4.1. uTubo — development and application of an alternative digital musical instrument

Tiago Ângelo[1]
Óscar Rodrigues[2]
Nuno Peixoto de Pinho[3]
José Alberto Gomes[3]

Abstract
This article describes the development and use of *uTubo,* a sound device planned to be neither a musical toy nor a "serious" instrument. The device, built using the Arduino platform to synthesize sound and read gesture data from a few coupled sensors, allows the instrument player to interact with the sound generating unit mainly by twisting/bending a plastic tube. Furthermore it is also possible to invert this interaction by clicking a big button on the top of the casing, changing the way input gestures are used to produce sound, which can substantially alter the relationship between the instrumentalist and the device.
uTubo was one of the instruments built for the project *Sonópia*, which proposed to create a set of novel instruments and interfaces developed by *Digitópia Collective* - Casa da Música and *LAbMóvel - Gulbenkian Foundation,* during March 2013*. Sonópia* was part of *Ao Alcance de Todos,* meaning by the reach of all, which was a larger group of projects with artistic and social scopes, led by *Serviço Educativo da Casa da Música.* And for this purpose, *uTubo* was designed for no specific person or type of person, aspiring to suit a large range of players, from people with certain degrees of physical/mental impairments to children or even "serious" musicians.

Keywords: Alternative Musical Instruments, Arduino, Mozzi, Karplus-Strong.

Introduction

New electronic/digital musical instruments have flourished since the last couple decades, which might relate to the ubiquity of computers and their kindred, the democratization of music, knowledge commons and DIY communities, or "simply" by our need for different tools or our urge for unique or highly specific sound and music expressions. While some of these recent instruments still resemble some characteristics with pre-established acoustic instruments, others make the musical instrument taxonomy quite blurred and can even make the distinction between electronic and digital instruments seem confusing (Ângelo, 2012).

The instrument hereby presented, named *uTubo* — simply because its main interface relies in a tube which can be bent and distorted in order to produce sound, might be categorized

[1] Digitópia Collective, Casa da Música, Porto, Portugal.
[2] Digitópia Collective, Casa da Música, School of Music, Art and Performance, Porto, Portugal.
[3] Casa da Música, Catholic University of Porto, CITAR – Centre for Research in Science and Technology in Arts, Portugal.

as an alternative digital musical instrument[4], using the Miranda-Wanderley classification model (Miranda & Wanderley, 2006). It might be considered so as it resembles no particular characteristics from any previously established acoustic instrument, and its interface might not even relate to any other mainstream commercial instrument or controller.

The instrument interface is usually known as the component used by the instrumentalist to control the instrument, and it was taken as a great concern for the design of *uTubo*, as required by the demands and goals of the project *Sonópia - Ao Alcance de Todos,* undertook by *Digitópia Collective* and *LAbMóvel*.

Digitópia is a digital music platform, based at the concert hall *Casa da Música*[5] in Oporto, which encourages the act of listening, performance and musical creation. Based on digital tools, although not exclusively, *Digitópia* emphasizes collaborative musical creation, software design, music education and social inclusion, aiming to merge multicultural communities of performers, composers, curious and music lovers. *Digitópia* as a team, *Digitópia Collective*, consists of artists with strong ties to new technologies. In his work the collective expands on processes and models as diverse as designing digital instruments and other musical hardware, circuit-bending, exploring the relationship between image and sound, the practice of VJ's and DJ's, the digital medium or interactive digital systems.

Ao Alcance de Todos, created in 2007, is a week of performances, workshops and training sessions on the theme Music, Technology and Special Needs. Within this larger project of social and artistic dimension, conducted continuously year after year, *Digitópia* developed an artistic residency in 2013, named *Sonópia* (see Fig.1), targeting the development and construction of new or alternative instruments and interfaces to be used in the performances of *Ao Alcance de Todos.*

Figure 1 - Sonópia residency (from left to right: José Alberto Gomes, Diogo Tudela, João Menezes, Pedro Augusto, Simão Costa and Tiago Ângelo). Photo by José Alberto Gomes.

[4] The definition given by Miranda and Wanderley (2006) is "alternate gestural controller", although it seemed more appropriate here to classify uTubo as an instrument, it still fits into the class provided by the authors mentioned above. For a clear distinction between musical controller and musical instrument the reader might refer to Paine and Drummond (2009).

[5] http://www.casadamusica.com/

In the following chapters we will address the design and development stages, going from the input interface to the sound synthesis model, then addressing the interaction design in its own chapter, due to its contribution for the development of *uTubo*. And finally we will address the use and application of this instrument in both musical and social contexts.

Design and development

Creating a nouvelle instrument for the *Sonópia* project required a careful design process since the instrument was intended to be used by virtually anyone: from children to senior adults and the specially impaired, to professional musicians, amateurs or simply music lovers. This goal demanded an instrument that could be easy and fun to play as well of being capable to stand alongside other "serious" instruments in performance contexts, thus becoming a musical and socially inclusive tool. To cut a long story short, *uTubo* needed an adequate input interface to suit a large range of instrument players, an interesting sound generator and a relationship between these two that would promote the instrument's playability and engagement from the player point of view, avoiding the dullness that could emerge from an excessively simple and easy to play instrument.

The design and development processes were driven in a back-and-forward manner, taking into account the end-users' (potential) needs, instrument cost of fabrication and the short development time span of one week, required by the *Sonópia* project. Trying to fulfill both reductionist and holistic approaches, as digital musical instruments traverse a large amount of disciplines, such as electronics and physical computing, human-computer interaction, sound synthesis or music performance, just to name a few. Thinking of the instrument and designing it as not only a mere group of components that comprise the instrument, but also of all these components working together to form a higher entity emerging as the holistic concept of the musical instrument *uTubo*.

By addressing musical instruments in a reductionist approach, one can divide them into three functional components: input interface, sound generating unit and mappings — which dictate how the input interface and the sound unit relate (Miranda & Wanderley, 2006). Since mappings can play a very important role in the holistic design of the instrument, working as a kind of glue that brings basic components together into forming a system with higher complexity, they will be addressed separately in the next chapter. While in this chapter we will describe the hardware and software components chosen for the input interface and sound generating unit of *uTubo*, justifying whenever plausible the choices made during the design and development stages.

Interface design

The core concept and metaphor of *uTubo* orbited the idea of touching, bending and deforming sound through a plastic tube. And while initial designs used this tube both as a playing interface and as the instrument body (Fig. 2), which required the instrumentalist to hold the instrument with both hands, the final design required to present an easier way of playing the instrument, especially for players with reduced motor skills. So, for that purpose, the final design consisted in a box or case, which could be laid in the musician's lap or in a table, having the plastic tube and all other interfaces attached to that case (Fig. 3a and 3b).

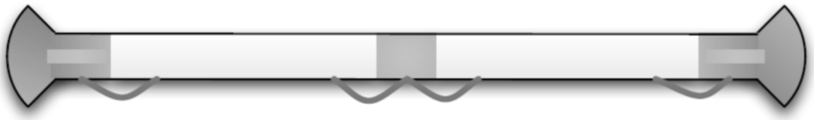

Figure 2 - *uTubo* initial draft

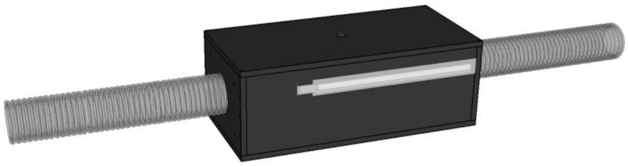

Figure 3a - *uTubo* final draft

Figure 3b - *uTubo* presented at Handmade Music, *Casa da Música*, Oporto, Portugal, 8 June 2013. Photo by João Messias/Casa da Música.

Another important concept around the design of *uTubo* was the possibility to change the instrument's behaviour (what it does in response to the player's gestures) through a simple gesture and interface — a big button on the top of the casing that could be pressed while playing the plastic tube at the same time. This ought to bring a bigger engagement between the instrumentalist and the instrument by surprising him with different instrument behaviours, as well as it could be used according to any special needs of the instrumentalist or even according to any compositional constraints or performance aesthetics.

And while these interfaces, the plastic tube and the big button, would remain as the central pieces of instrument control, there was also the will to add a continuous pitch control set through touch, that could potentially create more intricate musical phrases, in opposition to the tube and the big button, which could become very static across the pitch space.

Furthermore there was the ambition to develop an instrument which left behind personal computers, while still being able to develop it in a short time span of one week, given for the realization of *Sonópia*. This would result in a more independent and standalone instrument, and not just a musical controller that requires a connection to an expensive personal computer in order to become a musical instrument.

Tools and resources

One way of getting fast results in the development of *uTubo* was to use well established electronic prototyping platforms, such as the Arduino[6], Teensy[7] or similars. Which gave us a microcontroller that could run our code (reading sensor data to control synthesized sound) and an Integrated Development Environment (IDE) which gave us the language and tools to write our code and compile it into the microcontroller. Additionally there is a lot of support from DIY communities for these kind of prototyping platforms.

In order to avoid the dependency of a personal computer, a decision was made to implement all the code inside one microcontroller, thus using it for sound synthesis as well as to read gesture data and map it to the synthesis parameters. Using the Arduino platform, this could cost somewhere around 30 to 60 euros, which is much cheaper than any commercially available personal computer, thus becoming the chosen platform for the development of *uTubo.* Any other similar platform could theoretically be used with similar results, but Arduino was chosen instead for several reasons: I was already familiar with it (which obviously pended a lot in its favour although not exclusively), it has a very helpful online community and there was already some sound synthesis libraries developed for this platform (such as Mozzi[8], the sound synthesis library used for this project).

Regarding the input interface, *uTubo* required the use of two flex sensors (Fig. 4) placed inside the plastic tube, in order to read its deformation and to know when and how much the instrumentalist is bending and twisting the tube. And, for the continuous control of pitch the choice relied on a membrane touch-potentiometer (Fig. 4), which provided a 20-centimeter touch-sensitive strip, allowing to continuously set the instrument's pitch. Additionally there were a bunch of small electronic components (resistors, capacitors, etc.), nuts and bolts, wood to build the casing and a vacuum cleaner's plastic tube.

Sound synthesis model

Using microcontroller boards, such as the Arduino, to develop sound synthesizers obviously carries some disadvantages when compared to the use of personal computers (see Table I). Nowadays, it is certainly possible to obtain more intricate and complex synthesis models with personal computers, due to their processing power and storage capacity. And, although some microcontrollers, such as the ATmega2560[9], can still render some interesting sounds, it is necessary to carefully fill its small memory with efficient algorithms capable of providing more soundwise with less computational resources.

---

[6] http://arduino.cc/

[7] https://www.pjrc.com/teensy/

[8] Mozzi is an open-source sound synthesis library for Arduino, developed by Tim Barrass. Available at http://sensorium.github.io/Mozzi/

[9] http://www.atmel.com/devices/atmega2560.aspx

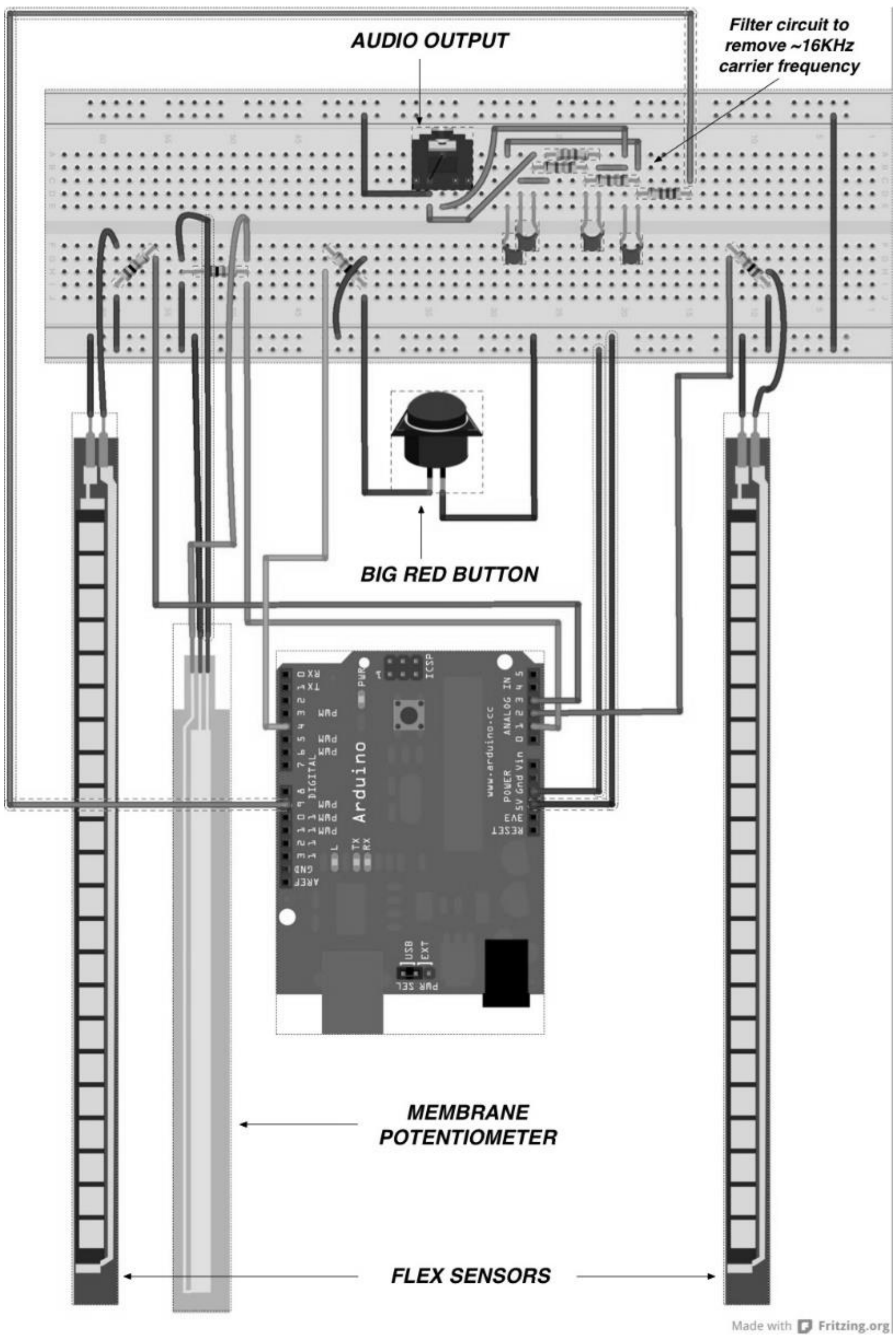

Figure 4 - *uTubo* bredboard schematics

Algorithms like frequency modulation (FM), additive synthesis or subtractive synthesis might be able to build complex tones but they do require some computational resources in order to perform the necessary arithmetics. FM can be a bit more efficient than additive synthesis, since it can achieve the same results with far less oscillators. Despite being possible to achieve complex tones with FM techniques it still requires multiplication arithmetics, which can be computationally expensive (Roads, 1996, p. 293). Other techniques based on the principle of delay lines or recirculating wavetables, such as the Karplus-Strong (KS) algorithm, developed by Kevin Karplus and Alex Strong, can be extremely efficient in terms of computational resources and can, nonetheless, synthesize enjoyable complex tones. These kind of algorithms were already known to run in 8-bit microprocessors with surprisingly good results. (Roads, 1996)

The Karplus-Strong algorithm is a simple physical modeling algorithm that aimed to simulate the implied physics of plucked strings. Although not fulfilling the entire physical variables and behaviours of a plucked string, the simple algorithm presented by Karplus and Strong achieved good results because it relied on the principle that the instrument's timbre should vary through time within the same sound event, mimicking the behaviour of traditional acoustic instruments.(Karplus & Strong, 1983)

The synthesis algorithm used for *uTubo* is an adaptation of the Karplus-Strong algorithm, and it had no intention of sounding like any pre-existing string instrument. In general terms it is still quite similar to the KS algorithm for plucked strings[10], but it has two delay lines instead of just one. So it is possible to think of it, in analogy to conventional plucked string instruments, as having two slightly dependent strings that are plucked simultaneously. In this case, plucking a string is analogous to sending a noise burst (or impulse) to the delay lines, which are then fed back, simulating the decay of a plucked string. Each delay line (or string if you wish to think of it that way) has a variable size, which corresponds to a variable pitch. And by adding both delay lines with different pitches it is possible to obtain slightly more complex sounds, since the sum of both sounds, coming from two different delay lines, can give rise to peaks in certain frequencies of the sound spectrum, while attenuating others. Thus behaving as a computationally cheap filter that could bring some timbral complexity to *uTubo*.

By using Mozzi library for Arduino one as the benefit of having an adequate language for sound synthesis, with most of the components common to it, such as wavetables and oscillators, envelopes, delays, etc. Leaving the instrument developer with more time to experiment on sound synthesis, by freeing him of the burdensome of microcontroller programming for audio. As seen in Fig. 5, the implementation of the adapted KS algorithm using Mozzi is quite straightforward, making more time available to experiment different ways of controlling and playing this little synthesizer, which could contribute to a more pleasant and enjoyable instrument from both the player and the listener point of view.

```
////////////////////////////////////////////////AUDIO//////////////////////////////////////
int updateAudio() {
  // IMPULSE
  impulse = (pinkNoise.next()*impulseGain);

  // DELAY BLOCK W/ FEEDBACK
  leftSig = leftDelay.next(impulse + leftSig, leftDelaySize);
  rightSig = rightDelay.next(impulse + rightSig, rightDelaySize);

  //OUTPUT
  return leftSig+rightSig;
}
```

Figure 5 - Karplus-Strong algorithm implementation using Arduino and Mozzi

[10] Besides the string model, Kevin Karplus and Alex Strong also devised a model for drum sound synthesis. (Karplus & Strong, 1983)

Interaction strategies

Mapping in musical instrument design is what connects variables from the input interface to variables of the sound generator, and it plays a very important role in the instrument's identity, as stated by Hunt, Wanderley and Paradis (Hunt, Wanderley & Paradis, 2002). In acoustic instruments, where the input interface and the sound generator are coupled together, such as the strings in a violin, which are at the same time sound generators and input interfaces for instrument control, where mappings are defined by the laws of physics. On the other hand, electronic and digital musical instruments have separated input interfaces and sound generators, and mappings don't occur naturally by any law, but instead are defined by the instrument's designer (or *luthier*) or even configured and changed by the instrumentalist before or during a performance.

Although the input interface and the sound generator might be easily recognized and identified at a first glance, the mapping layer is usually hidden under code, and is more difficult to identify, especially if mappings are not in a direct one-to-one relationship. Nonetheless, this layer bears an enormous potential on the instrument outcome from the instrumentalist perspective, as well as from the listeners perspective. (Hunt, Wanderley & Paradis, 2002) In this regard it was essential to carefully design *uTubo*'s mapping layer, as it would contribute for *Sonópia* goals, as much or even more than the input interface and the sound generator.

Musical control and interaction design

The control premisses of musical instruments lie in parameters such as rhythm, dynamics, pitch and timbre, as the core of instrument control and expression[11]. Since these are only conceptual semantic parameters, it is necessary to design the link (mappings) between sound synthesis parameters from the sound generator to parameters of the input interface, in order to evidence these musical controls. Therefore, the intended musical controls of *uTubo* were designed to deliver the following results:

- rhythm — one-shot and repeated events with variable time intervals;
- dynamics — besides silent and playing, dynamics are not directly controlled;
- pitch — (monophonic) continuous pitch control;
- timbre — detune and distortion.

Reminding *Sonópia*'s premisses, the musical control of *uTubo* needed to be simple enough for children, unexperienced musicians or anyone with certain physical or mental impairments to be able to play this instrument in a musical context, such as those performances realized for *Ao Alcance de Todos*. But it also should be fun and musical enough to be performed by virtually anyone, even by skilled musicians.

Although the input interface as a huge weight on the accessibility and ergonomics of the instrument, and the sound generator over sonic properties, parameter mappings define the instrument's behaviour and characteristics in response to the player's input. Thus exhibiting a focus area to fulfill *Sonópia*'s premisses, potentially making *uTubo* something between a musical toy and a "serious" instrument.

[11] For a longer discussion over digital musical instrument expressivity the reader can refer to (Dobrian & Koppelman, 2006)

Parameter mapping

Parameter mapping in digital musical instruments can undertake several topologies, describing the input-output relationship in terms of connection points. The simplest topology is a one-to-one mapping, where one input parameter is mapped to another output parameter. But it is also possible to map N input parameters to one output parameter, one input to M outputs and N inputs to M outputs. (Miranda & Wanderley, 2006) We know beforehand, thanks to Hunt et. al, that one-to-one mappings are usually too simple and straightforward for musical expression. And, as shown by Hunt, Wanderley and Paradis (2002), mapping topologies that are more complex usually drive better results in terms of musical expression, making the instrument less predictable but also more enjoyable to learn and play.

uTubo's interface has basically four sensor inputs: one big red button on the top of the casing, one touch membrane potentiometer and two flex sensors (one on the left and the other on the right of the casing) both attached inside the plastic tube. While the sound generator has basically six parameters: a noise impulse with trigger, attack, decay and duration controls plus two delay lines with variable delay sizes.

Acknowledging facts over mapping topologies, one had to design a somewhat complex mapping layer that wouldn't just care with explicit one-to-one controls, but instead could merge them into more complex topologies, making some controls inseparable from each other, which could contribute for an instrument with a stronger personality and behaviour rather than just a controller for individual sound synthesis parameters.

Parameter mappings created for the musical control of *uTubo* can be described as follows:

- **Rhythm** - in terms of rhythmic control one had to develop ways of making one-shot events as well as repeated events at variable time intervals defined by the instrumentalist, making it possible to apply *accelerandi* and *rallentandi* to this stream of events. One way of achieving this is to use Mozzi's *EventDelay* class[12] to generate this stream, where the time interval between events is controlled simultaneously through both flex sensors as well as the membrane sensor (see *metroTime* variable in Fig. 6). So, in order to play a continuous stream of events the player has to keep *uTubo* out of its resting state, while to play a single one-shot note the user needs to take *uTubo* out if its resting state and leave/take it again to its resting state after the one-shot sound event has been played/heard. In other words, the repeated stream of events is always active when *uTubo* is out of its resting state and it will stop plucking any more events once it reaches the resting state again. Additionally, it is also possible to control the note duration, although not directly, as it depends on the time interval of the metronomic note generator, which in itself depends on data from both flex sensors, as well as the currently selected pitch (see the *impulseDuration* function in Fig. 7).
- **Dynamics** – in *uTubo* there is no direct control over the dynamics of sound events, nor there is any volume control. Direct volume control is binary: it's either on (playing) or off (silent). This is defined by a mechanism that sets the instrument resting state by pressing the big red button on the top of the casing. When this button is pressed, values of both flex sensors are memorized and set as the resting state. So, whenever one of these sensors surpasses the memorized value by a certain threshold (see Fig. 6) the instrument leaves its resting state and starts its

[12] One of Mozzi's caveats is that it disables Arduino delay() function, so the EventDelay class had to be used.

rhythmic processes (see rhythmic description above). Furthermore it is possible to bring a feeling of louder dynamics by "stressing" the plastic tube, that is, by getting it more and more distant from its resting point, where the density of events increases as well as the distortion of the output sound. Additionally, the control of dynamics might occur indirectly, since both delay lines, with possibly different sizes (or different pitches), are summed at the output, and are fed back to the system, eventually causing some phase cancellations and boosts that will make the generated sound decay faster or a bit more slowly.

- **Pitch -** pitch and rhythm are connected because the membrane touch potentiometer, the sensor used for the continuous pitch control, is also used to define the time interval of the event stream (Fig. 6). So, higher pitches will also generate faster rhythms and lower pitches will generate slower rhythms. Besides controlling the pitch with the membrane potentiometer it is also possible to detune it slightly using the flex sensors inside the plastic tube.
- **Timbre -** timbre control in *uTubo*, such as the dynamics control, had a very simple implementation. Since the KS algorithm emulates a plucked string, where most of the timbral control comes from the attack portion of the sound event, analogously the way a string is plucked will influence its timbre over time. Since the impulse pluck was programmed using a wavetable filled with random values (*1/f* noise ratio, which corresponds to pink noise), and not actually a noise oscillator, one had to change its phase whenever a note was "plucked" in order to avoid having the same impulse over and over again for all plucked notes. A cheap way of setting a random phase to the impulse generator was to read an analog pin from the Arduino with nothing connected to it (see *setPhase* function on Fig. 7, this problem is also addressed by Karplus & Strong (1983)). The other timbral control comes from the sum of two slightly detuned delay-lines, which basically works as a filter, since summing two different sounds will attenuate some frequencies and enhance others, while detuning each delay-line is done through each dedicated flex sensor.

```
void mapping() {   //SENSOR->SYNTH MAPPING

  // SET METRO
  if(leftFlex < restLFlex + REST_THRESH && leftFlex > restLFlex - REST_THRESH
      && rightFlex < restRFlex + REST_THRESH && rightFlex > restRFlex - REST_THRESH )
  {
    metro = false;
  } else { metro = true; }

  // SET METRO TIME
  metroTime = ((leftFlex * rightFlex * membrane)>>THIRTEEN) + METRO_MIN;

  // SET DELAY TIME (wich corresponds to pitch)
  leftDelaySize = membrane + leftFlex + MIN_DELAY;
  rightDelaySize = membrane + rightFlex + MIN_DELAY;
}
```

Figure 6 - *uTubo* parameter mappings (Arduino code)

```
void setImpulse() { // SOURCE IMPULSE

  pinkNoise.setPhase(analogRead(A6)<<3); // cheap way of avoiding a noticeable loop (0~8192 noise)

  //IMPULSE ENVELOPE
  if(impulseDuration.ready() == true && metro == true) {
    impulseEnvelope.start(RAMP, RAMP); // impulse attack-decay values
    impulseDuration.start(metroTime); // impulse duration
  }
  impulseGain = impulseEnvelope.next();
}
```

Figure 7 - *uTubo* noise impulse (Arduino code)

Evaluation and musical output

The examination and evaluation of digital musical instruments has been already covered by a large group of researchers as seen in (Ângelo, 2012, pg. 34). Thus it is out of the scope of this article to present a new evaluation model or even to extend such models. Instead we will look at the seven-axis classification model of Birnbaum et. al (2005) to inspect and observe *uTubo*. This model provides a dimensional space representation of instrument characteristics and it was chosen as it builds onto previously presented classification and evaluation models comprehensively, providing a visual representation of the musical device. Furthermore we will look at *uTubo*'s musical output diversity as proposed by Jordà (2004).

The evaluation model presented by Birnbaum et. al relies on seven axis of representation: required expertise, role of sound, distribution in space, inter-actors, feedback modalities, degrees of freedom, and musical control[13]. In the next section *uTubo*'s design intentions are evaluated according to the dimension space proposed by Birnbaum et. al.

- Required Expertise: *uTubo* was designed to have a low entry level of expertise;
- Role of Sound - designed to have an artistic/expressive sound role;
- Distribution in space: by default *uTubo* has a small space distribution, although it can be extended (as mentioned in chapter 4.1);
- Inter-actors: by default *uTubo* was meant to be played by one instrumentalist, although it allows more than one player to interact with its input interface, as well as in the situation described in chapter 4.1;
- Feedback Modalities: there are only two feedback modalities in *uTubo*: the produced sound and the physical state of the plastic tube;
- Degrees of Freedom: input controls available in *uTubo* are moderate, and most of them are intertwined in 1-to-M mappings;
- Musical Control: the levels of musical control present in *uTubo* rely, for the most part, on the note level and on the control of musical processes (such as the ostinato created by repeating sound events).

In terms of musical output diversity, *uTubo* could be empirically evaluated as follows[14]:

- Macro-diversity (Stylistic diversity): medium to high Mac-D, meaning that it is not an instrument with a very high specificity over musical style nor it is completely

[13] For a complete description of the seven axis of classification please refer to (Birnbaum et. al, 2005)
[14] For a complete description of the output diversity classifications used the reader should refer to (Jordà, 2004)

adaptable to most musical styles and genres, although it can certainly fit into more than one style;

- **Mid-diversity (Performance diversity):** medium Med-D, meaning that two different performances (of different pieces) played with *uTubo* can produce moderately different results;
- **Micro-diversity (Performance nuances):** low to medium Mic-D, meaning that two performances (of the same piece) could bring only slightly different results, especially if compared to most acoustic instruments, such as brass or wind instruments.

Musical and social application

The developed instrument presented in this paper was then used in two different child projects of *Ao Alcance de Todos*: *Algodão Doce* (meaning cotton candy) and *Descobertas Sonoras* (meaning sonic discoveries). While the first project was dedicated to children the other was dedicated to the specially impaired. In the next subchapters we will describe and report the use and application of *uTubo* in both projects.

"Algodão doce"

uTubo was used at the *Algodão Doce* concert (see Fig. 8) at *Casa da Música* - Portugal (18 and 19 May 2013) and SESC[15] São Paulo - Brazil (12 October 2013). The concert was destined to children aged from 3 months to 5 years old and revolves around a group of characters trapped inside cotton candy. One of the characters, a wizard, casts many spells that invariably result in music. These include making a grand acoustic piano play on its own, with moving keys, enchanting a ball that makes sound as it moves, and bringing an old vacuum cleaner to life - *uTubo*. In *Algodão Doce*, both the action (that is, the storytelling) and the playing took place in very close proximity to the audience, which was encouraged to participate. The "stage" consisted of three to four small places, scattered around the room, full of stage props.

[15] http://www.sescsp.org.br/

Figure 8 - *Algodão Doce* concert. Photo by João Messias – Casa da Música

Although not originally designed for the concert, *uTubo* suits its goals perfectly: it's a seemingly normal object capable of producing otherworldly sounds, especially when connected to effects processors - as it was. That non-apparent relation between its looks and its sound was, in fact, crucial, as at first sight any member of the audience would think that it was just another stage prop. But when it was played it had an almost magical effect, contributing to the show's peculiar and surreal character.

The interface itself was also very useful, for two reasons: first, it allowed the instrument to be played by very young children, who obviously lack the coordination required to play somewhat more sophisticated instruments - they could just hit the instrument's tube, and hear a sound response in real time; second, due to it's standby features, it could remain silent for the majority of the concert, when it was not needed. And it was reliable enough to assure us that it would play when the tube left its standby position. Furthermore, adding effects pedals to *uTubo*'s output allowed it to be played by up to three people at the same time - one at the tube, other at the pitch control, and a third one at the effects, which was great for the family interaction these kinds of concerts look for.

From the composer's point of view, and keeping in mind this particular concert, our goal was to create a comfortable place for children to freely explore rhythm through the simple interaction previously described. It was, therefore, important, to keep some level of musical activity on the background, not only to take the pressure off the participants, leaving them more comfortable and avoiding possible silences, but also to create contrast - the almost unchanging background versus the clear, rhythmical attacks of *uTubo*.

"Descobertas sonoras"

The workshop *Descobertas Sonoras* (see Fig. 9) was part of *Ao Alcance de Todos* festival, which took place between 25 and 27 March 2013. During three days two different types of workshops were organized: smaller workshops of one hour and a half were led during the first day (March 25), while longer workshops were planned for the two upcoming days, which intended to prepare a presentation/concert open to public on 27 March.

Figure 9 – *Descobertas Sonoras* rehearsal. Photo by João Messias – Casa da Música

During the three days of workshops, several individuals with specific special needs, such as reduced mobility, blindness and cognitive limitations, participated in *Descobertas Sonoras.* In order to alleviate the arduous execution of traditional musical instruments from these participants, the workshop instructors/tutors had a set of electronic/digital instruments at their disposal. These instruments were previously developed during the *Sonópia* project, which brought a group of instruments aiming to ease and promote the participants' attention and performance. While presenting some of these instruments to the participants, alongside with other traditional instruments such as the piano or the electric bass, *uTubo* quickly got their attention as it presented a different timbre associated with a different mode of execution. And, during the workshops, it truly facilitated the development of activities set by the workshop instructors. In musical terms, *uTubo* was used to create long pedal tones as well as smaller rhythmic cells. Furthermore, some of the participants had the physical/cognitive ability to change the instrument's register (pitch). This musical gesture took place by two different means:

- by instructing the participants to change the pitch, through conducting gestures achieved by the tutors;
- or by giving total freedom to the interpreter for manipulating this musical resource in a free and creative manner.

One resource that was rarely used was the inversion of *uTubo*'s behaviour by pressing the big red button in the top of the casing. This was mostly due to some physical and cognitive impairments of the participants, and in most cases instructors would help participants press the button.

Descobertas Sonoras counted with the participation of 15 individuals from *Centro Integrado de Apoio à Deficiência* (CIAD – *Misericórdia do Porto*[16]) during 26 and 27 March 2013. This group of individuals, suffered from blindness, physical and/or cognitive impairments, and presented different degrees of disability, from moderate to severe, partially or totally dependent from third-person support. These two days were programmed having in mind the final presentation/concert. So, after getting the participants acquainted to all the available instruments, *uTubo* was ascribed to a pair of participants that held moderate cognitive impairments. And, in both cases, they were able to manage and fulfill the proposed tasks for the public performance in a quite satisfactory manner.

uTubo was set on stage on 27 March and it had the starting point of the musical concert, by representing the sounds of a thunderstorm. While in the intermediate section interpreters would improvise with *uTubo*, with the help of instructors to set register/pitch changes using the membrane potentiometer. And, in the final section of the performance, *uTubo* instrumentalists played a rhythmic cell in the musical *ostinato* form. During the concert, all conductor gestures provided to play *uTubo* were devised to simulate the gesture that the interpreter would need to apply in order to play the required sonic/musical events.

Conclusion

The possibility of developing personal synthesizers, synth modules and other electronic/digital musical instruments have been growing in the last couple years, mainly due to large knowledge sharing communities over these topics and the accessibility of electronic components and sensors allied to the use of microprocessor prototyping platforms such as Arduino. Which have become powerful enough to use in real-time sound synthesis. Furthermore, the selected programming tools, Arduino and Mozzi, proved to be a great resource for the development of stand-alone synthesizers.

Developing *uTubo* required some efforts to be made, so that the instrument wouldn't lend itself useful only for the mentioned performances (in 4.1 and 4.2, which weren't known before the development of *uTubo*), but could in fact be used in a variety of different musical and social contexts. Although, some of the programmed functionalities of the instrument tended to be used less often, such as the big red button and the membrane potentiometer, largely because they were difficult to activate/manipulate by some players, the overall capabilities and specifications of this instrument served the purpose of the project *Ao Alcance de Todos* quite well.

The use of different sensors and components, such as ultrasound sensors to replace the membrane potentiometer or a softer button for the button in the top of the casing, for example, could have rendered more control capabilities to this instrument, given the context of *Algodão Doce* and *Descobertas Sonoras*. Nonetheless it proved to be both musically useful from the composer's point of view and satisfactorily engaging from the instrumentalist point of view as well as surprising from the audience point of view.

[16] http://www.scmp.pt/

References

Ângelo, T. (2012). *Open Instruments: Framework para Desenvolvimento e Performance de Instrumentos Musicais Digitais em MaxMSP*. Faculdade de Engenharia da Universidade do Porto, Portugal. Retrieved from http://hdl.handle.net/10216/65229

Birnbaum, D., Fiebrink, R., Malloch, J., & Wanderley, M. (2005). Towards a Dimension Space for Musical Devices. In *NIME '05: Proceedings of the 2005 conference on New Interfaces for Musical Expression* (pp. 192–195). Vancouver, BC, Canada.

Dobrian, C., & Koppelman, D. (2006). The'E'in NIME: musical expression with new computer interfaces. In *NIME '06: Proceedings of the 2006 conference on New Interfaces for Musical Expression* (pp. 277–282). Paris: IRCAM—Centre Pompidou. Retrieved from http://dl.acm.org/citation.cfm?id=1142283

Hunt, A., Wanderley, M., & Paradis, M. (2002). The Importance of Parameter Mapping in Electronic Instrument Design. In *NIME '02: Proceedings of the 2002 conference on New interfaces for musical expression* (pp. 429–440). National University of Singapore, Singapore.

Jordà, S. (2004). Digital Instruments and Players : Part II – Diversity, Freedom and Control. In *Proceedings of the International Computer Music Conference*. San Francisco, CA: International Computer Music Association.

Karplus, K., & Strong, A. (1983). Digital Synthesis of and Plucked-String Timbres. *Computer Music Journal*, *7*(2), 43–55.

Miranda, E., & Wanderley, M. (2006). *New digital musical instruments: control and interaction beyond the keyboard*. Middleton, Wisconsin: A-R Editions, Inc.

Paine, G., & Drummond, J. (2009). Developing an Ontology of New Interfaces for Realtime Electronic Music Performance. In *Proc. EMS 09*. Retrieved from http://www.ems-network.org/ems09/papers/paine.pdf

Roads, C. (1996). *The Computer Music Tutorial*. Cambridge, Massachusetts: MIT Press.