually chooses a gesture class.

4.4 Choosing A Class

The classifiers initially need to be self-trained with a user-defined number of samples. Once a training count

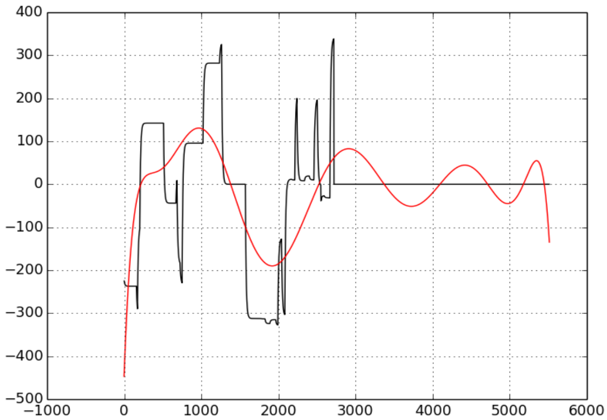

Figure 3. Visualizing a 10-degree polynomial best-fit curve produced by Python for a gesture event.

threshold is passed, the classifiers outputs a class based on the computed k-centroid values for each feature sample passed in. The classifiers track the most frequently occurring feature classification by employing a simple array count, equating to an indexed slot for each potential class ID. Once the classifiers registers a critical number of samples, they choose the most frequently occurring class by picking the index of each array with the highest count.

Once a "most frequent" class is chosen, the classifiers reset their counts to 0. However, the system keeps their historical training data for future classification decisions. This allows for changes in the timbral events that may occur in the installation space while considering historical data as an ever-present factor, and takes into consideration the system's own output to the space as a potential source of influence.

When a class is chosen the system selects a recent, corresponding audio recording for the timbre feature set and a control rate recording for the gesture feature set. These recordings are then transferred from short-term storage over into new, local-private buffers, which are then handed over to the *Q* module, where they are assigned to a newly created agent, along with pointers to the feature extraction control signals from *P*. The short-term class

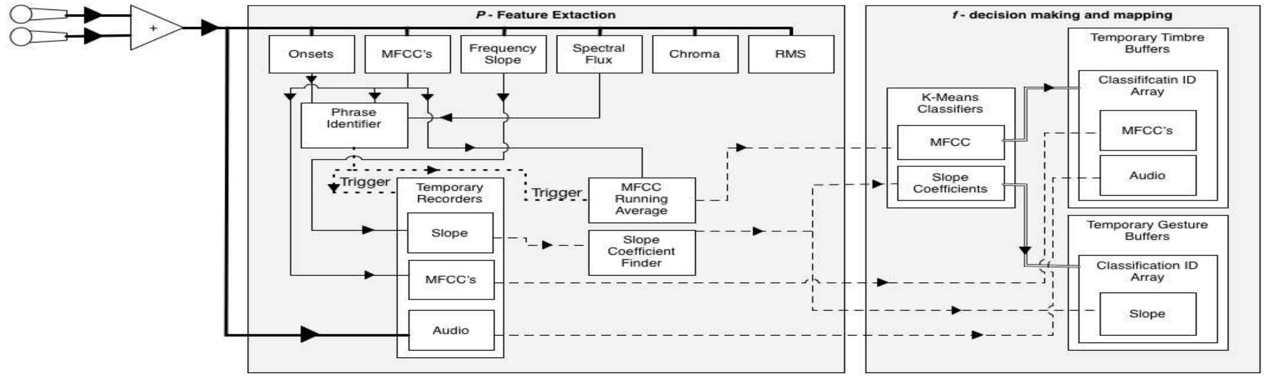

Figure 4. *P* and *f* flow diagram. After the classifiers make a decision, they send the candidate buffers to the new agent in *Q,* along with the feature extraction control signals.

audio and control recording buffers are also reset at this point, along with the entirety of the classification module. Figure 4 shows an overview of the *P* and *f* process, including the feature extraction, temporary recording buffers, and classifiers.

4.5 *Q* Module

The *Q* module is where new agents are created and also embedded with "characteristics" of the past. These agents are embedded with the identified "most-frequent" feature sets, which are used to:

1. Process current live signals occurring in the space.
2. Identify likely "future" events for playback, effectively predicting the "future."

The MFCC's were originally going to be used as a way of obtaining filter coefficients for the re-interpretation of live audio by agents. However, the Linear Predictive Coding (LPC) technique was far more suited for this task. The LPC algorithm effectively produces filter coefficients that can be used for resonant filters [29], which reflect the a signal's spectrum. Typical applications of this technique have been to separate a signal's spectrum from its excitation function, which offers a composer the ability to change parameters of the signal independently and then resynthesize the altered signal (e.g. Paul Lansky's *Idle Chatter*). For the purposes of this system, the LPC technique is used solely as a way of analyzing past spectra and shaping the spectra of new audio events.

Full audio recordings are captured and stored for potential use in this module during the classification process. Recordings can be up to 16-seconds in length if no other event triggers first. This 16-second long recording is then truncated to remove any silence, and only the portion of the signal containing active audio is passed to the agents in the *Q* module. This signal is then run through the LPC analyzer/filter (from this point forward, this signal will be referred to as the *analysis signal*).

By storing complete audio signals, the *Q* module can optimally process them prior to routing through the LPC UGen. This module does a number of processes in order to increase the "richness" of interactions. Chief among them is to alter the playback speed of the analysis signal. This is accomplished with the gesture feature that is also passed to the module. The gesture signal is slowed down by 25% of the original speed and then used as a scalar for the playback speed argument of the analysis signal. This gives the playback speed argument speed profiles, which have been derived from the historical events of the installation space. The playback speed is normalized to a floating-point number between $\pm$ 2.0 times the original speed, with negative values causing a reverse playback to occur.

The analysis signal is then run through a low pass filter with a cutoff of 8 kHz. Traditionally, LPC synthesis has been found to produce optimal acoustic results at lower sampling rates as it was developed to model speech spectra. Unfortunately, this system requires a sampling rate that is 44.1 kHz or higher. Dampening high frequency components has been used as form of compromise to achieve effective results. Subjectively, during prototyping, this process did allow for clearer results to occur when using the analysis signal to drive impulse trains running through the filter.

Next, the analysis signal is passed through a pitch shifting UGen, which uses a synchronous granular synthesis algorithm. The shift factor for this UGen is inversely proportional to the timescale of the playback, compensating for *some* of the pitch shifting effects caused during that process. However, at the outer edges this scaling does breakdown, which serves as an effect in itself.

Finally, the analysis signal's values are extracted before compression and normalization occur. When the uncompressed signal is above a tunable RMS threshold, the compressed signal is allowed to pass, otherwise it is gated. This allows for a stronger signal to be passed to the LPC module while suppressing background noise.

This is the final form of the analysis signal before it is routed to the LPC UGen. The LPC UGen then computes 128 poles, operating with a 512 sample, double buffer to handle the changing of filter coefficients. The input signal that is passed through the filter is delayed by 2 seconds from the original input in the room. This signal does not go through a true whitening process, as is suggested for LPC impulse signals, but is subjected to multiple stages of resonating comb filters and an all-pole reverb algorithm, which serve to broaden the spectrum with frequencies from the signal itself. When passed through the LPC filter, this signal is given characteristics of previous "frequent" occurring sonic events that have occurred in the system.

In addition to the “re-shaping” of present material in the historical sounds of the past, this system also tries to predict the future through the examination of the present against the past. During the analysis and recording process, the time-stamped MFCC values are stored in a separate signal buffer at a granularity of 20ms. When the candidate timbre class is chosen, the classifier takes this 14-channel signal along with the audio signal. This signal is then passed to a function that transforms it into a KD-Tree [30], where timestamps serve as the identifier values for each node of the tree. This tree structure is then embedded during the creation of new agents in the *Q* module. The agent analyzes the ‘present’ signal, as it looks for nearest-neighbor matches with a tunable threshold of the current MFCC values in relation to the KD-Tree. If a match within a certain threshold is found, then the time-stamp key is used to grab grains of sounds from the stored audio signal that are 20ms ahead of the time-stamp. These grains are then played back through the system, where pitch, position and playback speed are a function of the distance between the two MFCC sets.

This interaction is particularly interesting, as it reliably matches current moments in the room to ‘candidate’ signals in the agent’s tree. This ‘playback’ of what the system has decided is the future is usually also a close enough match as to confuse the participants’ perception as to what sounds occurred from agents in the room, or were created from the system. This interaction also has a tendency to ‘re-trigger’ itself, creating an interconnection between components, which ultimately adds to a sense of progression in the piece, as it creates rhythmic moments from grains of sounds that slowly fade away as the total composition changes.

In addition to the two main interactions that each agent is created with (transformation of the present and prediction of the future), they are also passed pointers to the internal control busses with the feature extraction signals from *P*. These signals include; a 12-tet chromagram that is sampled at every onset and is the average of the last identified phrase, spectral flux, and RMS. These are used as further control over the composed interactions in the *Q* module. Live control signals also allow the agents in *Q* to monitor sonic saturation of the space. When this occurs, the agents either die off, or take measures to reduce their sonic output to protect the balance of the ecosystem.

These processes described in this section are recreated in each agent for each new class chosen by the classifier. The number of candidates that can be playing at any one time is currently set to 4, but this is a user adjustable value. As new agents are created, old ones die out.

5. SUMMARY

This work asks participants in the installation space to consider the impact and influence that moments in history can have on the present and future. The echoes of the past are always with us, and *Timbral Hauntings* works to exploit this idea by building a memory of these events and choosing the most frequently occurring ones to directly change the course of the future. In this case, hauntings of these memories are embedded in the ‘nature’ of agents as they are created in the system. The system is programmed so that only a small group of agents can survive at any given time. This means that as new agents come into being, older ones die out. The contributed sonic events are what is left of them to haunt the system.

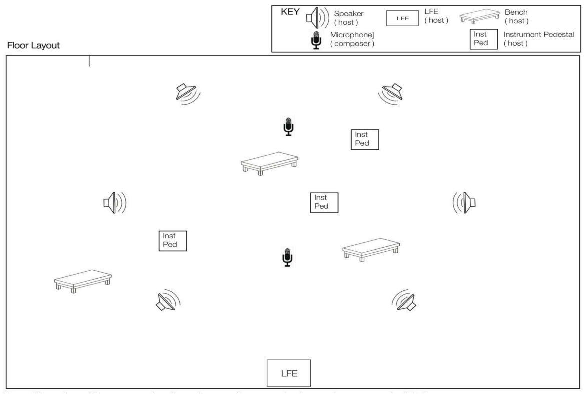

Figure 5. Possible floor plan for *Timbral Hauntings*

This piece is best suited in spaces that reinforce the notion that ‘we are always with the past.’ It is best if relics, mementos, and history can exist in the space of the system: affecting the sonic characteristics of the room, and reminding participants of the people and events that have come before them. Spaces that have their own unique sonic characteristics are also preferred. They impart their resonant frequencies, room nodes, and reverberant qualities into the composition itself. With the room as the interface, a strong interrelationship occurs between system, participants, and space. As can be seen in Figure 5, the piece is intended to offer room for participants to simply sit, ponder, and listen, as the music that emerges changes in response to the sonic events occurring in the space. Participants are also encouraged to influence the future of the system by contributing sonic energy to the space. They are free to use any of the instruments supplied around the room, or to make sound in anyway that they are inspired to. Regardless of what they do, their presence in the space effects the future, whether it is directly contributing sonic material, or simply existing in the space, allowing their mass to change the sonic characteristics of the room, and disturb potential nodes. This piece maintains strong relationships between the sonic events of the space; whether created from digital agents or external physical agents, the room itself, the past, and the present. All components rely on each other to create the final music that emerges during a performance of the ecosystemic composition and installation, *Timbral Hauntings*[2].

6. REFERENCES

[1] J. McCormack and O. Bown, “Life’s What You Make: Niche Construction and Evolutionary Art,” *Appl. Evol. Comput. EvoWorkshops 2009. Lect. Notes Comput. Sci. 5484*, pp. 528–537, 2009.

[2] A. Di Scipio, “‘Sound is the interface’: from interactive to ecosystemic signal processing,” *Organised Sound*, vol. 8, no. 03, pp. 269–277, Apr. 2003.

[2] Those wishing to view code or hear iterations of this work may visit: http://michaelmusick.com/tag/timbral-hauntings/

[3] E. Hoffman, "On Performing Electroacoustic Musics: a non-idiomatic case study for Adorno's theory of musical reproduction," *Organised Sound*, vol. 18, no. 01, pp. 60–70, Mar. 2013.

[4] M. Müller, "Computational Audio Analysis," *Rep. from Dagstuhl Semin. 13451*, vol. 3, no. 11, pp. 1–28, 2013.

[5] R. Rowe, *Interactive Music Systems: Machine Listening and Composing*. The MIT Press, 1992.

[6] G. E. Lewis, "Too Many Notes: Computers, Complexity and Culture in Voyager," *Leonardo Music J.*, vol. 10, pp. 33–39, Dec. 2000.

[7] J. Biles, "GenJam: A genetic algorithm for generating jazz solos," in *Proceedings of the International Computer Music Conference (ICMC)*, 1994.

[8] J. P. Forsyth and J. P. Bello, "Generating Musical Accompaniment Using Finite State Transducers," in *16th International Conference on Digital Audio Effects (DAFx-13)*, 2013.

[9] R. Meric and M. Solomos, "Audible Ecosystems and emergent sound structures in Di Scipio's music. Music philosophy enhances musical analysis," *J. Interdiscip. Music Stud.*, vol. 3, no. 1, pp. 57–76, 2009.

[10] A. Di Scipio, "Untitled Public Lecture Sound Installation as Ecosystemic Construction," in *SOUND – SYSTEM – THEORY: Agostino Di Scipio's Work Between Composition and Sound Installation.*, 2011.

[11] R. Meric and M. Solomos, "Analysing Audible Ecosystems and Emergent Sound Structures in Di Scipio's Music," *Contemp. Music Rev.*, vol. 33, no. 1, pp. 4–17, May 2014.

[12] M. Musick, "Examining the Analysis of Dynamical Sonic Ecosystems in Light of Brown's Analysis of Theories," in *International Computer Music Conference Proceedings (ICMC)*, 2014.

[13] A. Di Scipio, "A Constructivist Gesture of Deconstruction. Sound as a Cognitive Medium," *Contemp. Music Rev.*, vol. 33, no. 1, pp. 87–102, May 2014.

[14] O. Green, "Audible Ecosystemics as Artefactual Assemblages: Thoughts on Making and Knowing Prompted by Practical Investigation of Di Scipio's Work," *Contemp. Music Rev.*, vol. 33, no. 1, pp. 59–70, May 2014.

[15] T. H. Park, M. Musick, J. Turner, C. Mydlarz, J. You, J. H. Lee, and C. Jacoby, "Citygram One: One Year Later," in *International Computer Music Conference Proceedings (ICMC)*, 2014.

[16] T. H. Park, J. Turner, J. You, J. H. Lee, and M. Musick, "Towards Soundscape Information Retrieval (SIR)," in *International Computer Music Conference Proceedings (ICMC)*, 2014.

[17] T. H. Park, J. Turner, M. Musick, J. H. Lee, C. Jacoby, C. Mydlarz, and J. Salamon, "Sensing Urban Soundscapes," in *Mining Urban Data (MUD) Workshop Proceedings of the EDBT/ICDT 2014 Joint Conference*, 2014.

[18] M. Musick, J. Turner, and T. H. Park, "Interactive Auditory Display of Urban Spatio-Acoustics," in *The 20th International Conference on Auditory Display (ICAD-2014)*, 2014.

[19] T. Blackwell and M. Young, "Self-organised music," *Organised Sound*, vol. 9, no. 02, pp. 123–136, Oct. 2004.

[20] O. Bown, "Experiments in Modular Design for the Creative Composition of Live Algorithms," *Comput. Music J.*, vol. 35, no. 3, pp. 73–85, Sep. 2011.

[21] A. Gritten and E. King, *Music and gesture*. Ashgate Publishing, Ltd., 2006, p. 232.

[22] T. Park, *Introduction to digital signal processing: computer musically speaking*. Singapore: World Scientific Pub Co Inc, 2010.

[23] S. Wilson, D. Cottle, and N. Collins, Eds., *The SuperCollider Book*. The MIT Press, 2011, p. 776.

[24] J. Aucouturier and F. Pachet, "Tools and Architecture for the Evaluation of Similarity Measures: Case Study of Timbre Similarity.," *ISMIR*, 2004.

[25] N. Collins, "A comparison of sound onset detection algorithms with emphasis on psychoacoustically motivated detection functions," in *Audio Engineering Society Convention 118*, 2005.

[26] D. Stowell, M. Plumbley, and Q. Mary, "Adaptive whitening for improved real-time audio onset detection," in *Proceedings of the International Computer Music Conference (ICMC)*, 2007.

[27] B. McFee, "More like this: machine learning approaches to music similarity," University of California, San Diego, 2012.

[28] M. Chiang and B. Mirkin, "Intelligent choice of the number of clusters in K-Means clustering: an experimental study with different cluster spreads," *J. Classif.*, vol. 27, 2010.

[29] C. Roads, J. Strawn, C. Abbott, J. Gordon, and P. Greenspun, *The computer music tutorial*. MIT Press, 1996.

[30] I. Witten and E. Frank, *Data Mining Practical Machine Learning Tools and Techniques*. San Francisco, CA: Morgan Kaufmann Publishers, 2005.