The Integration of the PCSlib PD library in a Touch-Sensitive Interface with Musical Application

José Rafael Subía Valdez
IUNA - UNQ
Buenos Aires
Argentina
jsubiavaldez@gmail.com

Abstract

This paper describes the study and use of the PC-Slib library for Pure Data and its implementation in the project "Interface Design for the development of a touch screen with musical application" [Causa, 2011] The project consists of a touch-sensitive interface that allows the drawing of musical gestures that are then mapped to a harmonic structure generated by the PC-Slib library. Pure Data also is responsible of the translation and reproduction of the musical gestures via MIDI.

Keywords

Pitch Class Sets, XML, Musical Gesture, Harmonic Structure

1 Introduction

The creation of touch sensitive screens over the years, has led to many performance instruments and tools. However, this technology has not yet entered the music writing and composing for fixed media category such as scores. Touch Screen instruments like the *ReacTable* [1] or the *Lemur* [2] have existed for a few years now. These innovative instruments have broken down technological walls permitting the development of more interfaces including the one developed in this project. Nevertheless, the *ReacTable*, the *Kitara Digital Guitar* [3] and such, have exploited the "Real-Time" characteristic of these interfaces. Their use has been focalized as instruments to perform and not so much to write music. Some other developments that approach the same questions stated in this project have been scarcely documented. French composer, Philippe Leroux used a system that translated drawings entered in a *Wacom* [4] tablet to pitches resembling the inputted sketch [Vassilandonakis, 2008]. This project used OpenMusic [5] to do so. Nonetheless, while Leroux's system intends to capture human gesture such as hand writing to be used or to create music structures, this project explores the use of codified traditional nomenclature. "Ugarit" is a touch sensitive screen that allows the writing of music by entering codified drawings that represent specific and traditional musical gestures.The result is a graphic score that will only produce sound after the notes are entered. With the help of the *Pitch Class Sets* [Forte, 1974] theory and its implementation in Pure Data through the PCSlib library, "Ugarit" lets the user concentrate in writing musical gestures and building music pieces without preoccupying him or herself of the harmonic structure. "Ugarit" maps the drawings to a harmonic structure that the system previously creates allowing its operator to think the writing of music in a different way.

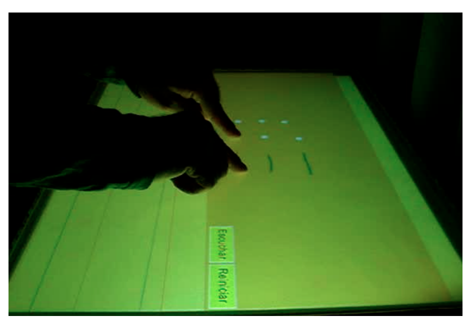

Figure 1: "Ugarit" Multitouch screen

"Ugarit" could also allows the teaching of mod-

[1] www.reactable.com

[2] www.jazzmutant.com/lemur_overview

[3] www.misadigital.com/guitars.htm

[4] www.wacom.com/

[5] http://repmus.ircam.fr/openmusic/home

ern music theories in a unique manner. Its cheap development, and its friendly interface, lets it to be assembled in schools everywhere. "Ugarit" would be perfect for teaching music for children or any kind of non-specialized public. Because it uses a modern approach to music paradigms, it would be updating the music theory taught to the public. It allows users to create music focusing in melodic contours and assembling musical gestures in time. Such ways of thinking music are the key to understand the creation of some of the modern music of the XXI century. It lets the user experiment with these nontraditional music concepts by taking care of complex harmonic structures internally. This permits the creation of "modern-like" music without the full knowledge of complex concepts needed to produce it.

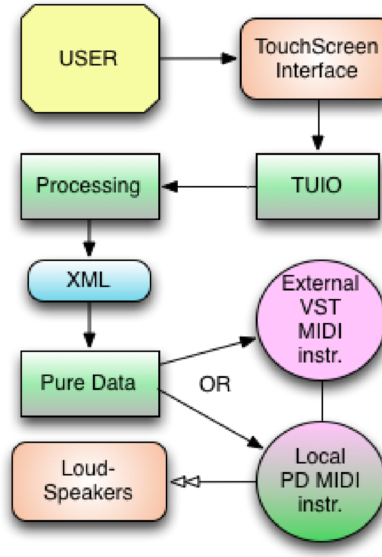

Figure 2: System workflow

2 Data Interpretation (Connection of the Touch Sensitive Interface and Pure Data)

2.1 Musical Data Entry

A team of four artists/programmers that were divided in two groups developed the complete system. Emiliano Causa and Sebastian G. Botasi developed the graphical part of the interface while Matías Romero Costas and the author of this paper programmed the audio part. This paper only describes in detail the implementation of a specific part of the program, but a brief schematic of the entire workflow is illustrated in the block diagram bellow. [fig. 2]

The musical data entry is accomplished through an XML[6] file generated by Processing[7] after the census and analysis of the touchscreen interface drawing. PD then reads the file through the "Detox" object. This object is developed as an external in the "jasch_lib" library developed by Jan Schacher. "Detox" allows reading the XML file informing the user when a "tag-tree" is opened or closed. The data is then routed through a series of abstractions that stores it in "coll" objects. Matías Romero Costas[8] developed this part of the program. It was later modified and completed with the part of the program that maps the values of the surface to the pitches of the harmonic structure, this will be explained later.

2.2 Music Gesture players

The musical gestures implemented in the touch screen interface are "chord", "trill", "tremolo", "melody", "note", "arpeggio" and "glissando". All of them were implemented in a way that all action constituting the gesture, are united in a group of parameters. For this, the players assemble messages, used as buffers or temporary memories, that feed the "makenote" and "noteout" objects. "Makenote" and "Noteout" objects convert and send the processed midi messages to the corresponding outputs. These messages are created by receiving stored data from the "col" objects that deliver stored data in groups corresponding to gestures one at a time when the groups indexes are equivalent to playback time.

2.2.1 Tremolo and Trill

Essentially, the algorithm for trill and tremolo are the same since the gesture works the same way. A buffer generates two messages that store the two values to be interleaved at a certain speed during the duration of the event. The only difference is that the tremolo player. Sets the two messages with the two pitches entering from the screen, while the trill player only receives one pitch and trills with a semitone above it. This is accom-

[6] XML files are text files with especific formats that can be interpreted by different programs

[7] http://www.processing.org/

[8] For more information, read documentation writen by the autor.

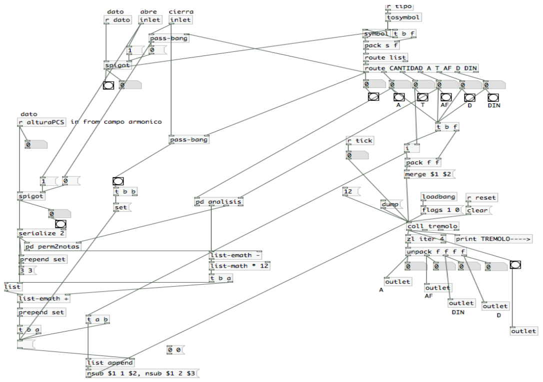

Figure 3: Tremolo Subpatch where the“coll” object is

plished by simply adding one to the entered pitch.

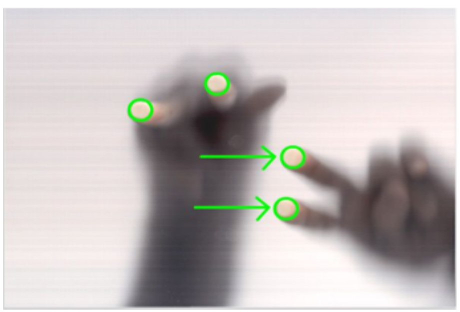

Figure 4: specific gesture to draw that produces a “tremolo”

2.2.2 Chord

The chord player uses a single line message as a buffer and builds one with all the pitches of the chord. The program must set the message correctly inserting the pitches and key velocities of the notes for the “makenote” object to play the chord properly. For this player, as well as for the “arpeggio” player, it was necessary to modify the abstractions where “coll” objects are, in order to filter the different start times of each individual note of the gesture entering from the surface and set only the time of the first note as the same for each. This was achieved with the abstraction “once” developed by Thomas Musil in Austria. It allows only the first value to pass acting as a gate.

2.2.3 Arpeggio

The Arpeggio playback, as well as the chord playback, results with the same modification of abstractions where the corresponding “coll” object is placed. In addition, the arpeggio involves using the start time of the first note and then an algorithm that plays the rest of the notes in succession. For this, we had to use 5 messages as temporary memory where the arpeggio notes (5 maximum) would be stored and then executed by a small counter that triggers them one by one in rapid succession, a fundamental characteristic of the arpeggio.

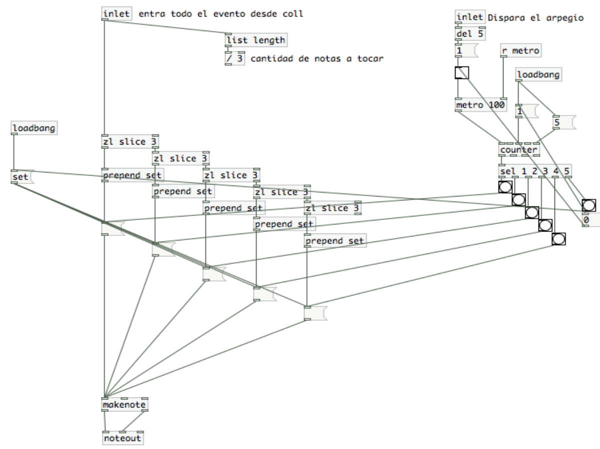

Figure 5: Arpeggio Player Subpatch

2.2.4 Glissandos

The glissando player is very simple. It always makes a linear interpolation between one note and another. It uses the note from which the glissando starts and the note to which it arrives and then interpolates these two pitches linearly using the “step” object, the object can also set the step size of the increase.

2.2.5 Melody and Notes

The two gestures “melody” and “note” use the same player called “NotaPlayer” (NotePlayer) since their respective abstractions where the “col” object are, delivers the pitches used in the musical gesture one by one. This allows the use of a single note player because the gesture may be divided temporarily without any problem and each note can have different and independent durations.

3 Pitch structuring using PCSlib

3.1 Basic Pitch Class Set theory background

Howard Hanson first introduced Pitch Class Set (PCS) theory in the 1960’s. It was initially created as a new way to analyze and classify tonal music but soon music theorists like Allen Forte and Robert Morris started to use it for atonal temperate music. The theory is based in the interval relations created between notes in a music piece. It allowed theorists to find coherent and close relations in harmonic structures of modern music that were difficult to find in those times.

The PCS theory uses the same analog Set Theory used in combinatorial algebra seen in mathematics. It allows classifying and studying relations between groups of notes that have the same interval characteristics. The theory establishes that each group of notes has a prime form, this permits a better way to classify groups of notes. To do this, the theory considers that all pitches should be enumerated from zero to eleven starting from C (see Table 1). This admits a better order for classifying same set types that only differ in the pitches involved (see Table 2). It also establishes that the octave relation is not important, meaning that if a Db4 is played and a F6 follows, the interval taken into consideration would be a major third, ignoring the two-octave difference.

C	C#	D	D#	E	F	F#	G	G#	A	A#	B
0	1	2	3	4	5	6	7	8	9	10	11

Table 1: Cromatic Scale enumerated from 0

Finally, the PCS theory allows mathematic operations in groups of pitches. Some simple operations include transposition, inversion and retrogradation. More complex operations are also viable and are the true potential of this theory applied to composition.

0	3	4	7	PCS 4-17 Prime Form
1	10	9	6	PCS 4-17 Inverted & Transposed

Table 2: Same Class Set with different Pitches, note that the same Set Class has the same intervals

3.2 Harmonic Structure Creation

The harmonic structure used is made with the "pcs_chain" object part of the PCSlib library [Di Liscia and Cetta, 2011a]. This object produces Pitch Class Set chains of the same set class [Di Liscia and Cetta, 2011b]. This part of the program is located in a subpatch and lets you choose a PCS of five or six notes. Then the object will split the PCS in two and will provide all possible

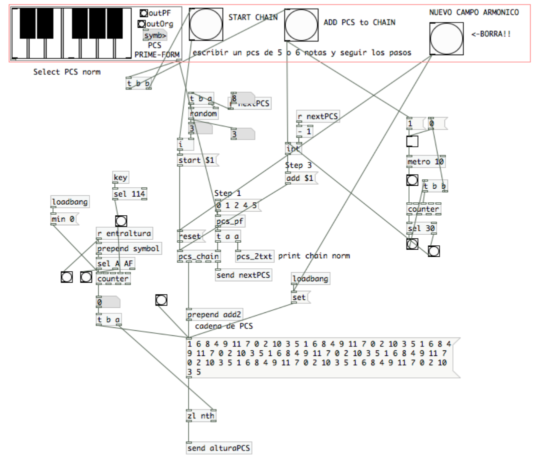

Figure 6: Patch that generates de Harminoc Structure with "pcs_chain"

combinations. The user then selects the partition to start the chain. The object provides a list of all possible partitions; the last option is always the best fit to saturate the total chromatic scale. The program uses this option to create the harmonic structure.[fig. 6]

3.3 Mapping of the harmonic structure to the inputted musical gesture

The harmonic structure will be essential to the color of any gesture entered. To achieve this, a compromise is necessary between the gesture, which is delivered in discrete MIDI values, and the pitches that will have to be changed to correctly fit the harmonic structure. This allows musical coherence of any piece of music written on the surface and allows the user greater freedom to concentrate on musical gesture without worrying about the pitches to use.

To accomplish this, we had to develop several algorithms that find the best accommodation of the pitches incoming from the gestures drawn on the surface to the harmonic structure. This way, the distortion between the drawing in the surface and the resulting pitches played is set to the minimum as possible. A clear example of such distortion would be the accommodating of the pitches from the drawn gesture to pitches that are too far apart from the registry and "ambitus" of the gesture entered. [fig. 7]

Through different algorithms the program is able to restructure the pitches inputted. Each solution was different for each musical gesture be-

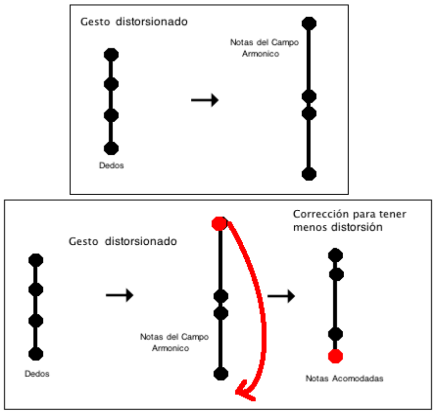

Figure 7: Diagram of how a part of the Permutation Algorithm works

cause the Pitch Class Sets theory relates to each gesture in a different manner. This means that the algorithm in the illustration above will not work to accommodate the pitches of the gesture called "melody". However, all the particular solutions to the problem have the same analysis system [Di Liscia and Cetta, 2011c] for the data entry from the surface. The analysis will help compare the pitches of the gesture to the pitches of the harmonic structure created. It will sort the data and extract the absolute pitches eliminating the octave relationship. On the other hand, it does keep track of the "octave" in which the pitches are subtracted from, finally with the help of the object "pcs_pf" it is able to find out to which set class the pitches entered form. All programming is achieved thanks to a set of abstractions included in "Pd-extended" known as "list-abs". These abstractions allow the manipulation of lists, a cornerstone in algorithmic programming for pitch relation manipulation.

3.3.1 Mapping of the gesture "chord" and "arpeggio" to PCS

The program that maps the music gesture "chord" and "arpeggio" is the same due to the fact that the "arpeggio" is a chord whose notes are deployed in time when played. The solution to the correct mapping has several steps. In addition to the analysis [fig. 8] previously explained, the program must compare the arrangement of the incoming pitches from the chord generated to the pitches previously generated in the harmonic structure.

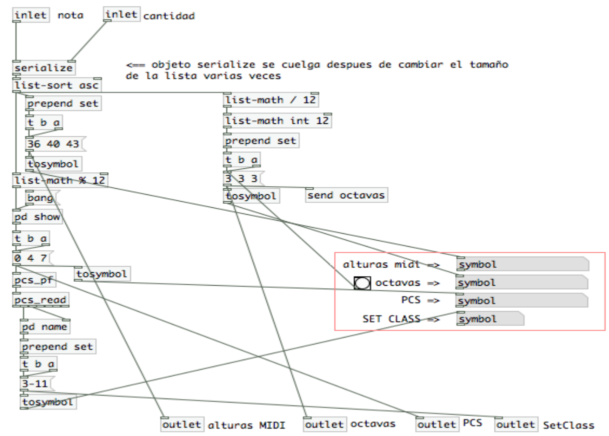

Figure 8: Patch that analizes the incoming MIDI pitches

This operation is performed by subtracting one chord to the other and finding the smallest difference in intervallic distance from one another. Then, the chord generated is swapped with the object "pcs_perm" until the permutation with least difference is found. This means that the chord that replaces the entrant is chosen according to which has the greatest similarity in the disposition of the entered pitches and the area covered by the chord. The program does not analyze deeper if there are several permutations with the same result, it chooses the first one with the least difference to optimize processing time. Finally, the program seeks the best accommodation for octave placement with an algorithm which states that if the difference in semitones between two notes is higher than 6, the interval can be inverted to bring one chord closer to the *ambitus* of the original inputted chord. Example of the algorithm, the 2 PCS are subtracted and then the intervals are added. (see Table 3 & Table 4).

3		5		7		10	Touch-screen
4		0		11		7	Harm. Struct.
1	+	5	+	4	+	3	= 13

Table 3: Interval Subtraction of PCS & addition of intervals

3		5		7		10	Touch-screen
0		4		7		11	Harm. Struct.
3	+	1	+	7	+	11	= 5 Best Option

Table 4: Interval Subtraction of PCS & addition of intervals (Best Option for Replacement)

3.3.2 Mapping of the gesture "glissando" and "tremolo" to PCS

Gestures "glissando" and "tremolo" were equally resolved because both receive data from the XML file the same way. The XML file delivers the initial and final pitch that the players need to reproduce the gestures, which must be accommodated to reproduce correctly the pitches entering from the harmonic structure. For this, the permutation algorithm explained in the above process is the same, but was simplified by omitting the "pcs_perm" object. The program compares the intervals entered form the XML file and the extracted from the harmonic structure. If the difference between the two intervals exceeds the augmented forth, the interval is inverted to better resemble the drawn gesture in the surface.

3.3.3 Mapping of the gesture "melodía" (melody) to PCS

The mapping of "melody" gesture to the previously designated PCS in the harmonic structure was accomplished in several steps. First, you must understand the concept of "melodic contour" and how the algorithm keeps its design and direction regardless of whether the area in which it develops is distorted. This is necessary because for the contour drawing to be maintained, it is more important to keep the direction of the intervals before the closeness of the notes.

The program creates a matrix where the two different PCS are entered, one incoming from the XML file and the other from the harmonic structure. The PCS are sorted according to a position index designated by the melody. The pitches are later rearranged in ascending order. First, an index number is allocated to each pitch of the melody entered from the XML file. For example, if you enter the PCS [3 0 7 9 5] as the melody contour, an index is given to each pitch producing a matrix like shown below. (see Table 5).

It is later re-order in ascending order for it to later be compared with the PCS entering from the harmonic structure. (see Table 6).

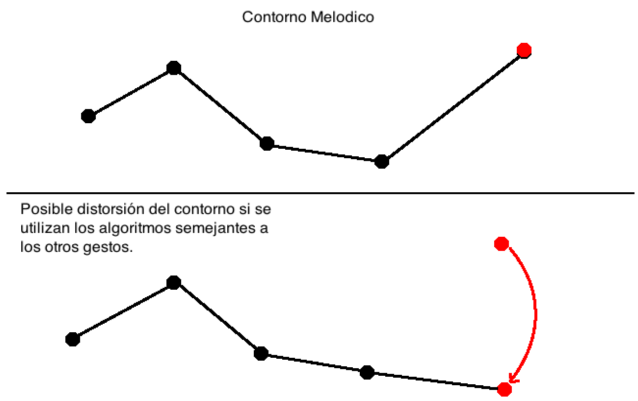

Figure 9: Diagram that shows the posible distortion of the gesture "melody"

0	1	2	3	4	Index (Sorted)
3	0	7	9	5	PCS from the XML

Table 5: Step 1 of the melodic contour mapping

1	0	4	2	3	Index (un-sorted)
0	3	5	7	9	PCS from the XML

Table 6: Step 2 of the melodic contour mapping

Now the PCS from the harmonic structure can be entered, it is sorted in ascending order and inserted into the matrix. For example, the PCS from the harmonic structure will be [4 11 6 2 8] that once re-ordered will look as follows: [2 4 6 8 11]. Now you can place that vector in the table and the matrix will be as follows (see Table 7).

1	0	4	2	3	Index (un-sorted)
0	3	5	7	9	PCS from the XML
2	4	6	8	11	PCS from Harm. Structure

Table 7: Step 3 of the melodic contour mapping

Thus we have the two PCS sorted from lowest to highest order and the index vector from the melodic order is now un-sorted. Then all that remains is to rearrange the matrix according to the first vector re-ordering from smallest to largest and extract the row number three. Resulting in the following. (see Table 8).

0	1	2	3	4	Index (Sorted)
3	0	7	9	5	PCS from the XML
4	8	11	2	6	PCS from Harm. Structure

Table 8: Step 4 of the melodic contour mapping

The PCS from the harmonic structure in the correct order to keep the melodic contour is [4 2 8 11 6]. As for the drawing of the melody, the result is the contour maintenance and so, it causes the least amount of distortion of the drawing entered from the surface.

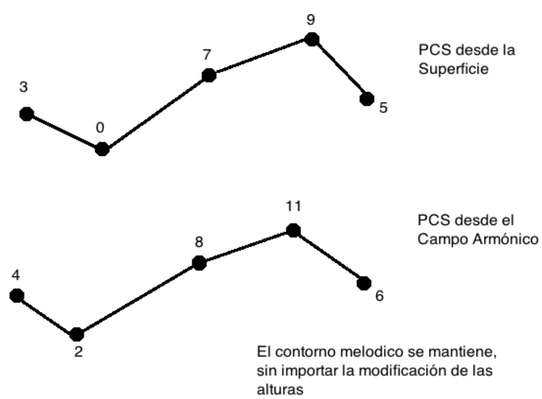

Figure 10: melodic contour maintained dispite the new pitches

All this part of program was development in the "abstraction" called "zzzzmelodia" and uses the "matrix" object included in the "iemmatrix" library. Finally, the new pitches are replaced in the "coll" object, which are then reproduced when the complete surface input is executed.

3.3.4 NO - Mapping of the gesture "Nota" (note) to PCS

The music gesture "nota" (note) was the only one left without mapping to a previously designated pitch entering from the harmonic structure. This is because the independence that "nota" has as a gesture and its immediate relation to a pitch, led to the conclusion that it was the only gesture totally free from the harmonic structure mapping process.

4 Conclusions

The Touch-Sensitive Interface with Musical Application was developed in 2011 and exhibited as a prototype on September 23, 2011. Throughout its development, objectives were achieved by solving problems step by step. Pure Data proved to be very versatile for data processing as implemented in this project even though it is difficult to achieve this type of programs in environments known as "max". The PCSlib library showed to be very complete and it allowed experimental approaches in the creation of harmonic structures with the use of the Pitch Class Sets theory. The touch sensitive surface named "Ugarit" opens new paradigms for cheap technology with extreme potential for teaching music in new ways. It could take music education a step closer to modern music by implementing PCS theory in the teaching of music and modern music theories by underlining the importance of musical gesture and intervallic relations.

The "Ugarit" was developed in a research program of a public university in Argentina. It is still in a complete experimental stage due to lack of income. Because the team depends entirely on public resources, the further development of the project is uncertain. However, its success after presenting it, proved to be a viable project. Now the developing team must wait and see what will become of this project. Most recently they have presented a complete report of the project and all the work done during this early stage. For more information about this project, such as requirements and building steps, it is encouraged to contact the author of this paper.

5 Acknowledgements

The Author would like to thank the entire team; Emiliano Causa, Sebastian G. Botasi & Matías Romero Costas. With whom he worked during the complete development of "Ugarit". And a special "thank you" to Dr. Pablo Di Liscia & Dr. Pablo Cetta for their guideness over the years.

PCSlib belongs to the research project "Musical Applications of sets and combinatorial matrices of set classes" Director Dr. Oscar Pablo Di Liscia and Dr. Pablo Cetta, in Quilmes National University.

Touch-Sensitive Interface with Musical Application belongs to the research project "Design and development of applications and interfaces for augmented reality for synthesis and digital audio processing" Part of the program PICTO-

ART. Director Carmelo Saitta and Pablo Cetta in Area Transdepartamental of Multimedia Arts from National College of Art (IUNA)

References

Emiliano Causa. 2011. *Diseño de interface para el desarrollo de una pantalla sensible al tacto con aplicación musical.* Revista de Investigación Multimedia (RIM), Buenos Aires, Argentina.

Pablo Di Liscia and Pablo Cetta. 2011a. *Composición asistida en entorno PD.* Revista de Investigación Multimedia (RIM), Buenos Aires, Argentina.

Pablo Di Liscia and Pablo Cetta. 2011b. *Elementos de Contrapunto Atonal.* EDUCA, Buenos Aires, Argentina.

Pablo Di Liscia and Pablo Cetta. 2011c. *Medidas de similitud entre sucesiones ordenadas de grados cromáticos.* Revista de Investigación Multimedia (RIM), Buenos Aires, Argentina.

Allen Forte. 1974. *The Structure of Atonal Music.* Yale University Press, London.

Miller Puckette. Pd documentation.

Yiorgos Vassilandonakis. 2008. An interview with philippe leroux.