INSCRIBING SOUND FORMS (DEMO)

Benjamin O'Brien
PhD, University of Florida
Marseille, France
http://benjamin-obrien.com

RÉSUMÉ

Composers interested in transferring sound forms can use their knowledge of sounds and contexts to guide the process of selecting and pairing together sound with inscriptional contexts. Depending on the difference between the two contexts, a cognitive dissonance can, in theory, emerge between the contextual associations of the transferred form and the context or domain into which it is transferred. If listeners are to be challenged by the effect of inscribing a sound's form into a new or different context, then it is necessary to choose sounds and contexts, into which to inscribe them, that are recognisable to listeners, despite the emergence of any cognitive dissonances. This paper discusses my development of a suite of computer programs used to produce materials that create the illusion of one sound's form as inscribed into another.

1. INTRODUCTION

Words such as *map* or *translation* are often used to help explain the processes developed by electroacoustic and computer music composers. One might say that a composer, who is interested in mapping sound, makes decisions about what qualities distinguish a sound and try to transfer them into a different context of sounds. However, if a composer wants to make this mapping clear to listeners, then the goal is more closely associated with translation - one sound's form has been translated onto and carried by another sound. While mapmaking and translating are similar, their differences are distinct, and both are problematic when discussed in the domain of music.

2. A FEW WORDS ON MAPS & TRANSLATIONS

2.1. Distinguishing Maps from Translations

From the outset, the goal of a language translator is to create an equivalent form of the sender's message, such that the translation is capable of conveying the same message to a receiver [1]. On the other hand, a mapmaker only aspires to make a map, which can assist in navigating or understanding, in a particular way, the territory being mapped [2]. So by design, maps are intrinsically linked to that which they map, whereas translations, which transfer the form of a message from one domain to another, work to prevent any dependence on the source or sender. Clearly the goals of both processes deal with the re-presentation of something, which underscores a need for clarification.

Fortunately, the blurriness in distinguishing mapmaking from translation becomes a bit clearer when discussing the two in the context of language. As previously discussed, one map can serve as a representation of a given territory. However, if a mapmaker produces many maps of the same territory from a different view or interpretation, she creates a language : Each map communicates a particular perspective of the territory through what it maps. While a single map presents a singular view of a territory, language, by design, presents a multitude of mappings that become codes whose use and reconfiguration effectively turn the territory into a domain, place, object, or entity out of which different forms and ideas about form may be drawn.

That being said, it is important to remember that translation in language seeks to find equivalencies for readers who are not familiar with the source language, and therefore need an equivalent form. Equivalence is sought but is understood as problematic for anyone who knows both languages - hence the illusion of transparency [3]. If one reformulates the definition of language in terms of mappings, then one might say that translation employs multiple maps, where each map points to a different meaning representing a particular view or interpretation of the territory. This collection of maps re-conceptualises the domain of language as a medium for carrying, transmitting, or conveying different forms. With this understanding of language in mind, translation in linguistics can be seen as the process of creating a map (a message in the receiver's language) using the mappings available to a language that can serve in place of another map (the sender's message). Thus, languages connected by translation are different maps that, through their linkage, seemingly map the same ideas or territory.

2.2. Elevating Maps to Languages

But there is a difference between creating multiple maps in order to elevate them to a language for communication and selecting existing maps (words) in a language for the purpose of communicating equivalent forms. Languages are, for the cultures in which they are used, schemes for designating, representing, and speaking about the language's culture [4]. In a sense they are like map collections for a particular culture and how communication is handled within it'a language of many maps for the things or territory a culture needs to represent. With that model in mind, when one seeks to translate, they assume, correctly or not, that what one language addresses is presumably the same as another language ; it is that the maps they independently make address the same ideas or territory.

One could say that translation is a sort of elevated form of mapping. Framed as such, when equivalence of language is sought, subsequent problems emerge because either the things each language represent are actually different or the culture's interpretation of the thing is different. So, languages are possibly like independently created collections of mapping designations, and translation is the attempt to claim that the maps or map collections made by each culture designate the same things or same interpretations of things, when in fact they may not.

In summary, translation seeks to find equivalencies between culture-specific ways of mapping - between maps and mapping languages - on the assumption that cultures map the same things and in the same way. That mapping distinguishes difference is simply a product of its role in distinguishing choice, which is part of language formation. To map is to distinguish the role of choice, whereas to translate is to (seek to) distinguish the absence of choice by finding or forming equivalencies. That being said, someone could choose to only map, both in the formation of a language and in its transfer of ideas from one language to another. If, however, they seek to translate, then they assume that there is equivalence.

3. FORM & MEDIA

If a listener is to identify one sound's form as transferred by another, she must know both the source and target sound domains, and know them in the same analytical and adaptive way the composer does. These source and target sound domains may be described in terms of two types of sound components : the sound or sounds from which a *form* [1] is abstracted and the carrying sounds or *medium* [2] into which the form is transferred. So for the listener to be a complicit translator, she must be capable of imagining the sound or sounds that were used to create a form in the source domain, while hearing the sound medium that has been reconfigured to carry the form in the target domain. Thus, the composer of musical translations uses an abstracted sound form to reconfigure a sound medium over time so that complicit listeners may interpret the medium as capable of not only carrying the inscribed form, but also projecting its most identifying features.

Of course there is nothing about sound that implicitly allows abstraction more than anything else. Given its elusive nature, listeners may be more inclined to interpret sound : Outside of spoken language, sound is not well defined in terms of its meanings and use relative to those meanings. What someone hears is no more translatable than what they see, aside from the fact that their memory of sound, which is presented through time, passes as time passes, and must be repeated to be re-experienced, whereas what one sees, if it is not changing or moving, remains more constant. So the listener who functions like a complicit translator knows the conceit being presented as an imagined kind of equivalency, mitigated by matters of fidelity and transparency to the source and target domains, respectfully.

In exploring the composer's motivation to preserve the characteristics of the form over the medium, or the medium over the form, it is useful to consider the contexts that frame each. What factors lead a composer to favour the domain of the source sound over the target domain (or vice versa) ? If listeners perceive and interpret a sound as a musical translation, do they do so the same way as the composer, who may show greater faith to one sound domain than the other ? Another thought to consider is how switching the source and target sound domains could affect the process of creating a musical translation. To do so, one would need to find a form to transfer and a medium in which to inscribe where previously there was neither. Would this domain-switch alter or possibly invert a listener's interpretation of the musical translation as being more faithful to either the source sound domain or the target sound domain ?

4. COMPOSITIONAL MODELS

4.1. Messiaen's *Réveil des oiseaux* (1953)

To begin investigating these considerations, one might consider a passage in Olivier Messiaen's *Réveil des oiseaux* (1953), where the composer chooses the xylophone represent a wren's song in mm 35-38 (Figure 1). In terms of forms and media, the form or pattern of the wren's song as performed or carried by the medium of xylophone notes. The presentation of the abstracted wren using xylophone tones succeeds, despite the fact that the xylophone is *not* a wren. The wren is what continues through representation of its form, and the xylophone, as the medium of presentation, is what changes, from wren sounds to xylophone sounds. It is the tones of the xylophone that are composed or configured to reproduce the distinctive contour or form that listeners identify as the wren's song. By doing so, Messiaen captures and reproduces, using a xylophone, the form identified with the wren's song in such a way that it (briefly) overcomes the dissonance of context created by playing the song on the xylophone.

With the above example in mind, how might one create an inverse example, where the source and target sound domains switch, where a sequence of bird sounds were composed to carry the form of an excerpt of xylophone music. Given this arrangement of source and target sound domains, what are some of the problems of interpretation that might emerge ? In comparison, why might the translation Messiaen composed - the xylophone playing the

1 . Form is the distinguishing features of a sound sequenced over time.

2 . Sound medium is the collection of elements, and possibly their distributions, without reference to any one ordering in time

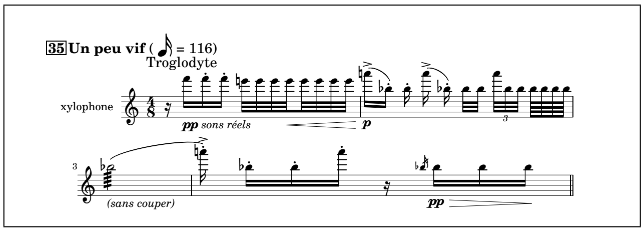

Figure 1. Wren (*Troglodyte*) from Olivier Messiaen's "Réveil des oiseaux" (mm. 35- 38)

wren's song - be more readily understood by the complicit listener ?

The underlying point being raised here is whether there are, within the domain of bird sounds, those sounds that are equivalent to xylophone notes, sounds that, as a medium, can be configured to play the xylophone melody. How can a bird "play" a xylophone melody ? At first glance, this question does not make sense, as there is no such thing as a bird medium in which bird songs have seemingly been written. To create a bird song medium capable of configuration on the terms that xylophone melodies are prescribed might well mean a loss of all that distinguishes bird songs. When a sound is complex, like a bird sound, it has distinct features, which may make the sound less equipped to carry other forms or act as a sound medium. So the invention of any medium through the analysis of any situated form necessarily leads to the loss of that which ultimately distinguishes the form - the search for medium leads to the loss of form, as well the synthesis of form necessarily leads to the loss of medium.

At the heart of this exercise is the problem of finding or inventing a medium of bird sounds into which xylophone melodies might be convincingly inscribed. The goal of course is to order the bird sounds in such a way that they not only carry the forms abstracted from the xylophone passage, but also preserve the features that make them complex. The problem is that bird sounds are not like xylophones or other musical instruments, whose sounds are ordered into twelve-tone equal-tempered pitch collections. Bird sounds are complex, changing over time, a fact that makes it difficult to reduce their complexity to something that might be placed in an order for subsequent configuration by xylophone melody patterns. So while Messiaen configured an order-able medium (xylophone) to carry the form abstracted from a complex sound (bird song), the proposition here is to analyse and reduce a domain of sound (bird sounds) into a medium of ordered pitch sounds that can carry the conventionally and musically understood xylophone melody.

4.2. Lucier's *Gentle Fire* (1971)

In a much different way, this idea of choosing what makes two sounds similar is elaborated by Alvin Lucier and his work *Gentle Fire* (1967). The work seeks to address the associations (and illusions) between a sound and the origin of its creation by proposing that sound can conceal the characteristics of another.

By instructing the performer to "slowly" and "gradually" transform the "unpleasant" sound image into a more "pleasant" image underscores the significance that time plays in a listener's perception and interpretation of sound [5]. Equally significant is how the performer "tak[es] care" in processing the sound images so that a transformation is "clearly heard." The method by which the performer develops or chooses to transform one sound image into another can have a profound effect on the listening experience. With these points in mind, one sees that the choice of sound materials for performance is equally critical. For example, many listeners would have difficulty imagining the sound of "crashing planes" and "laughing girls" as somewhat related. But if the performer collects sound images that, despite their origins, share characteristics, then it is possible that listeners may perceive the transformations presented. These three factors - time, method of processing, and sound materials - and the ways they relate to each other have an impact on the composition of musical translations.

In a realisation, at first, a listener may perceive and associate the sound heard with the "unpleasant image." As the piece continues and the transformation begins, the listener's perception may begin as well to include characteristics of "pleasant image." At some point, the listener may begin to flicker back-and-forth between characteristics of both images ; as the experience persists, the listener may become unable to exclusively associate the sound with either of the two sound images. Lacking a method to distinguish which features belong to each of the two sound images, the listener may subsequently decide that she does not know enough about the two imagined sounds, as she questions the features upon which she knows them.

Unable to definitively connect the sound heard with either image, the listener may resolve the matter arbitrarily, thus revealing the personal nature of the experience. By consciously making choices about her perceptions, she discovers how shared characteristics make sounds similar to each other, as they conceal what is distinctive about each. In the best of circumstances, the result of the perceptual challenge can have the effect of re-enchanting the listener with sound, as that which was concealed becomes revealed.

This experience is the result of the cognitive dissonance incurred from indiscriminate perceptions : How can distinct sounds *share* characteristics ? Isn't their difference what makes one classify them as distinct ? No, it is not, because the basis upon which one thinks they are distinct is flawed, they having not been put to the ultimate test in which they are compared with that which they are most similar to. This idea of a sound having the capacity to sound like another sound from Lucier's comment that the "truth is determined, but you're at different values along the way." Lucier's point is about the masking effect of shared characteristics, which are amplified when presented as equivalent through interpolation. Thus, the idea of transforming sounds, through sequencing and processing, so that listeners perceive and interpret them in such a way that they identify the characteristics of one sound as concealed in another sound is very much in line with the composer who creates musical translations.

5. PROPOSAL

5.1. Making Templates

This proposal of finding a way to use bird sounds, for example, as a medium to carry the form of a xylophone excerpt introduces the issue of scale. The scale at which forms are abstracted and transferred could be changed to something more atomic, where the acoustic qualities of a sound, such as frequency and amplitude, may be measured and used to create a template. By using computer technology, one could create a template of a sound and use it to assist in manipulating and sequencing the sound or sound collections that serve as a medium for carrying the forms associated with the analysis sound.

For example, a composer could analyse a single xylophone note, measuring its pitch and amplitude qualities to create a template. With this xylophone template, short bird excerpts could be altered using various forms of digital signal processing, such as band-pass filtering, and sequenced to reflect or evoke the forms and characteristics of the analysed xylophone sound. What becomes quite interesting is how listeners interpret this sound.

The goal of developing and using a template is to create a sound that has the form of a single xylophone note, but is composed of lots of short bird sounds. If listeners heard a sound like this, would they, in their reading of it, flicker between the short bird sounds and larger xylophone form ? What if the same template was used with different sounds ? Would listeners hear the sound of a xylophone if, instead of bird sounds, baby sounds were used ? If these two versions were juxtaposed in sequence, would listeners make associations and connections between the three sound classes ? These are potentially cognitively dissonant listening experiences in which several sounds or sound classes appear to occupy the same space.

5.2. Towards Musical Translation

The challenging, door-opening effect of changed sounds depends upon a listener's experience and understanding of sounds in their most familiar contexts. To a theory on musical translation, with its goal of hearing one sound within another, knowledge of typical forms and contexts becomes critical. This is especially true since the way one might in fact hear one sound written within another is not obvious, given the cognitive dissonance that naturally emerges between the transferred form and the domain of sound it is written into ; faced with such a sound, a listener might well have many questions regarding how to focus attention and otherwise discriminate the sound medium being heard from the form its reconfiguration presents. In short, it is not obvious how and why a listener hears what she hears, especially when the sounds are intentionally challenging.

Not only can sounds be intentionally changed, but also, any reading of a sound as a musical translation is temporary ! That is to say that a sound or sound class enlisted to be a medium for carrying another sound's form at some point reverts back to its original identity. So musical translations, if sensed at all, are temporary and transitory, given the failure of the proposed codes of correspondence and purpose to take hold. The next section discusses, in some detail, how one might abstract and inscribe sound forms by using a suite of programs, written in the computer-languages of C and SuperCollider.

6. ANALYSIS WITH PVCPLUS

In building an analysis-synthesis application, the first goal is to acoustically measure a sound so as to abstract its form, the characteristics that are perceived and associated with the sound. The application first conducts a phase vocoder analysis of a sound by executing the *pvanalysis* script from Paul Koonce's PVCplus suite of programs [6]. The *pvanalysis* routine outputs a phase vocoder analysis file that contains a collection of amplitude and frequency values, which are used by the application to create a template for synthesis.

Different *pvanalysis* parameters, such as FFT size or increasing the number of analysis frames per second (fps), may be adjusted to refine the phase vocoder analysis of a sound. These choices effect the analysis and subsequent synthesis. This being the case, an analysis ceases to be useful once a user identifies important nightingale characteristics otherwise not reflected in the analysis created. At this first level of analysis in which the goal is to abstract form, it is important that the user has a deep understanding of the sound if there is any hope in reconfiguring a sound medium to reflect its form.

7. TEMPLATE CONSTRUCTION

Once a *pvanalysis* data file is created, the application executes a sequence of three programs, *findPartials*, *makeSegments*, and *constructConnections*, which read a *pvanalysis* file and use its data to create a template for synthesis. The programs are written in the C programming language and accept variable user input. Each program prompts users for different input values that correspond to different function parameters, which affect how the template is fabricated. Finally, each program outputs the data onto two files : a binary (.bin) format file and a text (.txt) format file. While the binary files are for processing by other analysis or synthesis programs, the text files are for visualising the data with *gnuplot* [7].

7.1. *findPartials* program

The *findPartials* program reads the *pvanalysis* data file and finds partials in the phase vocoder analysis. To find a

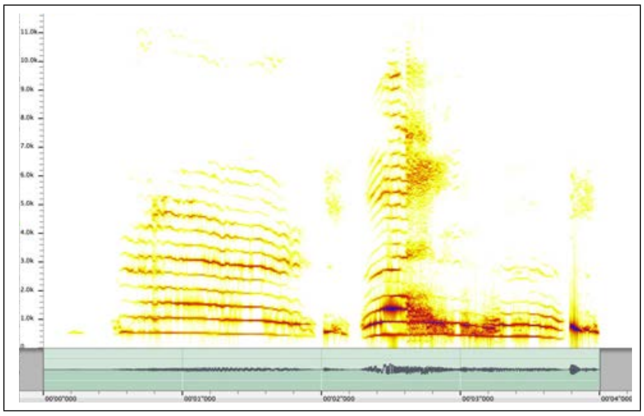

Figure 2. Crying Baby Spectrogram

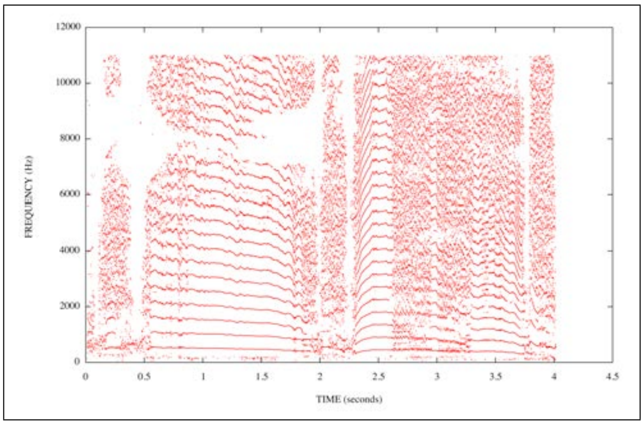

Figure 3. Visualisation of 76066 partials collected by *findPartials* program

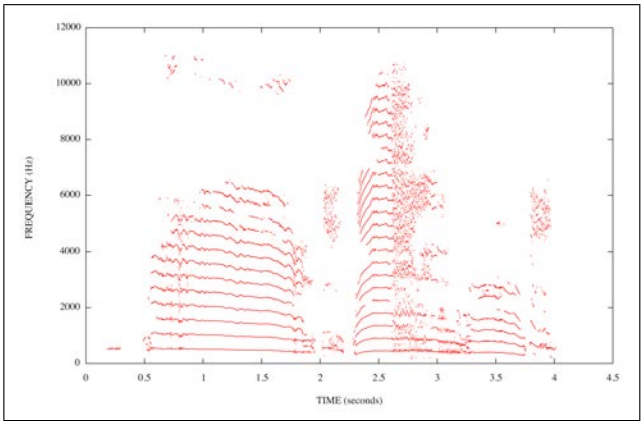

Figure 4. Visualization of 30121 partials collected by *findPartials* program

partial in the *pvanalysis* data file, the program adapts the *get_formants* sub-routine from the PVCplus' *freqresponse* program, which, in reading the *pvanalysis* data, looks at the frequency and amplitude values in each bin for an adjacent set or window of frequency bins and extracts a formant if its values meet particular thresholds. The *findPartials* program prompts users to specify the following input values : the minimum allowable amplitude of a partial (after normalisation) in decibels (dB) ; the lowest and highest allowable partial frequency in Hz ; and a generalised partial selection control that selects the number of partials by evaluating the difference between the prospective partial's average amplitude and its peak amplitude. The generalised control is specified through an output to input range that, relative to the sound's potential for partials, includes a varying number of partials between a calculated maximum and minimum. If the corresponding values of a partial meet these input conditions, the partial is written to a temporary file.

To illustrate the program's functionality, the following shows how two executions of the *findPartials* program with different values affect the number of partials identified in a *pvanalysis* data file of crying baby recording. First, Figure 2 is a spectrogram of a crying baby recording. Looking at the spectrogram, one can identify a preponderance of spectral activity in the frequency range from 60 to 11000 Hz. Thus, in both executions of the *findPartials* program, the minimum and maximum frequencies of an acceptable partial are 60 and 11000 Hz, respectfully.

Figures 3 and 4 visualise two applications of the *findPartials* program configured to produce a maximum of partials with peak amplitudes greater than -70 dB and -40 dB, respectfully. Figures 3 visualises 76066 partials, whereas Figures 4 visualises 30121 partials. By comparing Figures 3 and 4, one might notice many similar shapes in the organisation of partials. The figures representing the higher -40 dB peak amplitude threshold can be seen as reduced versions of the -70 dB implementation, as more than half the partials are eliminated. By eliminating partials, one maximises the variety of partials in a collection. But, how does a user know whether a partial is significant to the analysis ? How does she know that a more reduced collection of partials refines the analysis of a crying baby recording ? Which analysis - an analysis with more or less partials - is better for future synthesis ?

There is no definitive answer to this question, as it depends on the composer, her interpretation of the source material, and her goals in translating it into the target domain. Again these issues of fidelity and transparency relate to the process of fabricating a template that is based on both one's interpretation of the source material and their goal for translating the forms abstracted from it into some other target domain. Of course, with less rigid analysis rules comes the increasing probability that the analysis may not reflect the source. In transferring the template, if one biases transparency through deference to the sounds and context of the target sound domain, then fidelity to the analysed source sound is minimised.

Viewing the two together helps us to isolate and identify the core spectral activity of the source material. While sections of the *findPartials* data visualisations in Figures 3 and 4 appear to be line segments, the partials are in fact unconnected. The next program, *makeSegments*, aims to connect these continuous partials and eliminate those that do not contribute to the perceived characteristics of the sound of a crying baby. So, in this first stage, identifying more, rather than less, partials may be beneficial to the development of a template for synthesis.

7.2. *makeSegments* program

After reading the *findPartials* data, the *makeSegments* program looks for and connects together partials whose proximity to each other suggests continuities. In general, the *makeSegments* program first looks for an unconnected peak partial in the *findPartials* data. Next, the program looks for an adjacent partial in the frames of data to either side of the peak (be it forwards or backwards in time) to form a segment with the peak partial. This process continues in the same temporal direction, each time adding a new adjacent partial to the last one attached, until no more partials can be added to it, after which the process returns to the peak and searches for attachable partials in the other time direction.

The principal complication at the core of the *makeSegments* program is programming the conditions for identifying adjacent partials. By limiting the search for partials in the *findPartials* data to only adjacent bins, then there are only six potential partials with which to connect. However, limiting the options in this way would eliminate many other partials and potentially prevent segments from reflecting greater frequency change through connection to other partials. So, how does the program know whether to connect two partials ? Just because a partial is adjacent to another partial does not necessarily mean they should connect.

The following describes a couple of examples that address these concerns. First, given a recording of a violin sustaining the pitch A4, one can imagine a collection of partials around 440 Hz, as well as partials above this fundamental frequency. Thus, one could devise a method for finding and connecting together all partials at a particular level or within a frequency range. This method may be suitable for recordings of sounds with sustain and very little noise or frequency deviation. But would this method work with the recording of a violin performing various glissandi ? In this case, one could imagine partials located across the frequency spectrum, where, in terms of partial segments, each endpoint would correspond to the starting and ending notes (frequencies) of each glissando. In other words, partial adjacency concerns the frequencies above or below any given partial. Thus, in this instance, this method would not work. In addition, these examples only concern pitch, leaving the matter of noise and noisy partials unaddressed.

To deal with this complication to the construction of partial trajectories, the *makeSegments* program utilises the least squares algorithm. In general, the formula takes into account the slope of a partial segment so as to estimate where the next adjacent partial may - if it in fact does - exist. As previously detailed, the *makeSegments* program first finds an available peak partial in the data set. Given the peak partial's time value, the program subsequently looks backwards in time to find the next available partial. Specifically, the program looks for all partials with a time value that is no more than 1 millisecond earlier than the current partial segment endpoint's time value. If a partial has this time value and its frequency and amplitude values adhere to user input values, then the partial is considered for inclusion in the segment. If many partials are deemed suitable continuations of the segment, then the program selects the partial that best fits, as determined by a series of least squares estimations. This process continues until all partials preceding the peak partial have been located. Subsequently, the same process is applied forwards to all partials occurring after the peak. Finally, once a segment is constructed, the partials that compose it are removed from the data set, and the entire process repeats. Figure 5 is a visualisation of segments constructed by the *makeSegments* program that read the *findPartials* data in Figure 3.

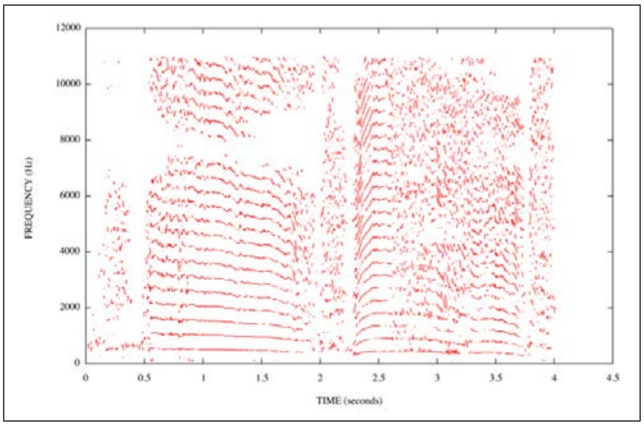

Figure 5. Visualisation of partial segments constructed by *makeSegments* program with a frequency threshold of 100 Hz and an amplitude threshold of 70 dB

7.3. *constructConnections* program

Once these short partial trajectories are constructed, the last step in creating a template for synthesis is to connect together segments whose partial trajectory segments, for whatever reason, remain separated from each other. The *constructConnections* program creates the template for synthesis by reading the *makeSegments* data file and connecting together adjacent segments that form plausible partial trajectories. Similar to the *makeSegments* program, the *constructConnections* program utilises a least squares algorithm to decide whether two partial segments connect. The process of connecting partial segments loops until all partial segments are connected, accepted as suitable, or rejected. Like all of the programs in the application, there are different input values that allow users analytical refinement : a minimum segment length ; the maximum allowable difference in time across which two segments can be connected ; and the maximum range between the segment's frequency extremes.

When determining whether or not a template is complete, one might compare it with a spectrogram of the analysis sound. While visual assessment is not a perfect solution, it does offer some direction. Once a suitable template is produced, the next step is synthesis.

8. SYNTHESIS WITH SUPERCOLLIDER

The last set of programs in the suite is written in the audio programming language SuperCollider (SC) [8]. In general, each program reads the partial segment data from a *constructConnections* binary file and stores each partial's time, frequency, and amplitude values in separate arrays. Following these restructuring processes, the SC programs use the partial segment data in a number of ways to drive the selection, processing, and sequencing of audio recordings. While these programs are continually adapted, depending on both the source material and the inspiration drawn from template information, the following provides a brief overview of their design and functionality.

Written as a series of integrated SC classes, in general, each program takes a variety of user input, including a *constructConnections* data file, an audio recording, and various other parameters that determine the sequencing and processing of sound. As previously stated, while each SC class is different, they are all designed to read the time, frequency, and amplitude values of each partial trajectory. With this in mind, there are two types of processing, temporal and spectral, which differ in the way they relate to the features of each partial trajectory. The two types vary in the method by which the SC program reads and processes the partial values in each partial trajectory, which, in turn, affect the synthesis of sound.

8.1. Temporal processing

The duration of a partial trajectory can be used to control either the playback of an individual sound, assigned to the partial segment, or to the processing of a segment of a longer sound. Given a partial segment, its length, as determined by its endpoints, defines the duration of either a particular recording's playback or the length of time a type of processing is applied to a recording. Depending on need, the length of each partial segment may also be scaled. Scaling the durations allows users to tune the process of sound synthesis to better reflect the characteristics of either the source sound, the sound whose form was abstracted from it in order to modify the target sound media, or the target sound, the sound medium configured to carry the other sound's form.

There are two ways a user may scale the duration of a partial segment. The first allows the duration of each segment to be scaled while preserving its start time. The second allows the overall template duration and its partial segments to be scaled so that the start time and duration of the corresponding audio playback are similarly scaled. In both cases, if the duration of each segment is longer than the duration of the recording, it is possible to loop the recording until the segment duration has completed. In this way, depending on the template file and the choice of recording, different input values can add variety to the synthesised sound.

8.2. Spectral processing

Similarly, there are also input values that affect the reading and processing of spectral features of partial segments. After the *constructConnections* data file is read, each partial's amplitude and frequency values are stored into arrays and loaded into separate buffers. When the SC program iterates through the array data and a partial segment is ready for playback, a synthesis instrument reads the values in from the amplitude and frequency buffers, which in turn drives sound synthesis.

As a way of generalising the data for each partial segment, some input values cause the SC program to calculate the peak and average amplitudes of each segment. Subsequently, these peak and average amplitude values are used instead of the individual partial values stored in the amplitude buffer, as a way of smoothing the data used to synthesise sound. Similarly, the average frequency of partials in a segment or the frequency value at its peak amplitude is used to drive the choice of synthesis frequencies.

Another way of changing the type of synthesis based on the template involves choosing different synthesis instruments for audio playback and processing. Many different types of synthesis instruments are available and deal with a variety of source materials and user input values, including synthesis instruments that band-pass filter recordings at particular frequencies based on partial segment data.

9. EXAMPLES

This section describes two examples developed by the application. Before outlining the technical ideas behind them, the following addresses their inspiration. The goal of these examples to create sounds that cause listeners to *flicker* between two sound images, where a flickering sound image is one in which perception shifts back and forth, as the other sound image is continuously re-located or re-discovered. The first example seeks to abstract and inscribe the form of a crying baby into water sounds, such that a listener might perceive and identify it as "crying water" or "water babies." Similarly, the second example seeks to use the analysis of a stream recording to order the sounds of violin pizzicati and create a "stream of pizzicati." While the form of a crying baby is quite robust and distinct, the form of water is, typically, not. Thus, these two example, collectively, aim to illustrate the importance of selecting distinct forms for inscription.

9.1. Crying Water or Water Babies

Not only is selecting a recording of a crying baby with a distinct form important, but also a water recording with a fairly even distribution of spectral activity. Figure 2 is the spectrogram of the crying baby recording and Figure 6 is the spectrogram of a small stream recording. To reiterate, in order to compose a musical translation, a sound

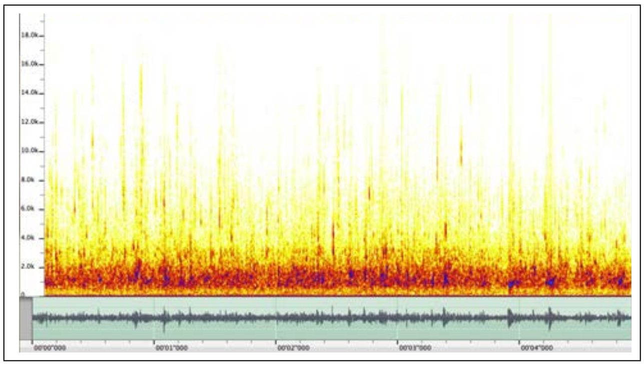

Figure 6. Spectrogram of small stream recording

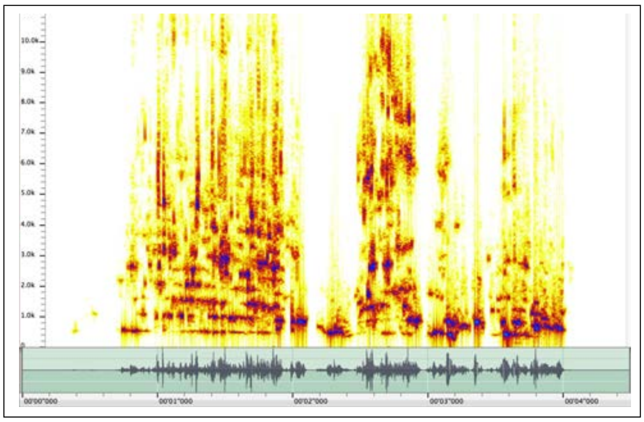

Figure 7. Spectrogram of stream recording as filtered by the crying baby template

medium that is not only distinct, with its own unique set of characteristics, but also capable of carrying the forms associated with another sound is required.

Once a crying baby template is constructed and a sound material is selected to be reconfigured by it, the next step involves synthesis. Figure 7 is the spectrogram of the sound synthesised in SC, which was produced by reading the crying baby template and using the partial segment data to filter the recording of a stream. To create this sound, a specific SC class, which reads the crying baby template and uses the frequency values of each partial segment to band-pass filter the stream recording. Briefly, the SC program first plays the stream recording and sends the audio output to an auxiliary bus. Next, a Synth reads the audio from the auxiliary bus at the appropriate time and filters it by using each partial segment's frequency values, as stored in a buffer, to control the centre frequency of a band-pass filter.

By comparing Figure 2 with Figure 7, one might notice many similarities and differences. That said, this process is not about making an exact copy of the spectrogram, but rather about synthesising sound that invites listeners to perceive and identify the forms associated with the sound of a crying baby as preserved in the context of water sounds. The synthesised sound featured in Figure 7 appears in the author's fixed-media work *along the eaves* from 0'54" to 0'57".

9.2. Stream of Pizzicati

Although the forms abstracted and associated with different water sounds are distinct in their own right, in general, one might describe them as generic - this of course was a reason why one might choose water to act as a medium for carrying the form associated with the sound of a crying baby as described in the previous example. When listeners hear a stream, they might hear lots of little splashes and droplets of water and possibly a direction as to where the stream flows. But in no way can they perceive the sounds of the stream in any particular order or sequence. In terms of acoustic measurement, one might imagine various spectral activities that do not have any specific order. Knowing that the analysis of a sound serves as a basis for creating a template for synthesis, how does one overcome this problem of analysing a sound that is dense with chaotic spectral material ?

Since the goal is to order violin pizzicati sounds, fortunately, the program only needs to identify frequency values between G3 (195.998 Hz) and G6 (1567.982 Hz). Because this obviously excludes many of the higher partials in a violin pizzicato sound, one could use lower violin notes with severe band-pass filtering that tracks the high partial segments. This would allow the inclusion of significant frequencies in the upper spectrum of the water. Additionally, when a listener hears a stream, she does not perceive every droplet in the stream, but rather general splashes of water. So, when fabricating a template for synthesis, one wants to connect only partial segments that are long enough to presumably represent these splashes. This is a way to reduce and filter out some of the spectral information that was collected in the initial phase vocoder analysis of the sound to create a template that better reflects the listener's perception. Thus, the goal is to identify features that are not only characteristically stream-like, but also capable of being mapped into the target sound domain of violin pizzicati.

Figure 8 is a visualisation of a stream template created from the analysis of seven seconds of the stream recording. One might notice that the partial segments are quite squiggly, which reflect the rapid changes in frequency that could be translated into string glissandi. One could imagine that for every squiggling segment, a violin pizzicato note is selected and, to reflect the segment's continuous and rapid changes in pitch, on could change the rate of audio playback. That being said, would listeners hear this collection of violin pizzicato glissandi as carrying the forms associated with a stream ? In addition, is seven seconds is enough time for listeners to hear and make connections between the sounds of a stream and violin pizzicati.

Unfortunately, it is hard to imagine listeners being inclined to make a connection between the sounds of a collection of pizzicati glissandi and a stream. If one wants to produce the affect of a gradual transformation of one sound image into another, one must counterpose the essential characteristics of each, which may take time for listeners to perceive and identify. Thus, in some cases, it is important to scale the temporal values of the template,

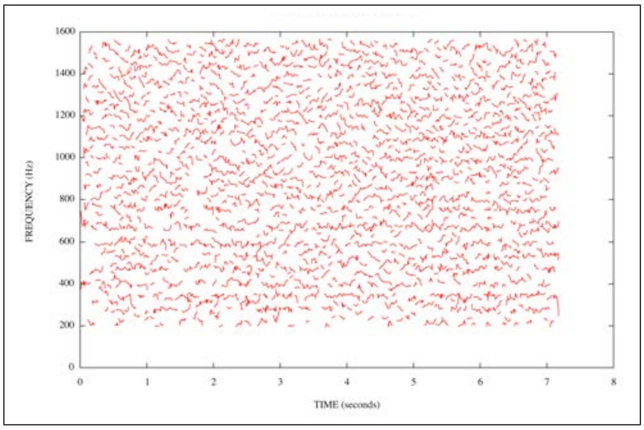

Figure 8. Visualisation of small stream template constructed by *constructConnections*

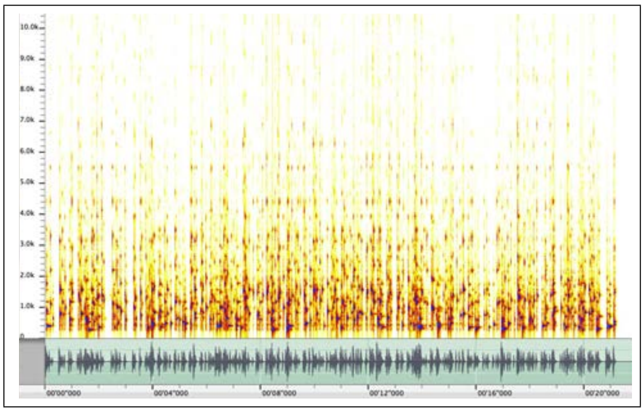

Figure 9. Spectrogram of synthesised sound using the stream template with violin pizzicato recordings.

as a way of providing listeners time to make discoveries.

Figure 9 is the spectrogram of a sound synthesised in SC, which used the stream template to select, sequence, and process recordings of violin pizzicati by first scaling all of its time values. To briefly describe this process, the SC program read the stream template in Figure 8 and scaled the start time value and duration of every partial segment by three. Once all of the template values were scaled, a fundamental frequency for each segment was determined using either the segment's average frequency or the frequency of its peak partial. Once determined, the frequency was used to select and play the closest chromatic scale violin pizzicato with the playback rate continuously adjusted to reflect the segment's squiggling changes in pitch.

As a way of creating some correspondence between the images of a stream and a violin pizzicato, one could use the stream template to order and process lots of short water recordings. Since water is not an equal-tempered instrument, one could then band-pass filter select water recordings relative to the frequency values in a given partial segment. By using the same stream template, one could create a link between the frequencies of the violin pizzicati and (band-pass filtered) water. One might then mix of these collections of sounds to create a hybrid sound that invites listeners to perceive and identify the characteristics of both sound images.

Of course by mixing the two sounds together does not create a "true" musical translation, as both sound images are being used in sequence. That said, just because listeners may be able to perceive and identify the sound of water and pizzicati does not mean they will make a leap to hearing the form of one (the stream) inscribed into the other (the collection of violin pizzicati). Despite the obvious problems with this example, it nonetheless begins to show some of the methods a composer might use to create cognitively dissonant experiences of sound, experiences that, in the best of cases, invite listeners to become re-enchanted with sound.

10. CONCLUSION

One may suppose that my process of creating a template and then using it to guide the arrangement of sound in another domain is a form of mapping. There are at least two mapped territories involved in the process. First, there is the template, which maps the sound using the template. Second, there is the inscription of the template information into the new sound domain. Being a two-step process, the quality of the second mapping is necessarily a consequence of both mappings, inheriting whatever transfer of form is or isn't achieved at each stage. Successful transfer at both stages would be required to claim transfer from the original sound into the new domain. Conversely, a failure to capture the sound in the template could not be followed by a successful transfer of the template into the new domain. Similarly, success at capturing the sound in the template could be followed by a failure to transfer the template into the new sound domain, as a result of the process or sounds used.

One could say that the overall mapping process involved with creating a template and using it to synthesise sound, in general, is affected by what each individual process subtracts and then adds. That is to say, the first process (the making of the template), which is about abstraction, necessarily omits information, whereas the second process, which is about inscribing the abstracted information into a new domain, necessarily adds information. How much information is subtracted and added, respectively, is what directly affects one's perception of the larger mapping. With little lost and little added, the mapping would be fairly equivalent. Whereas one or both of increased omissions and additions would begin to challenge, quite literally, evidence of the mapping, as the mapped form is increasingly lost through either a lack of proper representation of the form or a context that masks the form by the noise of what it adds. Thus, my aim in developing my process of mapping is to connect existing forms in search of mimicry that hides abstracted forms inside domains of mapping or inscription.

This process of drawing connections between sounds through the transfer of forms might be called translational. However, just because this connection is considered translational does not mean it is communicative, be it at the level of a spoken language or a map, seemingly designed to assist in understanding the sounds analysed.

11. REFERENCES

[1] Venuti, L. *The Translator's Invisibility : A History of Translation*, Routledge, London, 1995.

[2] Korzybski, A. *Science and Sanity : An Introduction to Non-Aristotelian Systems and General Semantics* 5th ed., International Non-Aristotelian Library Pub. Co, Lancaster, PA, 1945.

[3] Eco, U. *Experiences in Translation*, trans. by A. McEwen, University of Toronto Press, Toronto, 2001.

[4] Maturana, H. "Biology of language : The epistemology of reality," *Psychology and Biology of Language and Thought : Essays in Honor of Eric Lenneberg*, edited by G. Miller and E. Lenneberg, Academic Press, New York (1978), pp. 26-64.

[5] Lucier, A. and D. Simon. *Chambers*, Wesleyan University Press, Middletown, 1980, pp. 128-136.

[6] Koonce, P. *PVCplus*, software. Available at http://sourceforge.net/projects/pvcplus/

[7] Williams, T. and C. Kelly. *gnuplot*, software. Available at http://www.gnuplot.info/.

[8] McCarthy, J. *SuperCollider*, software. Available at http://supercollider.sourceforge.net/