AUDIO PHYSICAL COMPUTING

Andrea Valle
CIRMA - Università di Torino
andrea.valle@unito.it

ABSTRACT

The paper describes an approach to the control of electromechanical devices for musical purposes (mainly, DC motors and solenoids) using audio signals. The proposed approach can be named "audio physical computing", i.e. physical computing oriented towards sound generation by means of audio signals. The approach has its origin in a previous physical computing project dedicated to music generation, the Rumentarium Project, that used microcontrollers as the main computing hardware interface. First, some general aspect of physical computing are discussed and the Rumentarium project is introduced. Then, a reconsideration of the technical setup of the Rumentarium is developed, and the audio physical computing approach is considered as a possible replacement for microcontrollers. Finally, a music work is described, in order to provide a real life example of audio physical computing.

1. INTRODUCTION: PHYSICAL COMPUTING AND SOUND

The use of computers in music is typically associated respectively to algorithmic composition, that is, computational approaches to the organization of musical form, and to sound generation, that is, generation of digital audio signals as the sonic output of a computer-based musical system. Rather, a less evident possibility concerns the use of a computer for the generation of acoustic sounds, in order to regain a specific acoustic physicality in output. With respect to such a goal, a still new but now firmly established perspective is opened by "physical computing", the terms meaning "computation with physical objects" [1], [2]. Physical computing fosters the idea that computation can be taken outside standard computers and embedded into physical objects. The key element for developing physical computing are microcontrollers, that is, computation units packed in small-sized circuit boards, with I/O facilities allowing to connect sensors and actuators. Musicians are considered to be the first to practice physical computing [1]. In particular, a long even if often underground experimental tradition of analog electronic music has worked intensively on hardware hacking with the aim of coupling control information from electric signals, performer gestures and physical output [3]. In the digital reprise of such a perspective, microcontrollers can play a pivotal role. In particular, as microcontrollers can drive electromechanical devices, physical objects can be involved as final, acoustic sources of sound generation processes. In this way, it is possible to create "acoustic computer music", that is, a music entirely controlled by computational means, but in which sounds are generated from acoustic bodies [1] .

2. THE RUMENTARIUM PROJECT

The design and realization of the Rumentarium project [5] move from these assumptions. The Rumentarium is a set of handmade percussive instruments ("sound bodies"), made of heterogeneous resonators that are acoustically excited by DC motors. The motors are controlled via computer through microcontrollers. In the Rumentarium, the design and production of sound bodies is inspired by sustainable design [6]. The name "Rumentarium" originates from *rumenta*, in Northern Italian meaning "rubbish, junk". The design embraces an ecological perspective based on the reuse of common objects. Many practices around the world have traditionally developed specific attitudes towards the "refabrication" of objects as a normal way of shaping and reshaping the semiotic status of material culture [7]. According to refabrication, in the Rumentarium the DC motors are scavenged from discarded electronics (CD and DVD players, mobile phones, toys) and they can be extended by adding parts of various materials (plastic, wood, metal), thus implementing different modes of excitation (percussion/friction). Resonators are assembled from a huge variety of recycled/reused materials: e.g. pipe tobacco boxes, glass bowls, broken cymbals, kitchen pans. The resulting sound bodies are generally assembled via metal wires, glue, soldering and they can include Lego parts. Figure 1 shows a version of Rumentarium including 24 sound bodies, installed at Share Festival, Torino, 2009. Rumentarium couples the ecologic perspective with an investigation into digital control of physical objects, as the sound bodies are entirely computationally controlled. Very naturally, it turned to microcontrollers as the hardware interface between the computer and the motors. In particular, Arduino Diecimila and Duemilanove boards were used, as Arduino [8], being inexpensive, open source, easy to program, is a *de facto* standard in physical computing. In Figure 1, Arduinos are contained in the plastic boxes with loudspeaker connectors. One of the main ideas in Rumentarium is to use a high-level software as a control interface. The whole

[1] An analogous perspective on the digital control of acoustic sound production is in [4].

Figure 1. Rumentarium installed at Share Festival, Torino, 2009.

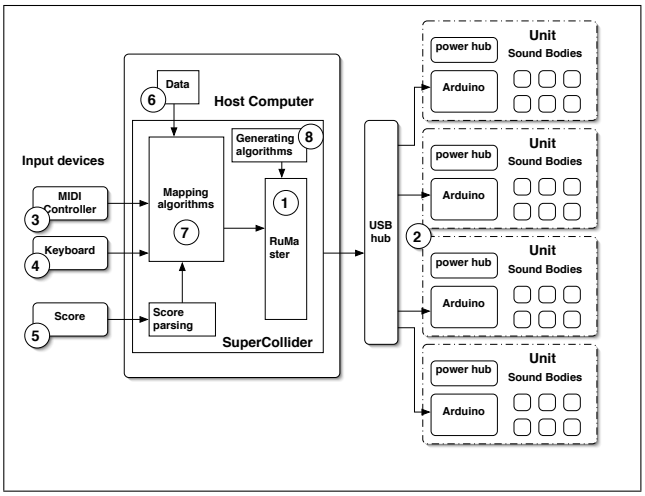

Figure 2. Rumentarium hardware/software setup.

software interface is written in the SuperCollider language [9], that summarizes features common to other general and audio-specific programming languages (e.g. respectively Smalltalk and Csound), and at the same time allows to generate programmatically complex GUIs. While a general programming language (e.g. Java) could have been chosen for controlling the boards, SuperCollider offers native audio DSP capabilities, useful for audio/musical application: as an example, in this way, through the analysis of audio signals performed by SuperCollider, Rumentarium can be seamlessly programmed to react to external sound sources (e.g. to speech). In addition, MIDI protocol is natively supported, as it is of common usage for musical devices (e.g. control surfaces). On the top left of Figure 1, a MIDI controller and the main computer to which is connected are visible. Rumentarium has been used extensively live, both as an autonomous sound installation and as a performing instrument (Palermo, Spazio Orioles, September 2009; Torino, Festival Share, December 2009; Milano, Festival Audiovisiva, May 2010; Roma, Riunione di Condominio, June 2010). It can be heard on AMP2's *Hopeful Monster* album, in the *Musica Improvvisa* box set by Die Schachtel [10], and two other albums are in press.

3. CONTROL PIPELINE: FEATURES AND ISSUES

While the computational control is a major (and aesthetically satisfying) feature in the Rumentarium, yet the overall technological pipeline has proven to be in some cases unstable and hard to debug, because of the many layers involved. Figure 2 shows the general structure of the Rumentarium, including both hardware and software elements (the latter specifically developed in SuperCollider for the project).

On each Arduino (2), the ports for PWM (pulse width modulation, where motors are attached) are indexed 3, 5, 6, 9, 10, 11, and the range of available values, to be converted in voltage, is [0, 255]. Each port is connected to the DC motor of a sound body through a Darlington transistor (following the basic design by [1]) that delivers current to a motor (through a power hub). The main software component is the RuMaster application (1), that acts as the software layer toward the physical output (microcontrollers and sound bodies). A RuMaster instance allows to treat a set of e.g. four different Arduino microcontrollers (2) as a unique abstract device with 24 abstract ports indexed from 0 to 23, to be sent values in the range [0, 1]. On this abstract software layer it is possible to easily build mapping strategies [5]. Such strategies can be related to data coming in real time from external inputs, such as MIDI gestural controllers (3) or from the keyboard (4). Data can also be gathered in non real-time from external sources such as a handmade graphical score (5), or from digital information (6). In all these cases, mapping algorithms (7) are needed to specify a semantics in terms of Rumentarium's behavior. It is indeed also possible to directly generate control data for the RuMaster (8). While Figure 2 offers a high-level, hence blurred, view of the communication protocol, Figure 3 shows the information flow and the traversed hardware/software layers, both on computer (I) and microcontroller (II). The presented software configuration dates back to 2008-09, and the situation has partially changed since then. Still, this multilayered software configuration can be at the origin of different issues, related to different layers, as each software layer depends on different third-party developers (or communities of developers), and the integration of the layers is exclusively a feature of the final project. First of all, in order to be accessed from the operating system, Arduino boards needed platform-specific drivers (3). Low level software components, as the ones required to interface with serial port, are tightly dependent on operating system: an OS update can easily break their functionality. The driver issue is a complex one, and not by chance the new Arduino Uno [2] tries to solve it by presenting the operating system a generic HID interface. Then, in order to drive the Arduino boards, SimpleMessageSystem (SMS) was used, a third-party library that forces Arduino to listen to the USB port, where it can receive instructions from the host computer (4). The library is now obsolete and the actual way to communicate in real-time with Arduino controllers through the USB port is no more

[2] `http://arduino.cc/en/Main/ArduinoBoardUno`

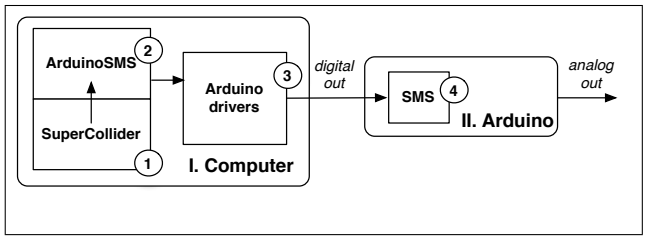

Figure 3. Hardware and software layers in the Rumentarium project.

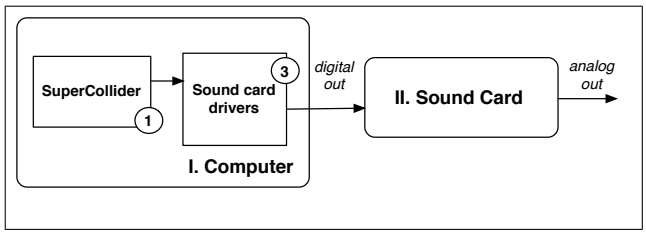

Figure 4. A typical hardware/software audio setup.

the SimpleMessageSystem library but the CmdMessanger one (that in the meantime has superseded an intermediate step, the Messenger one [3]. Finally, SuperCollider offered a third-party class, ArduinoSMS (2), as the interface –on the the SuperCollider side (1)– to the SMS library –to be loaded on the Arduino side. As evident from the previous discussion, even in a short timespan, such a situation has brought up relevant software compatibility issues. Software updates are not synchronized among layers and they can happen at different development rates. As an example, being SMS now almost obsolete, then the ArduinoSMS component on the SuperCollider side should be replaced with a new one. While this situation is indeed unavoidable in complex hardware/software systems, where maintenance routine is a major task, nonetheless it is quite complex to be handled by the composer/performer, even if s/he is acquainted to computer programming. Other issues are not related to software but depend on hardware. Instability in powering can lead to malfunction in the board. Data transfer rate and management on the board can be problematic: in the Rumentarium version involving up to 24 different sound bodies through 4 Arduinos, the number of control messages per second sent through the USB ports easily grows to a value that microcontrollers seemed not to be able to cope with. This became more apparent while using a MIDI controller, e.g. while turning a knob: in that case, a fast series of MIDI messages fires, each of them triggering a message to the controllers, easily getting beyond the maximum allowed.

4. AUDIO PHYSICAL COMPUTING: AN APPROACH

In order to solve, at least partially, all these issues, a possible new approach has been developed. The main mission for microcontrollers is embedding computation into autonomous physical objects. Besides, they provide input/output conversion from/to analog domain. As an example, in Rumentarium the microcontrollers simply act as digital-to-analog converters (DACs). Taking exclusively into account the conversion task, both in input and output, standard sound cards perform intuitively much better than a microcontroller. From the hardware point of view, sound cards are designed to receive in input and generate in output electric audio signals with minimized distortion. At least starting from the prosumer level, they are interfaced to operating system through reliable drivers and they can easily manage many parallel channels of audio [4]. Apart from sound card drivers, the only other software layer in such a configuration is audio software: available ones are many, and can fit various user's needs and approaches. As an example, SuperCollider is indeed a highly specialized software for audio synthesis and processing. Thus, comparing Figure 3 with Figure 4, layer 3 is substantially no more an issue, and layers 2 and 4 can be eliminated.

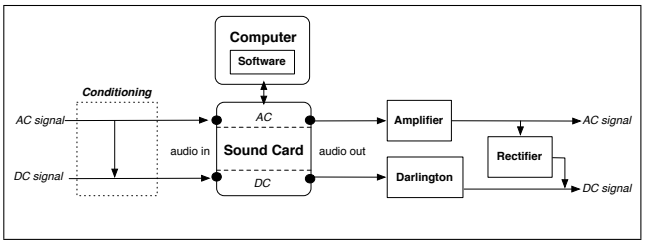

Figure 5. Audio physical computing chain.

While sound cards are faster and more reliable if compared to microcontrollers, they are indeed much more expensive, up to a factor of 10 for analog in/out. As noted by O'Sullivan and Igoe: "Musicians are the pioneers of physical computing. They have solved lots of problems of getting physical gestures into electronic form", but, "typically, their solutions are high-level and expensive" ([1], p. 354). While this holds true in a general physical computing context, sound cards are basic hardware tools for a musician, so they do not require an extra investment. On the other hand, while microcontrollers (or, better, the serial interfaces on which they are mounted) generally have a very limited bandwidth, the use of multichannel audio can really improve the input/output performance, as signals are efficiently computed at audio sample rate. An overall setup for input/output of audio signals to be used as control signals in a physical computing scenario is depicted in Figure 5. Such a configuration allows to implement "audio physical computing", the term referring to physical computing based on audio signals for sensing and controlling physical objects. Figure 5 must be considered as a first draft of manyfold possible solutions, to be finetuned empirically. A first distinction must be made between AC-coupled an DC-coupled sound cards. The first ones accept only AC signals in input and output, that is, standard audio signals. Most low-level sound cards are AC-coupled. DC-coupled sound cards are able to handle

[3] See in general `http://arduino.cc/playground/Code/`

[4] Indeed, this does not guarantee that interface problems are automatically solved by using an audio card: audio drivers can require some kind of optimization (e.g. setting the optimum buffer size, sampling rate, etc.) in order to maximize their performance on a particular system and hardware installation can be a complex issue to manage.

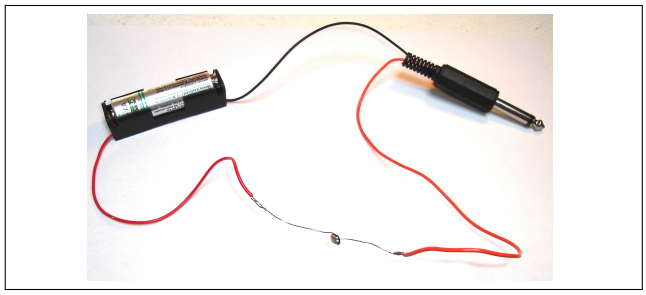

Figure 6. A photocell sensor as audio input.

not only AC signals, but also DC signals both in input and output, and thus represent a higher level subset (from the prosumer level) of AC-coupled ones. Here we just sketch some ideas on the input side, while focusing later on the output side, in order to provide a possible replacement for Rumentarium setup. A plethora of sensors can be connected to the sound card [5], provided a different kind and degree of "signal conditioning", that is, an analog manipulation stage on the signal so that it meets the requirements of analog-to-digital conversion [12] [13]. As conditioning strategies depend on each sensor's features, here we just sketch some possible approaches. By the way, most solutions in [1] can be directly used or easily adapted to work with sound cards. For what concerns input signals, indeed microphones (including piezo contact ones) do not require conditioning, as they can be directly connected to AC ins. Some sensors (e.g. typically accelerometers) are able to generate DC signals that have enough voltage to be connected directly to a DC input: in negative case they must be previously amplified. Passive sensors can be fed into a DC input after connecting them to a voltage source. Figure 6 shows a photo cell acting as a passive resistance between a power source of 1.5 volt battery and the sound card's DC in (here to be connected by means of a standard male 1/4 inch jack plug). The sound card's phantom powered input can be useful in order to increase signal gain. It is also possible, even if not straightforward, to connect a DC signal to the AC input of the sound card by using an inverter that converts DC to AC. An inverter can be implemented by means of a Voltage Control Oscillator (VCO) based on a function generator integrated circuit (IC), that is, a special-purpose oscillator used to produce sine, square, or triangle waveforms (such as the NE555). The signal frequency/amplitude of an oscillator can be varied by the external sensor. The resulting (frequency modulated) AC signal can be connected to the AC input of the sound card. A pitch/amplitude tracking software algorithm running on the host computer then allows to reconstruct the modulating DC signal quantity from the carrier AC signal. Indeed, once digitized, the signal can be processed (e.g. smoothed, sampled, filtered, analyzed etc) by means of usual, well-known DSP techniques.

On the output side, analog signals can be used as control signals for various applications. DC-coupled sound cards are able to directly output analog DC signals. These signals can be compared to voltage provided by Arduino outs and used similarly. As an example, they can be used to feed the Darlington transistor that has been previously discussed in relation to motor driving in the Rumentarium setup. An analogous technique is used in the Volta software by MOTU, a virtual instrument plug-in that turns the sound card into a voltage control interface [6]. Volta generates and sends DC signals via the sound card to voltage-controlled analog synths, thus allowing the user to reach a digital control over analog hardware. In turn, the audio signal output by the analog device can be connected in a feedback loop to the sound card input and received by Volta. Through this feedback loop, Volta is capable of advanced features such as autocalibration on analog synths. Volta is indeed a "hybrid control system" in which the computer generates digital control functions to be converted into voltages feeding the control input of synthesizer modules [14]. Thus, audio physical computing can be thought as a generalized hybrid system strategy.

A second option is based on AC signals (i.e. usual audio signals). In this case, an audio amplifier is needed after the sound card in order to ensure enough power, usually in order to drive loudspeakers. While this means a further increase in terms of invested money, a multichannel power amplifier is again a very general component in an audio/music studio setup, that can be literally used for decades. Furthermore, inexpensive stereo amplifiers are available in assembly kit from electronic resellers. The resulting output configuration, made of computer, sound card, amplifier, is indeed very stable and can be implemented without depending on specific hardware models or technologies. Moreover, in the context of physical computing, loudspeakers can be modified in order to exploit them as sources of mechanical force [3], e.g. to make vibrate different surfaces. Loudspeakers are a straightforward means to convert audio information into mechanical force. In the perspective of a generalized audio physical computing framework, an easy and effective solution in order to convert AC into DC signals is to add a rectifier as a further element to the chain, after the amplifier. The rectifier converts alternating to direct current by mirroring the negative part of the electric signal in the positive domain (as in the absolute value in the numerical domain). It can be made up of a single diode component with two inlet (from AC) and two outlets (to DC), thus resulting very easy to build (much more than the already minimal Darlington design or than the VCO circuit mentioned above). Figure 7 shows an 8-input (that is, audio channels) DYI rectifier. An amplified and rectified signal can be used to directly feed DC motors. In general, an amplifier is mandatory when using motors, as it not only provides the current required to drive them, but, also, protects the sound card over inductive back current loads that can damage it.

To summarize, the simplest solution among the possible audio physical computing configurations represented in Figure 5 involves a DC-coupled sound card: in this way, DC signals (as typical output by sensors) can be directly used in input. AC output is simpler to handle than DC, as it does not

[5] The process can be termed "analog audio sensing", as suggested by [11].

[6] `http://www.motu.com/products/software/volta/`

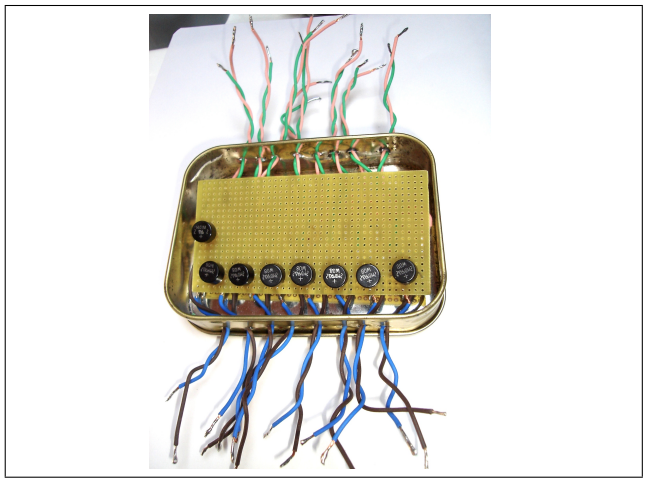

Figure 7. A DYI 8-input rectifier, AC (bottom) to DC (top).

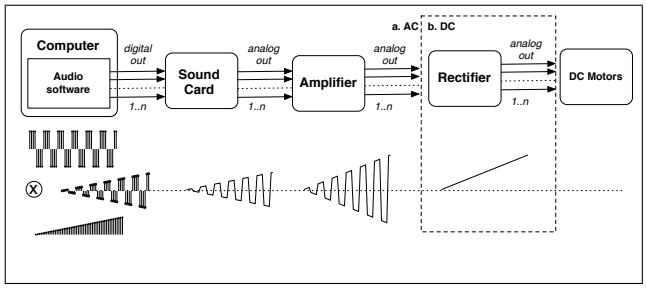

Figure 8. Audio physical computing scenario for DC motor driving: a. alternate current, b. direct current.

require further complex circuitry, it allows to include loudspeakers in the setup, to be used to deliver audio or to drive objects, and can be easily rectified, thus providing also DC signals if needed. Moreover, by inserting or removing the rectifier for some signals it is possible to define a mixed AC/DC scenario. The output part of this subchain will be described in the next section.

5. RE-ENGINEERING RUMENTARIUM IN THE LIGHT OF AUDIO PHYSICAL COMPUTING

By means of the previously discussed approach, it becomes possible to re-engineer the Rumentarium setup in order to replace microcontrollers with an audio chain. A different case, involving solenoids, will be described in the next section. As we have seen, the sound card-amplifier-(rectifier) subchain, even if starting from AC audio signals, allows to use both AC and DC signals. This possibility opens the way to two scenarios (Figure 8). The first one makes use of audio signals directly to feed the electromechanical devices, that is, it does not make use of the rectifier (Figure 8a). Collins has already proposed to use audio signals in order to drive electromechanical devices, in particular little DC motors and vibrators for cell phones and pagers [3]. He suggested to use the motors as audio drivers, by clamping their body directly to the object, thus transmitting the vibration. That is, the motors act as final amplification stages for audio signals. In the Rumentarium, DC motors are instead used at a control rate, to excite a resonator. In Figure 8 a waveform is first digitally generated via software, then converted by the sound card and finally amplified. In case of DC motors, the alternating voltage sign in the AC signal does not impede working, but simply determines a corresponding inversion in rotation's direction (that is, a negative rotation speed). For each zero crossing in the signal there is an inversion in the motor's rotation direction, where the inversion rate is determined by the signal frequency. Rotation inversion was not possible with the previously discussed microcontroller setup, while it is largely useful in the Rumentarium context, as it allows to implement a backward mechanism for motor-driven beaters. The waveform is relevant too, as it determines the way the voltage is delivered to the motor. A square wave results in a abrupt change of rotation direction while maintaining the same rotation speed. The duty cycle determines the duration ratio between opposite sign rotation phases. In Figure 8a (bottom), if a DC motor is connected to the output, the ramped square determines a constant oscillation between increasing opposite rotation speeds. Instead, a saw waveform results in a ramp signal that linearly decreases from the maximum rotation speed in one direction, to no motion (while crossing zero), to the maximum rotation speed in the opposite direction; then, it jumps abruptly to the initial maximum speed and rotation. A triangle waveform continuously increases/decreases the motor's speed while changing rotation direction two times per cycle. Even if the real, final behavior depends indeed on many mechanical constraints, varying on per sound bodies basis, nonetheless it can be grossly determined by the specific waveform in the digital signals. In this sense, a greater complexity can be achieved than with microcontrollers. In Rumentarium, the original design involving Arduinos allows only to vary the amount of direct current fed into the motors by a power supply: thus the only parameter that could be controlled was rotation speed. The control depended exclusively on the PWM modulation provided by the microcontrollers, as a result of the digital-to-analog conversion. The real transfer function for conversion is not known, and indeed a greater approximation is tolerated with microcontrollers than with sound cards. In short, in the audio physical computing scenario, algorithmic strategies for the generation of digital signals (that are straightforward processes in the electronic music domain) allow for a higher degree of control.

While still in the first scenario, it is indeed possible to use unipolar AC signals (i.e. only positive): an unipolar signal, varying only in the positive domain (the contrary would be the same in the opposite direction) will result in a speed rotation varying from null to the maximum, with no direction changes. In any case, if the aim is the recreation of a direct current, i.e. as in the original microcontroller setup, the most effective strategy is to use DC signals. This is what happens in the second scenario, where the rectifier intervenes. The ratio for using the rectifier –instead of directly connecting a DC output from the sound card into the amplifier– is to be found in the presence of the amplifier. From a computational perspective, the generation

of a DC-like signal is not an issue per se. However, even if DC-coupled sound cards are in use, the generation of DC signal can be prevented by the amplifier: designed to work in the analog audio domain, where direct current is considered a problem as it typically results from electric interferences (and can damage loudspeakers), the amplifier circuitry typically operates an automated correction step on the final signal, precisely in order to remove the (supposed) DC offset. This correction typically leads to output a silent signal. But a DC signal can be obtained by rectifying the AC signal from the amplifier. In the perspective of rectification, while different waveforms can be indeed used, the general solution is represented by a pulse waveform, oscillating between maximum values. In Figure 8b (bottom), the digital pulse wave signal is rectified after amplification: the signal resulting from rectification is a straight direct current. In this way, by varying the amplitude of the digitally-generated square wave, it is possible to continuously control motor speed, as the amount of DC current in output is varied proportionally to the amplitude of the square wave itself. In short, the amplitude envelope modulates the square wave that is, in turn, demodulated by the rectifier, so that the same envelope, once amplified, is returned in output. In this way, the same control architecture of the original microcontroller setup can be achieved.

6. CIFRE DEL COLPO: SOLENOIDS

In this section, audio physical computing as discussed before is applied to the control of solenoids. By discussing a music work, *Cifre del colpo*, it will be possible to introduce some aesthetic issues at the basis of the approach. *Cifre del colpo* ("Numbers/Ciphers of the beat") is written for 8 percussions whose sound is captured through microphones and expanded by a set of other objects, activated by solenoids. These "Expanders" are intended as a landscape of accidental sound bodies, following the Rumentarium model. But while in the Rumentarium sound production is entirely electro-mechanical, in the *Cifre* sound generation is divided between a human player and a mechanical one.

An overview of the setup showing both elements and their placement, is depicted in Figure 9. The setup includes Percussions, Microphones, Sound card, Computer, Amplifier, Expanders. Percussions are indicated with capital roman numerals (I–VIII) and must be chosen following the criterium of sharing a clear common feature. Microphones are to be placed around Percussions, in order to uniformly sample the set. Figure 9 shows four microphones, but their number can vary. In short, the microphones are listening to the soundscape created by the percussion player and reply by expanding it through the sound bodies activated by the solenoids. The number of microphones determines the number of channels, indicated in Figure 9 with numbers (1–4). Each microphone is connected to a discrete input channel of the Sound card. In turn, the Sound card is connected to a computer analyzing the input signals coming from the microphones, and generating control signals for the Expanders. The output pipeline is the one described in Figure 8, in relation to the

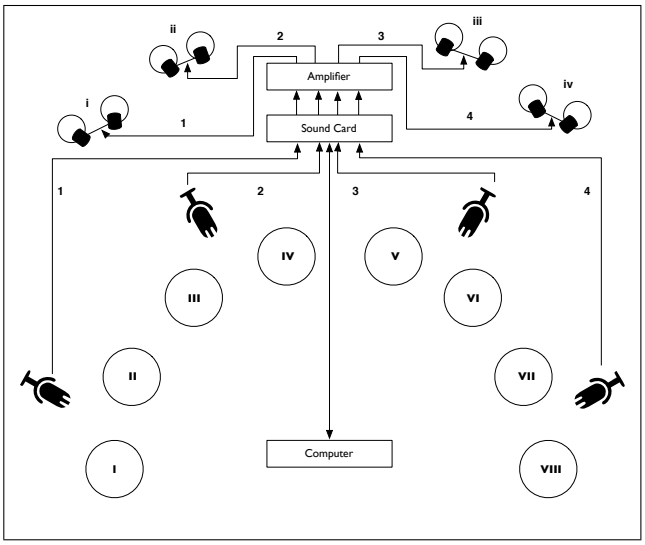

Figure 9. *Cifre del colpo*, setup.

first AC scenario. The sound card outputs another array of discrete channels, their number matching the number of input channels (hence, Figure 9 includes 4 output channels). Output channels are amplified by the Amplifier, so that the resulting audio signals can feed the solenoids in the Expanders. Expanders are indicated with lowercase roman numerals (i–iv). Sound bodies are grouped in Expander sets, and all the Expanders belonging to the same Expander set receives the same audio channel. Solenoids (and motors) are at the core of many projects involving musical robots. As an example, ModBots are digitally controlled acoustic percussive robots developed by Bill Bowen (2002), in collaboration with Lemur and installed at Angle Orensanz Foundation in 2002 [7]. ModBots are miniature, modular instruments designed with an emphasis on simplicity, and making use of only one electromechanical actuator (a rotary motor, or linear solenoid) [8]. While Bowen's Modbots are used mainly as components of sound installations, Lemur is the main technological provider of Pat Metheny recent Orchestrion project, where solenoid actuators are used interactively by the renowned guitar player in real time [9]. The ModBots project has been at the origin also of William Brent's Ludbots (2008): LudBots have been used as instruments in live performances, but also as components of the "False Ruminations" installation [10]. All the previous projects use microcontrollers to drive the motors/solenoids. Coherently with the ecologic and low-profile assumption at the basis of the Rumentarium aesthetics, the solenoids intended to be used in the *Cifre* project are coil components used in hydraulic valve systems coupled with a simple bolt, that acts as an iron shaft to be pulled/pushed when the coil is given current. As electromechanical noise (both in acoustic and behavioral sense) is a key concept in the work, they are not intended to provide

[7] http://lemurbots.org/videoandaudio.html
[8] http://public.bilbowen.net/home/digitally-controlled-acoustic-percussion
[9] http://patmetheny.com/orchestrioninfo/qa.cfm
[10] http://williambrent.conflations.com/pages/projects.html

Figure 10. Testing a solenoid on a snare drum.

a precise output, rather to show complex and partly unforeseen results. They must be applied to Expanders following different strategies depending on empirical fine tuning. Figure 10 shows a test where a solenoid is suspended over a snare drum thanks to a clip microphone holder. When the coil is magnetized, the bolt is raised above the snare skin (the solenoid is connected by two alligator clips to the amplifier visible back on the left). Then, when magnetization is over, it falls on the skin generating sound. In order to obtain a random bouncing effect, a nut is placed between the skin and the bolt. The overall signal flow for the *Cifre* is shown in Figure 11. The process applies independently to each signal from each microphone (that is, in case of e.g. 4 microphones, there would be 4 parallel processes like the one in Figure 11). The audio physical computing scenario is the following. An onset detection algorithm is running on the computer. When an onset is detected in the input signal is, a pulse train t is digitally generated. The pulse train contains an unipolar square waveform of 0.25 seconds. The pulse train is converted, amplified and sent to the solenoid, so that it flips abruptly between no motion (amplitude is 0, and applied voltage is null) and maximum motion (where signal amplitude is at its maximum too). The pulse train waveform must have a frequency in the range $[1.0, 10.0]$, to be chosen by the interpreter. The lowest value results in a single beat, the highest in a tremolo-like effect. The amplitude of t is determined by amplitude tracking of input signal is, scaled by function f_{scale}, that truncates amplitude intensity (expressed in dB) in the range $[-30, 0]$, and linearly scales the result in the range $[0.1, 1]$, to be used as a multiplier for t amplitude. All these processes, integrating audio analysis on input signals from microphones and signal generation for the solenoids, are easily handled by a software application named *Guardian*, that has been specifically developed in SuperCollider for the work. Figure 12 shows the GUI for a 4-channel I/O setup, where GUI elements are the replicated for each channel. From top to bottom, GUI shows: a scoping window showing the input signal for the channel (channels 1 and 2 are receiving each an input signal); four knobs/number boxes respectively for input signal volume, for onset detection threshold, for frequency of output square signal (0-10) and for output signal volume, a scoping window showing the output signal for the channel (channels 1 and 2 are outputting control signals as a response to onset detection on respective input). The application is not intended as an official version, but just as one of the possible implementations with respect to the description of the algorithm provided by the score.

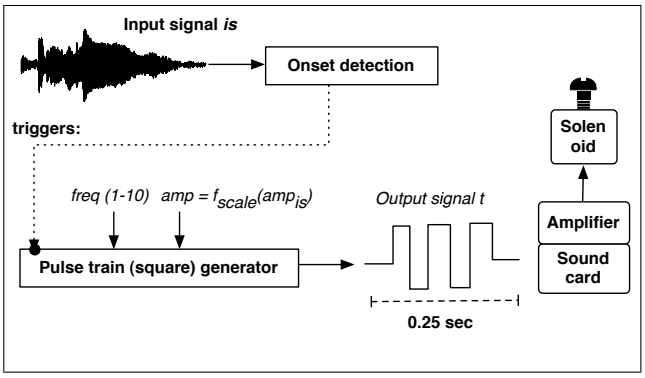

Figure 11. Onset detection, triggering and signal generation.

Figure 12. GUI for the Guardian application.

7. CONCLUSIONS AND FUTURE WORK

Audio physical computing is not intended as a general approach to physical computing. Nevertheless, at least for the discussed specific tasks, it has proven to be able to solve many issues emerged while using microcontrollers. First of all, the overall hardware/software chain is reliable, as it uses very stable technologies (both on software and hardware side). Moreover, the setup is of immediate realization, as it uses standard audio connectors to link the element of the chain. The only element to be created from scratch is the rectifier, but its assembly is a trivial task. Probably, the most relevant benefit that comes from us-

ing audio signals as control signals for electronic devices is the deep integration between electronic music competences and physical computing, since strategies for algorithmic generation of digital signal can be directly applied to the control of physical objects. Finally, the setup has revealed a specific flexibility, as it allows to mix in output sound from electromechanical devices (like the ones discussed before), with digital sounds (and, moreover, in a sort of continuum, sounds not generated but amplified by motors can be added). The setup devised for *Cifre del colpo* is also passible of further developments in terms of sound installation. In particular, it can evolve into a feedback system, where the audio output of the sound bodies, captured in input by microphones, is mapped into audio signals for the sound bodies themselves, like in audio feedback installations (see [15]).

Acknowledgments

A thank is due to Francesco Richiardi for technical inspiration and support, to Enrico Cosimi for his suggestions on analog synths, and to Antonino Secchia for his dedication to the performance of *Cifre del colpo*. I am also grateful to the anonymous reviewers for their insightful comments.

8. REFERENCES

[1] D. O'Sullivan and T. Igoe, *Physical Computing. Sensing and Controlling the Physical World with Computers*. Boston, Mass.: Course Technology, 2004.

[2] T. Igoe, *Making Things Talk*. Beijing–Cambridge–Farnham–Köln–Sebastopol–Taipei–Tokyo: O'Reilly, 2007.

[3] N. Collins, *Handmade Electronic Music. The art of hardware hacking*. New York–London: Routledge, 2006.

[4] S. Goto, "The case study of an application of the system, "bodysuit" and "roboticmusic": its introduction and aesthetics," in *Proceedings of the 2006 conference on New interfaces for musical expression*, ser. NIME '06. Paris, France, France: IRCAM — Centre Pompidou, 2006, pp. 292–295. [Online]. Available: http://portal.acm.org/citation.cfm?id=1142215.1142287

[5] A. Valle, "The Rumentarium project," in *Proceedings of the international conference on Multimedia*, ser. MM '10. New York, NY, USA: ACM, 2010, pp. 1413–1416. [Online]. Available: art05609s-valle

[6] P. Tamborrini, *Design sostenibile. Oggetti, sistemi e comportamenti*. Milan: Electa, 2009.

[7] S. Seriff, *Recylced Re-seen*. Santa Fe: Museum of International Folk Art, Santa Fe, 1996, ch. Folk Art from the Global Scrap Head: The Place of Irony in the Politics of Poverty, pp. 8–29.

[8] M. Banzi, *Getting started with Arduino*. Sebastopol: O'Reilly, 2009.

[9] S. Wilson, D. Cottle, and N. Collins, Eds., *The SuperCollider Book*. Cambridge, Mass.: The MIT Press, 2011.

[10] AMP2, "Hopeful Monster," Die Schachtel-Zeit Imp1.9, Milan.

[11] S. Kersten, M. A. Baalman, and T. Bovermann, *The SuperCollider Book*. Cambridge, Mass.: The MIT Press, 2011, ch. Ins and Outs: SuperCollider and External Devices, pp. 105–124.

[12] J. S. Wilson, Ed., *Sensor Technology Handbook*. Burlington, MA – Oxford: Newnes, 2005.

[13] H. Austerlitz, *Data Acquisition Techniques Using PCs*, 2nd ed. San Diego – London: Academic Press, 2003.

[14] C. Roads, *The Computer Music Tutorial*. Cambridge, MA, USA: MIT Press, 1996.

[15] A. Di Scipio, "Sound is the interface. sketches of a constructivistic ecosystemic view of interactive signal processing," in *Proceeding of the XIV CIM 2003*, N. Bernardini, F. Giomi, and N. Giosmin, Eds., Firenze, 2003, pp. 128–131.