INTEGRATING A SIMPLE SENSOR-BASED TESTBED IN SIGNAL PROCESSING COURSES

A Recipe to Design an Electronic Instrument

Belén Torrente, Adriana Dapena, Santiago J. Barro, Julio Bregáins and Carlos J. Escudero
Department of Electronics and Systems, University of A Coruña, A Coruña, Spain

Keywords: Learning applications, Supervised projects, Hardware and software integration, Music synthesis, Fourier series.

Abstract: The widespread idea that traditional classroom lectures is not the best teaching approach has motivated educators to continuously seek new ways to encourage students towards active involmenent. Studies show that constructivist learners tend to explore the concepts covered in laboratory activities, resulting in a richer understanding. In this sense, we will propose to integrate simple sensor-based electronic musical instruments in signal processing courses to help students understand basic concepts, while making them to interact with a real systems that integrates, at once, software and hardware components.

1 INTRODUCTION

Our experience of more than 10 years teaching signal processing related courses as part of the Computer Engineering degree shows that students have difficulties for understanding the mathematical foundations of this engineering area. This limitation can be mitigated by proposing laboratory activities consisting of computer simulations performed using software tools such as Matlab/Simulink, or similar (Blanchet and Charbit, 2006). The experiments are designed not only to write code for a specific method, but also to test the students' comprehension of the materials covered in previous theoretical lectures. However, we have observed that this kind of laboratory activity does not allow us to exploit the "rich world" of signal processing because students tend to regard laboratory classes as an activity where the most important task is to write code.

Over the past few years, we have invested great effort in setting the possibility of taking advantage of integrating hardware and software in our laboratories (García-Naya et al., 2010). This allows students to face a series of problems that they do not encounter in computer simulations like, for instance, hardware selection, faults and integration of hardware and software.

In this paper we present a "recipe" to create custom sensor-based electronic musical instruments conformed by three elements: sensors, an acquisition device and a synthesis environment. Sensors are devices capable of turning the measurement of a physical quantity into an electrical signal. Their outputs are processed by the acquisition device to provide a specific list of sound parameters needed by the synthesis algorithm. Instead of using comercial products (like LabView (National Instruments, 2010)), we have preferred to use open-source software adapted to our courses needed. In particular, the selected acquisition device has been the Arduino board for several reasons: it is an open-source project, it uses a C++ like programming language (called Wiring), includes a built-in ADC, and has USB connectivity. Finally, a synthesis algorithm is implemented in a Personal Computer (PC), generating music in accordance with the sensors' parameters.

The modular structure of our testbed allows access to it at different levels of abstraction. This allows the system to be used with different purposes:

- Explaining basic concepts of signal processing.
- Developing supervised projects where students must integrate software and hardware.
- In addition, although this paper focuses on education, the ideas presented here have many other applications like electronic music instrument design, generation of sound banks and didactic museum activities,

This paper explains the two phases needed to develop such electronic instruments:

Torrente B., Dapena A., J. Barro S., Bregáins J. and J. Escudero C..
INTEGRATING A SIMPLE SENSOR-BASED TESTBED IN SIGNAL PROCESSING COURSES - A Recipe to Design an Electronic Instrument.
DOI: 10.5220/0003311700870092
In *Proceedings of the 3rd International Conference on Computer Supported Education* (CSEDU-2011), pages 87-92
ISBN: 978-989-8425-50-8

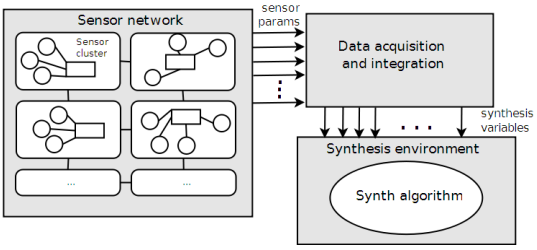

Figure 1: Overall System.

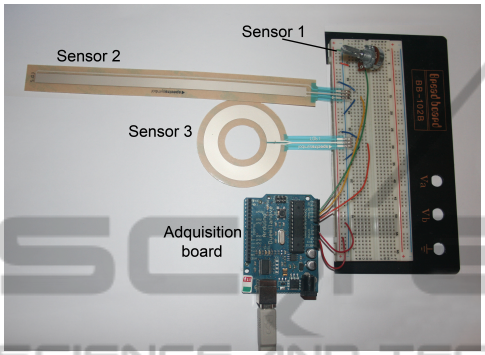

Figure 2: Our testbed.

- Component selection. Section 2 offers a general description of the system structure by presenting the parts involved.
- Component integration by developing a graphical application (Section 3).

In the 2010-2011 academic year, we have used electronic instruments to explain the ideas behind the Fourier series. Section 4 shows the results obtained from this experience.

2 GENERAL SYSTEM OVERVIEW

The idea behind our laboratory activity is that students must interact with real music instruments obtained by integrating hardware equipment and software programming. Figure 1 shows the organisation of our testbed and Figure 2 shows a picture of a specific configuration. Three main parts can be distinguished from left to right in Figure 1. The first elements we encounter are the *sensors*, which are capable of providing context information of a varying nature. Next, data must be collected by an *acquisition device*, which will typically do some preprocessing on the sensor information. Finally, the output of the acquisition device is the input to the *computational system* (for instance, a PC) in charge of a synthesis algorithm that should be able to provide two kinds of outputs: a text-based descriptive model, and the real-time sound output of the synthesis algorithms, both using the parameters provided by the sensor integration.

2.1 Sensors

A sensor is a device which responds to a certain stimulus (physical quantity), converting it into an electrical signal, that electronic devices can use (Wilson, 2005). They can be classified according to different criteria: the technology used (inductive, capacitive, resistive, piezoelectric, optical, etc.), the kind of feature measured (acceleration, distance, position, rotation, etc.).

The first step to design the student's instrument is to make an appropriate selection of the sensors involved. This must be done taking into consideration several parameters (Wilson, 2005): sensibility (smallest variation that can be detected), dynamic range (signals outside this range are expected to cause unacceptably large inaccuracy), etc. In particular, for our purpose, it is very attractive to consider touch sensors (potentiometers) because, by using them, the correct functioning of the overall system is easier to check. That is, the numeric result can almost be calculated at a guess. As an example, the system shown in Figure 2 uses three potentiometers, each of them controlling an individual parameter of the synthesis algorithm described in Section 2.3.

2.2 Acquisition Device

The sensing devices are connected to a circuit that captures the input values with a certain periodicity and transforms signals into a specific list of sound parameters required by the synthesis algorithm. For this task, we have selected Arduino (Arduino project, 2010).

The Arduino platform is composed by a single-board microcontroller and a software developing environment. The Arduino board is based on an 8-bit Atmel AVR microcontroller with other additional components, such as input pins. Its layout makes it easy for the CPU board to be connected to a wide range of modules.

The Arduino IDE (Integrated Drive Electronics), written in Java, which makes it cross-platform, is an application derived from the IDE made for the Wiring project and the Processing programming language. It was initially designed for teaching purposes, so it is perfect for newcomers to software development. Using the so-called Arduino2Max patch it is possible to program our instruments visually by using

Table 1: Popular Synthesisers.

Model	Year	Synthesis method
Moog Minimoog	1971	Subtractive
Yamaha DX7	1983	FM
Casio CZ-101	1984	Phase distortion
Kawai K5	1987	Additive
Roland D50	1987	Sample-based
Korg Wavestation	1990	Wavetable
Korg Prophecy	1996	Physical model

MAX/MSP objects (as will be explained later in this paper). Arduino2Max is also responsible for sending the data in real time so that users can see the values at each instant.

Communications in Arduino are performed via the serial port. From the existing approaches, the one adopted by Arduino2Max is based on the serial communications library called SimpleMessageSystem (Arduino SimpleMessageSystem, 2010). It provides three different operations: one for clearing a pin and the other two for reading data as an ASCII string (one for digital pins and the other one for analog pins). SimpleMessageSystem allows us to send, receive and parse characters, integers and lists between Arduino and other applications, such as MAX/MSP or Pure Data.

2.3 Synthesis Techniques

Sound synthesis is the process of generating sounds artificially (Pinch, 2002). Nowadays, the most popular kind of synthesis uses sounds sampled from an acoustic instrument (sample synthesis, granular synthesis). Other types use cyclic patterns (wavetable synthesis), physical mathematical models, or basic operations using simple waves (additive, subtractive and FM synthesis) (Russ, 2008). Table 1 provides a list of some popular synthesisers, and which method they have used.

Additive synthesis is directly related to the Fourier Series which describes a signal as a combination of elementary trigonometric functions. It is the tool most extensively applied in both signal analysis and synthesis and indeed finds application in many areas (Vaseghi, 2007) such as telecommunications, signal processing (filtering, etc.), music processing , signal coding and feature extraction. However, this concept proves difficult to understand by many undergraduate students (Friedberg, 2001) because the mathematical derivations are tedious and the students are not able to see the practical applications. In our opinion, programming and testing an additive synthesiser is a good opportunity to catch students' attention towards Fourier Series.

Fourier's work can be summarised as follows. Denote by $f(x)$ a function of the real variable x. This function is usually taken to be periodic, of period 2π, i.e., $f(x+2\pi) = f(x)$, for all real numbers x. Fourier's work states that this function can be expressed as an infinite sum, or series, of simpler periodic functions with period 2π. More formally, an integrable function on $[-\pi,\pi]$ denoted by $f(x)$ can be expressed as an infinite sum of sine and cosine functions on the same interval, given by

$$f(x) = \frac{a_0}{2} + \sum_{k=1}^{k_{max}} [a_k \cos(kx) + b_k \sin(nx)] \quad (1)$$

where k_{max} denotes the number of harmonics, and a_n and b_n are the amplitudes of the components of the Fourier series,

$$a_n = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x) \cos(nx) dx \quad (2)$$

$$b_n = \frac{1}{\pi} \int_{-\pi}^{\pi} f(x) \sin(nx) dx \quad (3)$$

From these equations, it is a good exercise for students to determine the expression of the Fourier series in terms of the complex exponentials $e^{jnx} = \cos(nx) + j\sin(nx)$ (Oppenheim et al., 1998). This development is beyond the scope of our paper.

It may be noted that the sound created by any musical instrument can be expressed in terms of the Fourier series (Vaseghi, 2007). This is the basis of additive synthesis (also called Fourier synthesis), which consists of adding together simple sinusoidal waves to create a more complex timbre (Russ, 2008). More rigorously, any periodic sound in the discrete time domain can be synthesised as follows

$$\begin{aligned} x(n) &= \frac{a_0[n]}{2} + \sum_{k=1}^{k_{max}} \left[a_k[n] \cos\left(\frac{2\pi k f_0}{F_s} n\right) \right. \\ &+ \left. b_k[n] \sin\left(\frac{2\pi k f_0}{F_s} n\right) \right] \end{aligned} \quad (4)$$

where $a_k[n]$ and $b_k[n]$ are the series coefficients, f_0 is the fundamental frequency and F_s is the sampling frequency.

3 COMPONENT INTEGRATION

The production of sounds can be done by giving control over the frequency and amplitude of each individual harmonic. For instance, in our testbed (Figure 2) we have three potentiometers for controlling timbre

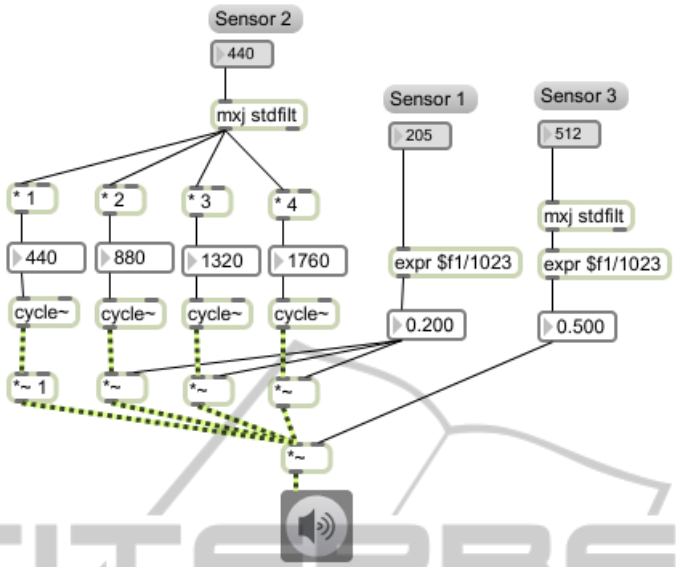

Figure 3: Example of additive synthesis (fundamental frequency is 440 Hz).

```
import com.cycling74.max.*;
import java.math.BigInteger;
public class stdfilt extends MaxObject {

 public stdfilt(Atom[] args) {
   declareInlets(
       new int[]{DataTypes.ALL});
   declareOutlets(
       new int[]{DataTypes.ANYTHING});
 }

 private float num1,num2,num3,num4 = 0;

 public void inlet(float f){
   num4 = num3;
   num3 = num2;
   num2 = num1;
   num1 = f;
   double mean = (num1+num2+num3+num4)/4;
   double std2 = (num1-mean)*(num1-mean)+
                 (num2-mean)*(num2-mean)+
                 (num3-mean)*(num3-mean)+
                 (num4-mean)*(num4-mean);
   if (Math.sqrt(std2)<10)
     outlet(0,f);
   else
     outlet(0,0);
   }
 }
}
```

Figure 4: JAVA external component *mxj stdfilt*.

colour, pitch and amplitude. Timbre is related to the weight of the components of the Fourier series ($a_k[n]$ and $b_k[n]$), pitch corresponds to the fundamental frequency f_0, and by amplitude we mean the weight of the fundamental frequency component, $a_o[n]$.

Using the Arduino2Max patch, the additive synthesiser is programed by adding and combining boxes (objects) as seen in Figure 3. From top to bottom, we can see the following elements:

- (mxj stdfilt) is a Java external component that we have coded to avoid the undesirable behaviour of contact sensors that produce erroneous outputs when the sensor is not pressed (see Figure 4). Basically, this function computes the variance with respect to the mean of the sensor's values in four consecutive samples, and compares it with a threshold. If the value is lower than the threshold, the sensor is pressed. On the contrary, a high variation implies that the sensor is giving an erroneous output.
- Multiplication boxes ($(*1)$, $(*2)$, $(*3)$ and $(*4)$) are used to multiply the value of the fundamental frequency indicated in Sensor 2 by a scalar quantity (1, 2, 3 and 4), i.e., the outputs are the harmonic frequencies.
- (cycle ~) generates a wave with the input frequency.
- (* ~) multiplies the amplitude of a wave times the value indicate by Sensor 2.
- (expr $ f1/1023) is used to convert the values into the range $(0,1)$

```
<instrument id='ins1' synth='add' f0='440'>
 <overtones>
   <overtone freq='1' phase='0' amp='1'/>
   <overtone freq='2' phase='0' amp='0.2'/>
   <overtone freq='3' phase='0' amp='0.2'/>
   <overtone freq='4' phase='0' amp='0.2'/>
 </overtones>
 <envelope length='200' finite='true'>
  ...
 </envelope>
</instrument>
```

Figure 5: An instrument described by means of XML.

The example in Figure 3 corresponds to a fundamental frequency $f_0 = 440$ Hz introduced by pressing sensor 2, and an amplitude of 0.2 for the Fourier coefficients $b_k = a_k = 0.2$, $k = 1,2,3$ (sensor 1). The amplitude of the composite signal is fixed by sensor 3 to 0.5.

The application provides two kinds of output: the sound output of the synthesis algorithms and a text-based descriptive model. The sound allows the students test in real-time the effect that the Fourier Series parameters have in the synthesised signal.

The text-based descriptive model consists in enriching alternative way of storing music information. This has been done using XML. It seems a good solution because XML documents can be parsed without difficulty, thus simplifying a possible port to other platforms. For our purpose, we have studied the most widely supported format, called MusicXML. It is mainly aimed at constructing staff notation representations, that could be played back by any MIDI enabled device. Due to the nature of MIDI, that rests upon a finite number of sound banks, referred to as instruments, and there is no room for synthesis parameters. We have therefore opted for a different XML format tailored to suit this project's specifications, including dynamic timbre changes, as well as pitch and velocity.

Figure 5 shows part of the XML representation of the additive synthesis parameters given in Figure 3. For the sake of simplicity, envelope configuration has been omitted. When the performance starts, the <instrument> tag will be generated. The instrument description will be followed by a series of tags describing any change that takes place to the original parameters, containing a timestamp.

4 AN EXPERIENCE

In the current academic year (2010-2011), we have used electronic instruments to explain the applicability of the Fourier Series. The experience has consisted in explaining how to generate music using the additive synthesis method described in subsection 2.3. Change the sensors parameters, the student have tested the effect of varying the Foruier serie parameters: fundamental frequency, amplitude of harmonic and amplitude of the final wave.

We have evaluated the interest of our students in this experience by means of a survey containing three questions:

- Question 1. Did you known that the Fourier Series can be used to generate music?
- Question 2. Would like you create your own electronic instrument?
- Question 3. What do you think about the presentation?

There were 12 students in the course. Figure 6 shows the results of the survey. It seems that the students have positively evaluated the fact of using an electronic instrument to illustrate the concept of the Fourier Series. For this reason, in the following academic years, we will propose a laboratory activity where students must design their own electronic music instruments.

5 CONCLUSIONS AND FURTHER WORK

In this paper, we have explained the steps followed to build a sensor-based electronic instrument by integrating the open-source Arduino tools and low cost sensors. The system that has been implemented is characterized by its modularity and its innovative focus towards dynamic timbre features. We hope that its utilization in courses of signal processing can help the students to understand some important concepts, like the Fourier Series, because the students can experiment directly with hardware and software which is always more enriching that mere computer simulations. Besides, no proprietary software is needed, and the hardware used has a very low cost.

Some improvements on the work presented here are possible. First, implementing synthesis techniques other than additive synthesis is one of the goals envisioned by the authors in the short-term. Second, testing other kinds of sensors, such as gyroscopes or artificial vision could be a source of much innovation with regards to the user-system interaction. Besides, using wireless technologies (such as Bluetooth) can make the interaction more natural. Finally, we have

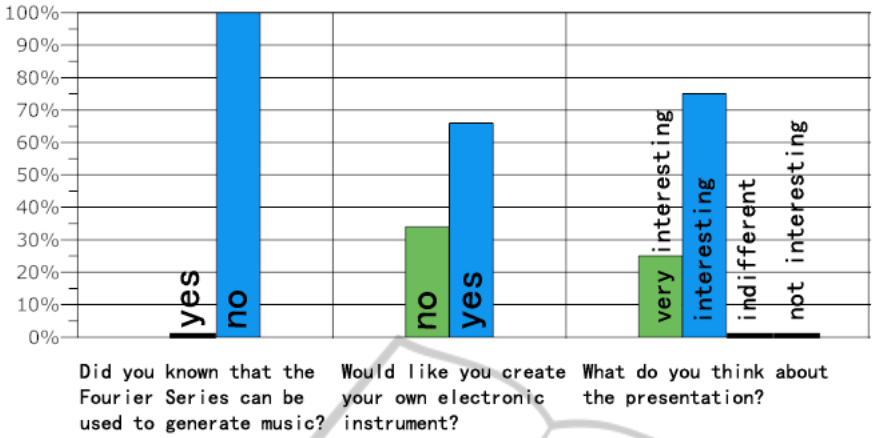

Figure 6: Survey results.

found that Arduino's 10-bit ADC may be not accurate enough. Even if the 1024 value resolution is more than adequate for certain parameters (e.g., velocity), in some cases it could fall short in representing a rich pitch range. For instance, such measurement-frequency mapping would have steps that are too big for a glissando. Further improvements could be made in this direction, using a different ADC.

ACKNOWLEDGEMENTS

This work was supported by Xunta de Galicia (grants numbers 10TIC105003PR and 10TIC003CT), Ministerio de Ciencia e Innovación of Spain (IPT-020000-2010-35).

REFERENCES

Arduino project (2010). *http://www.arduino.cc*.

Arduino SimpleMessageSystem (2010). *http://www.arduino.cc/playground/Code/SimpleMessageSystem*.

Blanchet, G. and Charbit, M. (2006). *Digital Signal and Image Processing using MATLAB*. Wiley.

Friedberg, S. (2001). *Teaching Mathematical in Colleges and Universities: Cases Studies for Today's Classroom*. American Mathematical Society.

García-Naya, J. A., Castro, P. M., González-López, M., and Dapena, A. (2010). Testbed-assisted learning for digital communications courses. *Computer Applications in Engineering Education*, pages 1–11.

National Instruments (2010). *Laboratory Virtual Instrumentation Engineering Workbenc (LabVIEW)*.

Oppenheim, A., Willsky, A., and Nawab, S. H. (1998). *Signal and Systems*. Prentice Hall.

Pinch, T. (2002). *Emulating Sound. What Synthesizers can and can't do: Explorations In the Social Construction of Sound*. Science Studies, Cornell University.

Russ, M. (2008). *Sound Synthesis and Sampling*. Oxford, Focal Press.

Vaseghi, S. V. (2007). *Multimedia Signal Processing*. Wiley.

Wilson, J. S. (2005). *Sensor Technology: Handbook*. Elsevier.