G54UBI final report (2015/16)

Title: Virtual Keyboard, approaches to digital music instruments

Student ID: 4260783
Date: 06/JAN/2016

Summary

The purpose of this report is to describe the highlights of the development of a virtual keyboard that has been created under the principles of the Ubiquitous Computing. This musical instrument is thought as a metaphor of a traditional piano, however, the challenge is to delete the physical interface producing sounds based on the data recognizement. Accordingly, in this coursework will be described the conceptual basis that have been identified to explore the new musical instrument, as well as, the criteria that have been taken into account to design the system. Additionally, the elements that allows the best performance of the artifact will be outlined, those are the structure of the algorithm and the features of the code. Finally, a set of two evaluations will be described to understand the relationship between the sensors and the accuracy of the virtual keyboard.

Background and motivation

The virtual keyboard is a system designed for keyboard performers who want to explore the essence of the digital music. It has been thought as a portable instrument that should fill in a small cube, with the possibility to adapt its commands to the surface on which it place it, likewise, the option to calibrate the type of sound and effects to reproduce the music performed.

The artifact has been designed to show a magical effect in the spectators, the interface of the instrument is hidden and the musical results are magnified to the public. The physical keyboard is been deleted, therefore, the musician interacts with the instrument through the auditive feedback without constraints between the notes, in other words, resembling the fingerboard of a double bass. Nonetheless, the effects of pressing a non-existing keyboard are revealed with the music composition that is played. In the Figure 1 is displayed in a sketch the essence of the virtual keyboard in a randomic scenario.

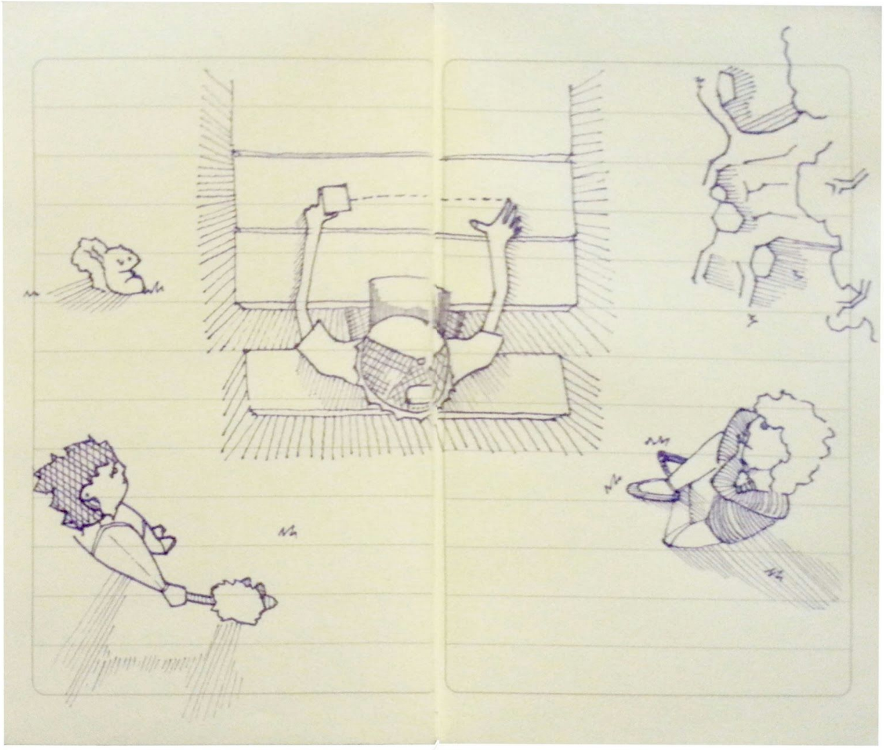


Figure 1 - Escene of a virtual keyboard performer in a park

To make possible this project are implied other several systems that will be listed but not implemented like: a software application with a effects library, a Rotary Angle Sensor to modify the effects, a set of portable speakers, an electric battery, a designed cubic case, among others.

Related Work

Based in previous frameworks, two principles has been defined to explore the digital music world. On one hand, the concept of *invisibility* considered in PARC; on the other hand, the guidelines for crafting musical computers published in Barcelona.

As mention Weiser, artifacts might be examined as tools to achieve tasks, nonetheless, most of the time it has been designed as the center of attention of the user. Thus, Ubiquitous Computing, introduce a different scenario developing objects merged with its environment, specially, hiding the machinery of the computers (Weiser, 1993). In the a like manner, the main effort of this product is to mask the technology used for data collection and to pretend that the interface disappear.

In the last decade relevant progress in the digital musics has been developed, particularly in the creation of new musical instruments. Thus, studies in University of Pompeu Fabra have made several approaches to digital lutheries explaining in three steps the routes that engaged computer science with digital music instruments. Firstly, to identify the essentiality of the digital music; secondly, to determine the problems in traditional instruments; thirdly, to reduce the problematic mixing the logic of traditional instruments with the peculiarities of digital music (Jordà, 2005).

Design

The artifact has been assembled connecting a set of sensors on the boards, the aim is to identify when an event is produced by hitting in a particular place. In the Figure 2 the configuration of the system is shown; (A) a GrovePi and a Raspberry Pi boards are connected together; (B) in the port A0 is plugged a Sound Sensor to digitalize sound waves; (C) in the port D4 is plugged an Ultrasonic Range Sensor to detect the distance between the artifact and the subject; (D) events in particular range and frequency are collected, later transformed in musical notes; (E) LCD RGB Backlight, which is plugged in the port I2C-1, displays the note played.

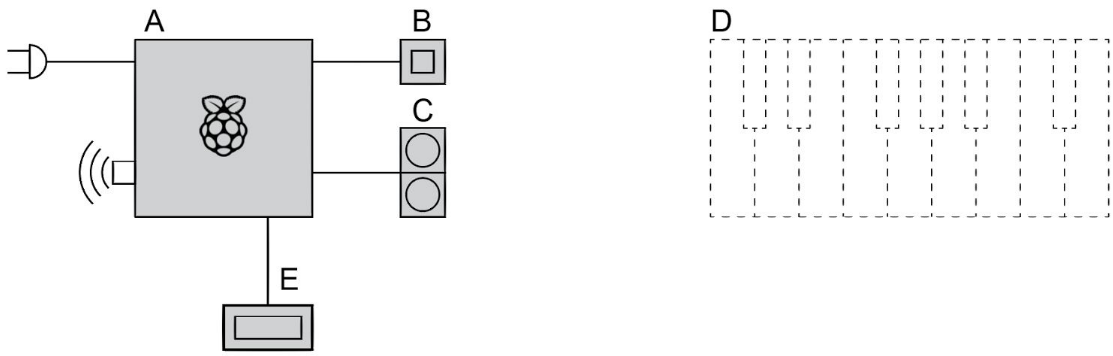


Figure 2 - Diagram of the interaction design

The sequence of the algorithm starts when the system receives inputs of the environment, accordingly, a serie of actions determine if there is a musical note, the process is represented in the Figure 3. In this report is detailed the collection data and the its processing, the musical output it is mentioned, nonetheless, it has not been implemented.

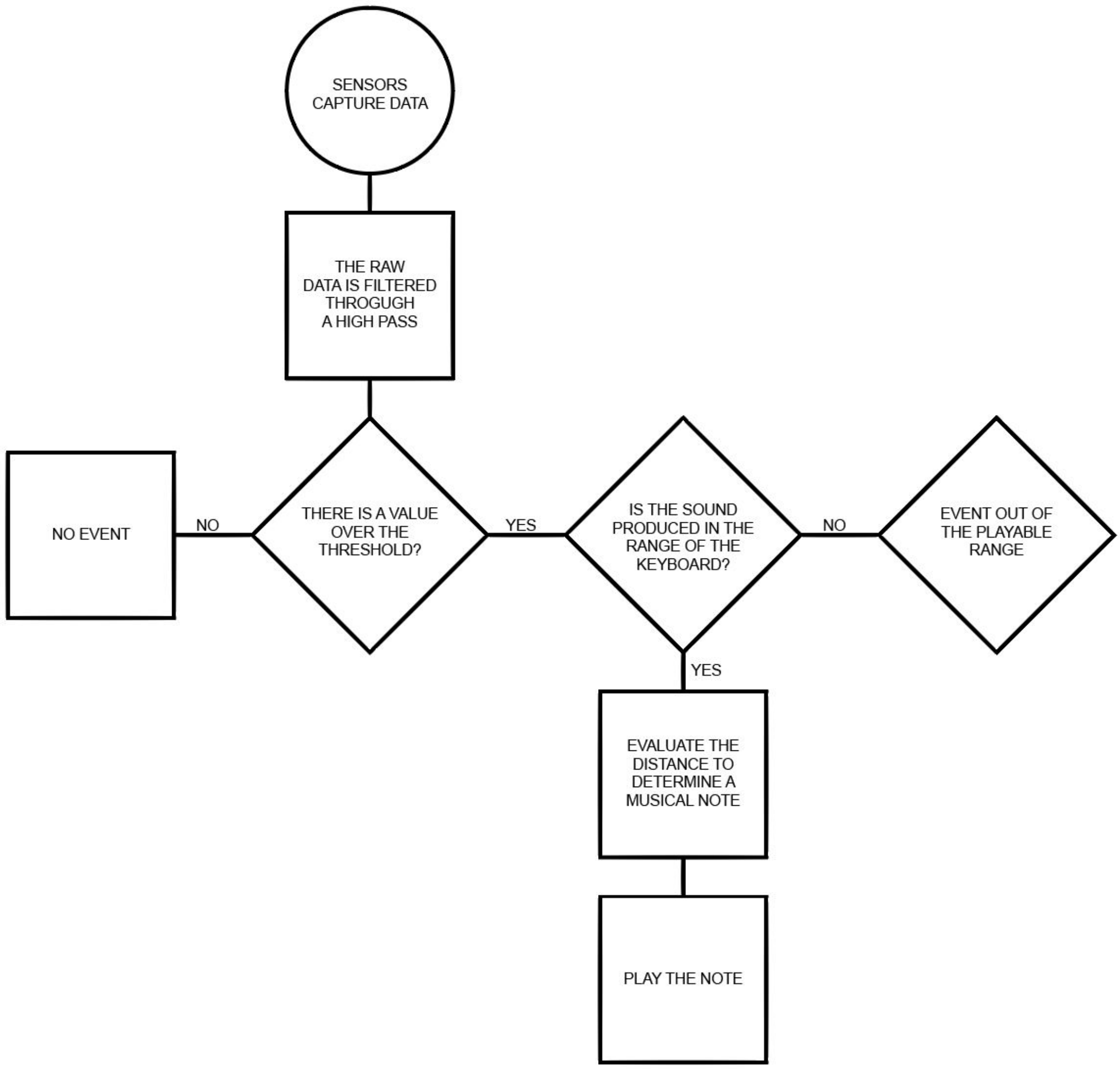


Figure 3 - Algorithm flow diagram

Important decisions in the design process have been made by evaluations and analysis, i.e. to determine the best sensor for detecting individual sounds it were analysed the behavior of two different sensors. By way of example, the Figure 4 displays a chart with the performance of the Sound Sensor and the Loudness Sensor plugged to the same boards under the same conditions; the results evidence that the Sound Sensor got data with greater ranges of values, hence, to choose particular events it might be easier. Similarly, the sound sensor appeared to recover the 0 value faster than the loudness sensor; under those circumstances, it was decided to use the sound sensor to identify the hits in the system, as well as recommend Marshal and Reeves (2015).

Table 1
Descriptives testing sound sensors

	SOUND SENSOR	LOUDNESS SENSOR
MEAN	60	132
MAXIMUM VALUE	780	347
MINIMUM VALUE	33	41
RANGE	747	306
STD. DEVIATION	56	25

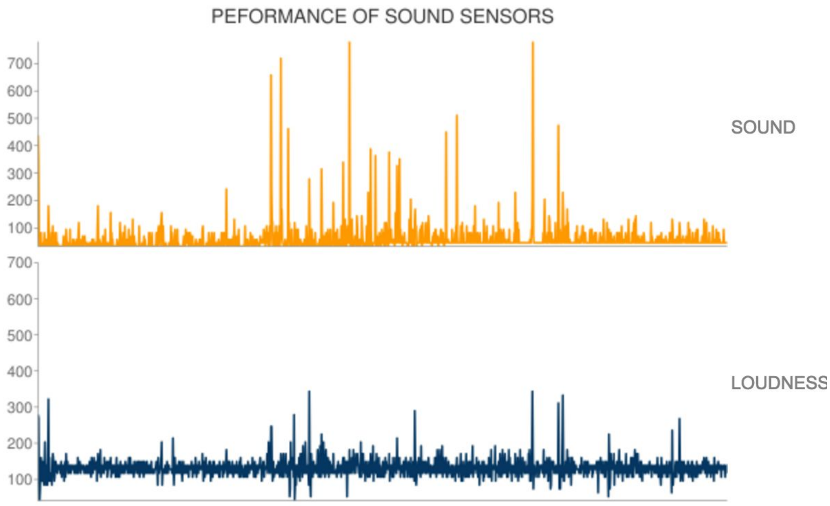
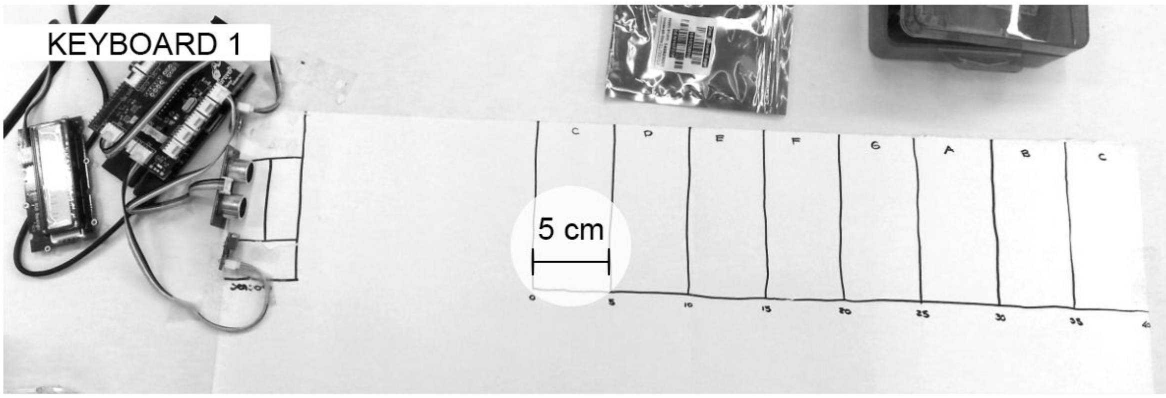


Figure 4 - Performance of sound sensors

Implementation

The artifact designed is able to combine the sound and range inputs in the process, likewise, it is ready to define if the stimuli correspond to a musical note. To make the process suitable has been necessary to polish both data collection process. On one hand, defining the suitable range for the virtual keyboard; on the other hand, cleaning the audio collection process.



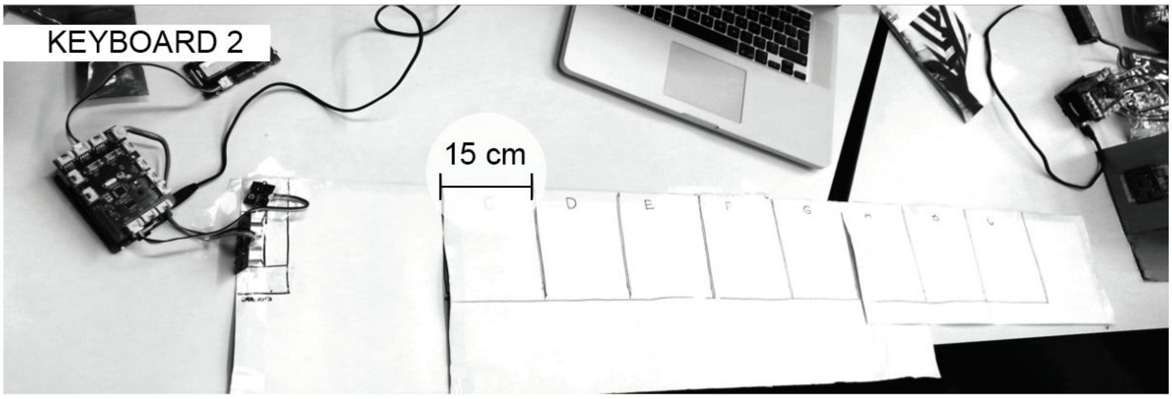


Figure 5 - keyboard's grid

To run the first prototype it was created a grid with a representation of the keyboard, and it was defined one octave from C to C that started in the 25 [cm], with a separation of 5 [cm] between each note. However, during the firsts evaluations it was observed a difficulty in pressing the virtual boxes without interfer in the adjacent box. The size defined appeared to be smaller than the width of the standard hands. Hence, a second grid with a greater range it was generated as is shown in the Figure 5. Nonetheless, in the code was created an operation to modify easily the start and end distances of the range, as well as the number of octaves of the virtual keyboard; in the Figure 6 is displayed a section of the code to illustrate the process.

```
start = 25
end = 125
octaves = 1
listnotes = ['C','D','E','F','G','A','B']

while True:
    try:

        distance =  ultrasonicRead(4)
        tvar = "n/a"

        if distance <= end and distance >= start:
                tvar = listnotes [(distance-start)/((end-start)/((octaves*7)))%7]
                grovelcd.setText (tvar)
                grovelcd.setRGB(156,156,156)
```

Figure 6 - Code applied to transform and use range data

To use the auditive information has been applied a set of filters on the raw data. Firstly, the input is processed with a high-pass filter to clean up noise and attenuate low frequency signals (Marshal et.al, 2015). Secondly, a threshold is applied to select the high-frequency events individually. Finally, the prominent events are filtered with the distance conditional to determine if the trend is generated by a hit in the keyboards range.

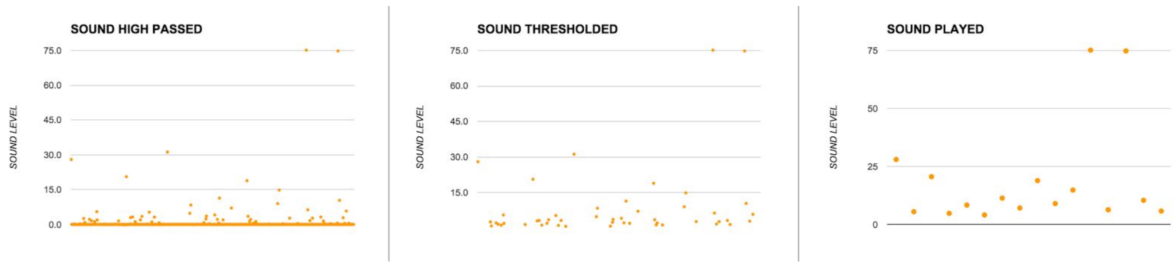


Figure 7 - Audio data captured in with different filters

In the Figure 8 is shown a part of the Python code that includes the sound filters that has been used in the algorithm, the original code was taken from Marshal et.al (2015).

```
filtOut = 0
constant = 0.1
lastValue = 0

threshold = 1.5

while True:
    try:

        value = grovepi.analogRead(0)
        filtOut = constant * (filtOut + value - lastValue)
        lastValue = value

        if filtOut > threshold:
```

Figure 8 - Code applied to transform and use audio data

Testing

Two tests were applied to evaluate the accuracy of the artifact in circumstances where the performance of both sensors were involved. Thus, the studies were placed in a laboratory to examine the relationship between distance and sound in controlled environments. To assess these conditions the keyboard was configured in a range of 105 [cm], in other words 7 notes with 15 [cm] of separation, and the Threshold was set in 1.5.

In the first study a musical pattern it was defined, a sequence of six notes should be payed in a particular order (E - E - E - C - E - G) with a separation of five seconds per note. In the Figure 9 are shown the results of three tests applied progressively under the same conditions.

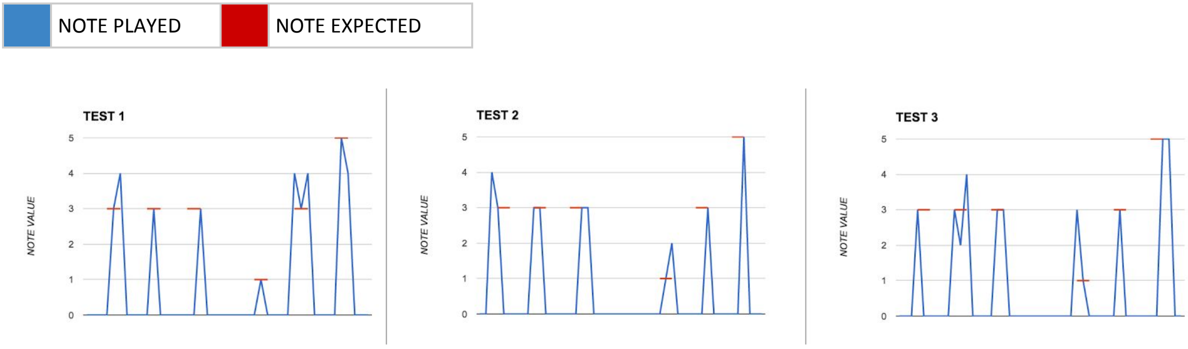


Figure 9 - First study, accuracy reproducing patterns

In the Figure 9 the numbers represent the values of the notes (1 is C, 2 is D, 3 is E, 4 is F, 5 is G). Therefore, it can be observed that all the events were collected by the system. To determine an event as true positives the value collected should coincide with the value expected, thus, If one event was played in the right moment with a different note value it was considered as false positive. During this evaluation the percentage of true positives was bigger (61.1%) than the percentage of false positives (38.9%); in the 85% of the cases when a note was identified as false positive the deviation was no more than 1 tone, the other 15% presented a deviation of 2 tones. Additionally, the data captured showed that the median of the number of events collected by hit was 2.

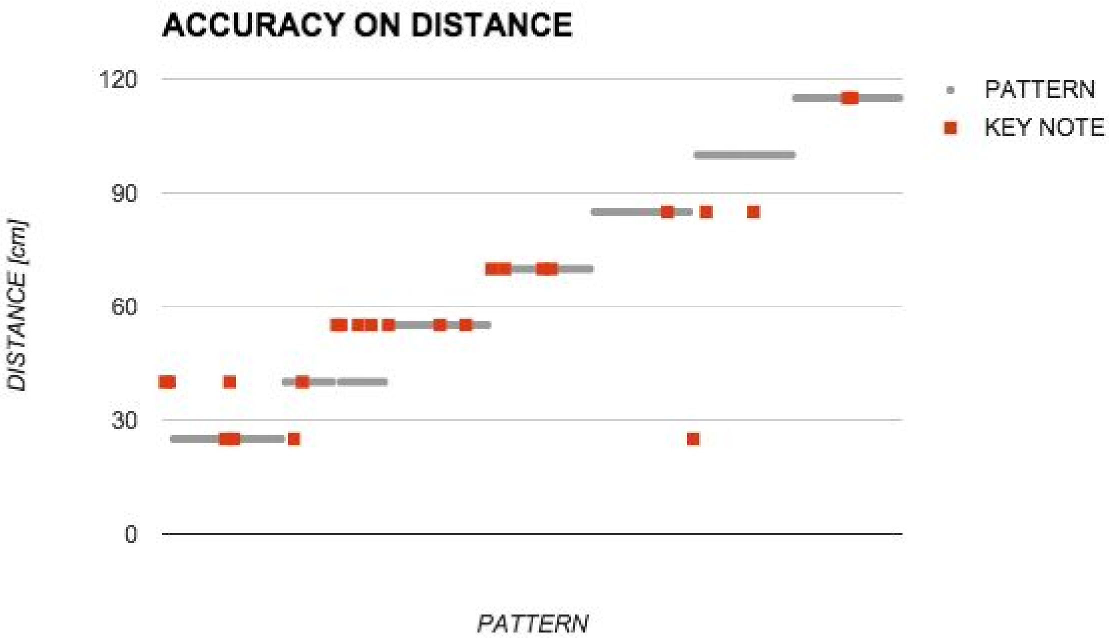


Figure 10 - Second study, accuracy in distance

The second study was defined as an evaluation of the recognition accuracy against the distance of the hit. In an interval of 5 seconds two events should be played in each segment.
Most of the notes were captured when the hit was produced close to the sensor. Thus, as is displayed in Figure 10 in the range of 0 - 60 [cm] the system recognized 20 events, however, some

events were collected several times as different inputs; between the range of 60 - 120 [cm] were identified 9 events. In the closest range all the notes expected were played, as well as other false negatives that were not expected; similarly, in the farthest range just half of the events were played in the expected value, the other half of events was not detected.

Table 2
Accuracy on distance

Range of 0 - 60 [cm]	
TRUE POSITIVE (WELL PLAYED)	FALSE POSITIVE (BAD PLAYED)
5	6
FALSE NEGATIVE (BAD REJECTED)	TRUE NEGATIVE (WELL REJECTED)
0	2

Range of 60 - 120 [cm]	
TRUE POSITIVE (WELL PLAYED)	FALSE POSITIVE (BAD PLAYED)
4	3
FALSE NEGATIVE (BAD REJECTED)	TRUE NEGATIVE (WELL REJECTED)
2	7

Critical reflection

To evaluate the efficiency of a digital musical instrument is necessary to take into account the factors around the Control Input Complexity as precision and mapping strategy (Jordà, 2005).

Using of sound sensors in Ubiquitous Computing artifacts is a big challenge to solve, the factors that modify the sonoric environment are infinite. Under those circumstances the system has been implemented and tested in a laboratory; notwithstanding, the number of people sharing space, the type of tasks executed around, the state of the sensor, among others, could modify the accuracy of the musical instrument. To correct those problems were applied a set of filters as a high pass to highlight individual events, a threshold to clean of useless events, as well as a match with the other sensors. Certainly, applying this improves to the algorithm it was enhanced the performance of the artifact, nonetheless, every change oppen an important group of variables to study deeply. As a consequence, it is recommended analyse the balance between range and sound sensors to correct the lack of recognizement of events when the hit is far of the sensors and the overstimulation when is near.

Dispense with a physical interface is the one of the most important innovations in the virtual keyboard, however, the Human Computer Interaction considerations force the designer to understand how to control other type of maps affordances and constraints to allow the communication between the artifact and the performer (Rehman, Stajano and Coulouris, 2002). The close relationship between form and function will determine the dimensionality of control and the degrees of freedom in the mapping strategy.

Finally, to continue the development of this artifact it is recommended to start designing the Musical Output Complexity based on the results of the evaluations to take advantage of the nature of the system, and return to assess the code based on a musical defined logic.

References

JORDÀ, Sergi, *Digital Lutherie Crafting musical computers for new musics' performance and improvisation*, Ph.D. thesis, University of Pompeu Fabra, 2005.

MARSHAL, Joseph and REEVES, Stuart, *Python and Sensors 2: Characterising and filtering sensor data* [online]. Nottingham: University of Nottingham, 2015 [viewed 24 Nov 2015]. Available from: moodle.nottingham.ac.uk/

REHMAN Kasim, STAJANO, Frank and COULOURIS, George, *Interfacing with the invisible computer*, ACM New York, pp. 213-216.

WEISER, Marc, *The world is not a desktop*, ACM Special Interest Group on Computer-Human Interaction, 1994, vol 1, pp. 7-8.

APPENDIX A

Instructions

1.- Plug the Sound Sensor in the port (A0)
2.- Plug the Ultrasonic Ranger Sensor in the port (D4)
3.- Locate the sensors in the same way as Figure 2
4.- Plug the LCD RGB Backlight in the port (I2C-1)
5.- Connect the devices in the Secure Shell
6.- Copy source_code.py in Raspberry PI
7.- Run the script across Python

APPENDIX B

In this Appendix are listed the .csv files that contain the data collected from the tests applied, as well as the source code in .py extension. The original files are included in the APPENDIX B folder.

- 11_03_sound_sensors.csv
- 11_17_reproducing_patterns_test_A.csv
- 11_17_reproducing_patterns_test_B.csv
- 11_17_reproducing_patterns_test_C.csv
- 12_01_accuracy_distance.csv
- 12_01_sound_filters.csv
- source_code.py