Extending Physical Computing On The Reactable

Mathieu Bosi

MASTER THESIS UPF / 2009
Master in Sound and Music Computing

Master thesis supervisor
Sergi Jordà
Department of Information and Communication Technologies
Universitat Pompeu Fabra, Barcelona
To my family and friends
Contents

1 Introduction 5
  1.1 Motivation ........................................ 5
  1.2 Aim of The Work .................................. 5
  1.3 Outline of the Document .......................... 6

2 State Of The Art 7
  2.1 The Reactable ..................................... 7
  2.2 Intimacy and Enaction in NIMEs .................... 8
  2.3 Position, Impact, and Pressure Sensing Technologies for Interactive Surfaces .......................... 8
     2.3.1 Acoustic sensing techniques .................. 8
     2.3.2 Computer Vision Techniques .................. 11
     2.3.3 Approximate Pressure and Position Detection using Few Force Sensing Elements .................. 12
     2.3.4 Dense Surface Sensor Arrays .................. 13

3 Implemented System 15
  3.1 Enabling Technologies ............................ 15
  3.2 Acquisition System Setup ........................ 15
     3.2.1 Force Sensing Resistor ....................... 16
     3.2.2 Mechanical coupling between sensors and tabletop surface .................................. 17
     3.2.3 Analog channel multiplexing and signal conditioning .................................. 17
     3.2.4 Arduino ........................................ 19
     3.2.5 The Serial Communication Protocol ............ 20
  3.3 Analysis Of The Digitized Signals .................. 20
     3.3.1 Individual sensor signal processing: Hit detection .................................. 22
     3.3.2 Pressing-points position computation on the table perimeter .......................... 22
     3.3.3 Hit position computation on the table perimeter .................................. 23

4 Test Application 27
  4.1 Pure Data patch ..................................... 27
     4.1.1 The [comport_fsr] Pure Data external .......... 27
CONTENTS

4.2 Latency Measurements ................................................. 28
   4.2.1 Serial over USB latency issues with the FTDI chip .... 30
   4.2.2 Decreasing the latency ..................................... 30

5 Proposed New Reactable Objects .................................. 31
   5.1 New Reactable Objects ........................................ 31
       5.1.1 Envelope Trigger ...................................... 32
       5.1.2 Pitch Mallet .......................................... 32
       5.1.3 Pressure-Angle Follower .............................. 33
       5.1.4 Collective Control Object ............................. 34

6 Conclusions and Future Work .................................... 35
   6.1 Using a more performing hardware solution ............... 35
   6.2 Incorporating the new Objects into the existing Reactable application 35

Bibliography ............................................................ 37

List of Figures .......................................................... 41

List of Tables ........................................................... 43
Abstract

Multi-touch Tangible User Interfaces (MTUI) offer new possibilities for the control of sound and music, but they often lack of effective responsiveness to some musically important gestures such as tapping or pressing force. The proposed project has been aimed at improving the quality of interaction with the Reactable audiovisual MTUI. A prototype system based on force sensing has been developed. The system is able to detect impacts and continuous pressures exerted on the Reactable perimeter in more points at the same time. The system’s development has been based on a tight loop between the tuning of the hardware and software, and personal testing of musical control effectiveness. After the description of the peculiarities of the implemented system, a set of new possible Reactable objects is proposed. The developed prototype system opens the door to the further expansion of the Reactable system, allowing for a richer and more enticing music creation experience.

Keywords: Multi-touch Tangible User Interfaces, Force Sensing Resistor, Microcontroller, Reactable, Enaction, NIME
Acknowledgements

I would like to thank all the people who accompanied me during this one year of life here in Barcelona:

- the Reactable team, especially Sergi Jordà, Carles Fernandez Julià and Daniel Gallardo, for being such nice office mates and for being supportive in my research.
- the whole MTG department, for also supporting my research work.
- the SMC Master and my course mates: Leonardo Aldrey, Juan Jesus Cabrera, Laia Cavigal, Richard Eakin, Nathaniel Finney, Frederic Font, Francisco Garcia, Stefan Huber, Robert Hutchison, Victor Jimenez, Olivier Lalonde, Daniel Martin, Agustin Martorell, Pablo Molina, Saso Musevic, Thomas Najnudel, Tan Hakan Ozaslan, Giotis Panagiotis, Cristian Quirante, Anandhi Ramesh, Zurie Resa, and Charalampos Christos Stamatopoulos. You are all great and I’m happy that we lived together this master experience.
- Agustin, Graham and Owen for being so nice singing mates
- The ESMUC Laptop Orchestra ensemble: Josep Comajuncosas, Ariadna Alsina, Alex Barrachina, Graham Coleman, Oriol Tiò, Medin Peiron, Regina Domingo, Enric Guaus.
- Luis, Miguel, and Jonathan, for being such nice guys and music mates in the band Thy Omen.
- Timothy Duffin, for helping me out in the proof-read work, and my other flatmates from the CSIM master, Ina Ghita and Cosmin Pojoranu: I’m so happy I could live with you this so intense master period. I really love you all.
- My family, which has been always supportive toward me.

Thank you all

Mathieu
Chapter 1

Introduction

Musical ideas are prisoners, more than one might believe, of musical devices.
Pierre Schaeffer, 1977

1.1 Motivation

Recently one of the new promising directions in the field of NIMEs is that represented by live music performances based on tangible tabletop interfaces, which allow for a collaborative real-time interaction. The Reactable, developed by the Music Technology Group of Universitat Pompeu Fabra, is an example of a successful tabletop NIME of this kind. Each Reactable tangible object is associated with a virtual sound synthesis module which dynamically gets connected with the others to create a modular synthesizer. At present, Reactable objects can be controlled by moving them in two dimensions and by rotating them. Also, finger gestures on the tabletop are used both for controlling the modules’ parameters and for influencing the connections between them.

While the Reactable already offers a wide enough spectrum of musical expression possibilities, the system is still not able to exploit other important user gestures like percussive impacts, and continuous finger/hand pressure and their nuances. The involvement of one’s body when playing an instrument is an important factor for a more effective musical expression, and it is the key when trying to create an enticing musical interface.

1.2 Aim of The Work

My master project is aimed at improving the quality of interaction with the Reactable, to make it a more complete musical instrument by trying to recreate the deep coupling existing between a musician and his musical instrument. The choice has been that of combining the already existing system with a circular array of force sensors to detect angular position and intensity of pressing and hit gestures on the table’s perimeter.
1.3 Outline of the Document

In the next few chapters I will present a state of the art on previously explored pressure and impact sensing technologies for interactive surfaces. I will then overview the up to now implemented system and then explain how the data gathered from the sensors will be mapped to new Reactable objects.
Chapter 2

State Of The Art

Here follows a brief review of the Reactable system. Interaction intimacy and the enactive approach to user interaction design for musical interface are then presented. To conclude, the already existing solutions to track user gestures on surfaces are recalled.

2.1 The Reactable

The Reactable [17, 18], is an electronic music instrument based on a multi-touch tangible tabletop interface, exploiting a modular synthesizer approach and a dynamic visual data-flow programming language [21]. The Reactable system supports the detection of tangible objects on its surface, which are associated with various units of a virtual modular sound synthesizer. Finger touches and strokes are detected, and are used both to control the connections between the objects and to adjust some parameters related to each unit.

The recognition of the position and rotation angle of the tangibles on the surface is achieved by tracking fiducial symbols located underneath them. A digital video camera captures the image of the fiducials from below the table surface, which is translucent. The live video stream is then processed by the free and open-source computer vision software reacTIVision [19], appositely developed for the Reactable project. reacTIVision efficiently detects the 2D position and the rotation angle of each fiducial on the table surface, and then sends this data by mean of the TUIO protocol [20], based on the Open Sound Control network protocol [45], to the further stages of the Reactable application. These further stages are in charge of implementing the dynamic patching between the synthesizer modules, to produce the actual synthesis results, and to offer an augmented reality feedback which is back-projected to the translucent tabletop surface (fig. 2.1).

A diagram of the system’s working is depicted in fig. 2.2. The augmented reality system shows to the performers and to the public the connections between the elements of the synthesis patch by mean of waveforms traveling between them. This kind of signal visualization is also found in the Reactable’s predecessor FMOL [16] and it has the key-role of making aware the musician-programmer and the public
CHAPTER 2. STATE OF THE ART

about what’s going with each part of the synthesizer in a quick and intuitive visual way. As previously said, in its current state the Reactable system is still not able to detect gestures like percussive impacts and continuous finger/hand pressure.

2.2 Intimacy and Enaction in NIMEs

The notion of control intimacy when playing an instrument was introduced by Moore in [27]. A person having a high degree of intimacy with a device can communicate through it in an effective way, as if it was an extension of herself. This process is described by Fels as embodiment of the device [10]. Control intimacy also has been discussed by Wessel in terms of a system’s latency and jitter (latency variation) in [44]. The upper latency bound is set by Wessel to 10ms and the jitter amount to 1ms for quick gestures like flams.

The enactive approach poses particular emphasis on the role of sensory-motor engagement in musical experience, in order to create a symbiosis between the musician and its instrument. Wessel has first proposed in [42] the use of the enactive approach to computer music performance. In this approach, rich and precise gestural interfaces are necessary for the instrument to inspire the development of virtuosity. Also, the instrument design process needs to include aesthetic criteria. Quoting Wessel, “the instrument must be composed”. The other fundamental elements are connectivity devices, proper gesture analysis and mapping software, and richly controllable sound synthesis and processing. The aspects of the enactive approach to digital musical instrument design are also extensively discussed by Armstrong in [1]. Recently Marshall has proposed in [25] various ways to improve the overall feel of digital musical instruments through the study and design of their physical interface, that is: the instrument body, sensors, and feedback actuators. The enactive approach in designing tangible interfaces has been discussed in [9, 2].

2.3 Position, Impact, and Pressure Sensing Technologies for Interactive Surfaces

When interacting with NIMEs, important interactions are those which are bound to the detection of user contact with the physical interface. In particular, various techniques have been developed to sense hit intensity, position, and pressures on planar surfaces.

2.3.1 Acoustic sensing techniques

Tactile interfaces based on acoustic detection techniques are usually referred to as Tangible Acoustic Interfaces (TAIs). Acoustic sensing techniques for tracking objet hits and knocks on interactive surfaces of large dimension were carried out in [14] and in [29]. Several techniques and tools for transforming daily life objects
2.3. POSITION, IMPACT, AND PRESSURE SENSING TECHNOLOGIES FOR INTERACTIVE SURFACES

Figure 2.1: A typical Reactable session. The connections carry the dynamic visual information about the waveforms which are traveling from a node to the other before of reaching the central sound output “sink”.

Figure 2.2: Schematic view of the Reactable system’s working.
into tactile interfaces have also been explored in the TAI-CHI project * (Tangible Acoustic Interfaces for Computer-Human Interaction), improving the knowledge and use of TAI s by developing acoustics-based sensing technologies which are able to transform almost any object into an interactive interface. The latest outcome of this research has been presented by Crevoisier in [4, 5] with various interactive surfaces including the Percussion Tray system, shown in figure 2.4.

Figure 2.3: The original concept of TAI as musical interface (from [6]).

Figure 2.4: The Percussion Tray with four acoustic sensors connected to the Presto Kit processing module (from [5]).

*http://www.taichi.cf.ac.uk/*
Overall, the biggest limitations of sound sensing based techniques are the following:

- need for elaborated hardware to achieve low latencies and high spatial resolutions
- continuous pressure can’t be sensed
- multi contact input is not possible.

### 2.3.2 Computer Vision Techniques

In the particular case of back-projected tabletop surfaces, multiple contact points positions and pressures can be detected by using computer-vision detection and tracking algorithms, with a setup based on the *Frustrated Total Internal Reflection* (FTIR) technique \[11\] (fig. 2.5) coupled with the use of a silicone rubber layer acting as compliant surface †, as specified by Smith in \[38\] (fig. 2.6). This setup allows not only to use fingers to interact but also other kinds of objects making contact with the surface, like in example brushes or other graspable objects.

![FTIR technique diagram](image)

**Figure 2.5**: Representation of the FTIR technique’s working, as proposed by Han.

Such vision-based system can provide high spatial resolution, however the biggest drawback when it comes to NIMEs, is the need of costly high frame-rate cameras (i.e. a 200 FPS camera for a 5ms sampling rate) and proper processing hardware and software to achieve low enough latencies in interaction.

†The compliant surface has the fundamental role of keeping an effective optical coupling between the finger or object and the surface while in movement.
CHAPTER 2. STATE OF THE ART

Figure 2.6: The pixel intensity grows as touch pressure grows. The left image is a zero-force touch, center is a light press, and right is a hard press (from [38]).

2.3.3 Approximate Pressure and Position Detection using Few Force Sensing Elements

Force sensing devices are commonly used in biomedical applications [24, 15, 7] and in mechatronics [22, 34]. In particular, Force Sensing Resistors (FSR) represent an interesting choice due to their mechanical resistance and low cost [46]. FSRs have often been used for the construction of NIMEs [41] and are commonly found in drum-pad controllers and in middle-end or high-end controller keyboards to provide individual or global key pressure information (channel after-touch and polyphonic key pressure parameters in the MIDI standard). Their use is particularly desirable due to the low crosstalk when they are used to detect hits on a surface [40].

A interesting contribution for this thesis is represented by the work of Schmidt et al. about a context acquisition system based on load sensing [36, 37]. In this work the forces detected at the four corners of a surface are used to determine the 2-D position of objects on it and to detect context changes and user pointing interaction (fig. 2.8). Another example of position and force sensing device based on few sensing elements is the one developed by Wessel et al. in [43], where an array of Interlink FSR touch-pads are disposed into a matrix layout and are connected with
custom hardware to a software synthesis system handling multiple musical control structures.

![Diagram of force distribution on a surface](image)

Figure 2.8: The system developed by Schmidt. For their higher precision and weight tolerance, strain gauges are used.

### 2.3.4 Dense Surface Sensor Arrays

These techniques rely on an array of discrete sensors of various nature directly disposed in the whole interaction surface. The amount and spatial disposition of the sensing element changes from few to many units per unit surface area, influencing the achievable spatial resolution. Some examples of research in this field can be found in the so called Sensate Skin developed by Paradiso et al. [30, 31]. One
of the first commercially available devices of such kind has been offered by Tac-tex [13], whose devices are based on optic fibers used to measure the compression of a translucent compressible foam.

A much cheaper and efficient technique of capturing high-quality anti-aliased pressure images at high frame rates from a surface has recently been presented in [32]. This technique is based on a newly developed sensor, the Interpolating Force Sensing Resistor (IFSR). This system is rugged, durable, has a wide dynamic range and is capable of capturing even subtle variations in gesture pressure. It is easily scalable from the smallest surface to the widest, like for example a wall. Moreover the sensor is paper-thin and transparent, a fundamental aspect when dealing with a touch-screen type of interface. Two of the IFSR based prototypes are shown in figure 2.9. At the moment of writing these IFSR systems uses the PIC24H micro-controller by Microchip to implement the sensor array scanning circuit and the communication with the host PC. A view of this circuitry and a diagram showing the layout of the sensing array are shown in figure 2.10.

Figure 2.9: Two of the IFSR-based touch-pad systems.

Figure 2.10: On the left the sensor multiplexing circuitry based on a set of shift registers and on the PIC24H micro-controller. On the right a schematic view of the implementation of the sensing array, based on orthogonal sets of conductors separated by an FSR material.
Chapter 3

Implemented System

The sensing capabilities of the Reactable hardware have been expanded by means of a pressure sensing device interfaced with the host PC by mean of a micro-controller. Following here is the description of the acquisition system hardware setup and of the implemented signal analysis algorithms.

3.1 Enabling Technologies

The technologies enabling the extension of physical computing on the Reactable are micro-controllers and pressure sensing technologies. Pressure and impacts on the table perimeter are detected by means of sensors. In particular, the chosen solution is based on the use of force sensing resistors disposed under the perimeter of the table to detect continuous pressures and impacts on the whole tabletop perimeter. The key points followed in developing this system have been:

- low implementation costs (hardware complexity and cost in money)
- embedding factors
- effectiveness (sensitivity, low latencies)

3.2 Acquisition System Setup

An array of 16 force sensing resistors is laid at regular distances beneath the perimeter of the circular tabletop surface. The sensors are then connected by mean of a multiplexing circuit to a signal conditioning circuit to prepare the signal for the analog to digital conversion on the micro-controller.

The micro controller samples the analog signals and converts them to a lightweight serial protocol, this is then used to send the sensor data to the host computer where further signal processing is carried out to extract the final hit position and pressure informations. A schematic diagram of this system is shown in figure 3.1.
3.2.1 Force Sensing Resistor

An useful report on the performances of commercially available FSRs is presented by Hollinger and Wanderley in [12]. After looking for possible alternatives, an Interlink FSR force sensing resistor was chosen (figure 3.2) for its price and performance factors.
3.2. ACQUISITION SYSTEM SETUP

3.2.2 Mechanical coupling between sensors and tabletop surface

An optimal mechanical coupling between the sensors and the tabletop surface is the first needed step to provide the best possible dynamic range and sensing response. The guide provided by Interlink [8] aids in this important task. The key aspects which have been kept into consideration are now reported.

**Sensing accuracy:** while an FSR is not accurate as a strain gauge, for the use made here this accuracy is considered to be good for the implemented application.

**Force sensing resolution:** force resolution is very good ($\pm 0.5\%$ of full use force). This enables even very subtle gestures to be detected.

**Sensing range:** the sensing range of the used FSR covers 3 orders of magnitude, from 10$g$ up to 10$Kg$ or more. This allows the detection of both subtle nuances and of gestures like strong hits.

**Sensor geometry:** the chosen sensor (Part No. 406 – 1.5 ” Square) is the one amongst the available ones that best matches the geometry of the application.

**Set-up of the Mechanical Actuation System:**

During the construction of the system, the following aspects had to be kept into consideration for the sensing system to properly work:

- FSR response is very sensitive to the distribution of the applied force. To keep the force distribution more consistent, a thin elastomer had to be used. After experimenting with different materials, a thin layer of expanded polyethylene foam was found to offer the best mechanical coupling, with the fastest response and nearly no hysteresis. The elastomer layer also has the function of protecting the sensor.

- The FSR needs to be laid on a firm, flat, and smooth mounting surface. This is the case in this implementation, since the Reactable surface already lies on such kind of surface.

- During installation, special care is needed to avoid the presence of even very small kinks or dents in the FSR active area, that can cause false triggering of the sensors. Once the surface is clean, the covering elastomer layer protects the sensors from the deposit of extraneous material.

3.2.3 Analog channel multiplexing and signal conditioning

After being multiplexed by means of two 8-channel analog multiplexer ICs (HEF4051), the sensors resistance is transformed into a voltage by mean of a voltage divider
CHAPTER 3. IMPLEMENTED SYSTEM

Figure 3.3: View of the sensing perimeter surface and detail of one of the sensing parts together with the expanded polyethylene elastomer.

type of circuit. The relation between the resistance values and the output voltage is represented by the following equation:

\[ V_{out} = V_{in} \frac{R_2}{R_1 + R_2} = \frac{V_{in}}{1 + \frac{R_1}{R_2}} \]  

(3.1)

Figure 3.4: The voltage divider circuit (one of the two resistors is the FSR of which we want to know the resistance).

When we deal with a voltage divider circuit special care is needed, trading range of response with sensitivity *. To interface the voltage divider with the ADC, an operational amplifier (the LM324 by Linear) is used to buffer the voltage. In order

*A tutorial on the optimization of the voltage divider circuit can be found at [http://cnmat.berkeley.edu/](http://cnmat.berkeley.edu/) search engine terms: “optimizing voltage divider”.
3.2. ACQUISITION SYSTEM SETUP

to operate the operational amplifier in the whole 0–5 Volts range used by Arduino ADCs, a bigger voltage is needed, so an additional regulated supply circuit had to be added. This implementation with its relative force-voltage curve is shown in figure 3.5.

Figure 3.5: The used buffered voltage divider circuit.

An improvement to be tried is the use of a current to voltage converter (transimpedance amplifier) which is able to convert the force-voltage relation into a more linear one [8].

3.2.4 Arduino

Arduino [26] is a programmable micro-controller platform created to enable low-cost, simple and rapid prototyping of physical computing systems based on sensors and actuators. A picture of the used version, Arduino Duemilanove, is visible in figure 3.6. The Arduino platform is based on the ATMega168 micro-controller family and, thanks to its boot-loader firmware, no external ICSP programmer is needed. This allows the device to be directly programmed from the PC by simply plugging it into a free USB port and by using the provided IDE. Arduino can be programmed in a C-like language that gets then translated to machine code before of being uploaded to the chip. The tasks for which the micro-controller has been programmed are:

• controlling the multiplexers to periodically route the 16 input channels into two of the available analog inputs

• analog to Digital Conversion of the sensor values at 10bit resolution, 77 KHz as the maximum achievable sampling rate.

• implementing a lightweight serial protocol where the channel ID and values are encoded into 2 bytes.
• sending the encoded data to the host computer for further processing.

The serial data is sent to the host PC by mean of the on-board FTDI chip which implements a virtual serial connection over USB.

3.2.5 The Serial Communication Protocol

As previously said, Arduino communicates with the host computer by mean of a serial communication over USB. A lightweight protocol has been implemented to send each one of the 16 sensors 10\textit{bit} values with 2 bytes.

A limitation was later discovered is that with the 16\textit{MHz} resonator used to clock the on-board ATMega168 micro-controller, the maximum usable baud-rate is of 38400 bps. This limit can be overcome by reprogramming the Arduino bootloader (with an ICSP programmer) to allow the new clock rate and by then replacing the resonator with a 20\textit{MHz} one, which is the maximum allowed for the used micro-controller [3]. With this modification it would become possible to reach a speed of 115200 bps, which represents a factor of 3 in speed-up.

3.3 Analysis Of The Digitized Signals

After the analog signal acquisition and digitizing step, signal processing techniques have to be implemented in order to extract information useful to control the musical instrument. The implemented techniques make use both of the single sensor data, and of the information coming from more of the sensors. For a single sensor, hits and continuous pressures are computed. By then analyzing the data coming from the whole array, approximate pressing point positions and magnitudes are detected.
3.3. ANALYSIS OF THE DIGITIZED SIGNALS

Figure 3.7: The built prototype circuit together with the Arduino programmable microcontroller.

Figure 3.8: Diagram showing the software architecture of the implemented system.
Up to a certain extent, also approximate hits positions and magnitudes can be detected.

### 3.3.1 Individual sensor signal processing: Hit detection

Envelope detection is now obtained by applying a simple attack detection algorithm. A lower activation threshold above background noise is first set. Once the pressure value goes above this first threshold, the following maximum peak value is used as hit magnitude information. The second threshold is then used to allow attack re-triggering only after the pressure level has gone below a given value.

![Diagram of peak detection algorithm functioning.](image)

An adaptive lower threshold is computed for each channel by mean of an exponential weighted moving average (Low-pass filter) of the incoming values over a wide enough temporal window.

\[
T_{activation} = k x_n + (1 - k) x_{n-1}
\]  

(3.2)

where \( k \) is a chosen constant close to zero.

An option to be explored to reduce latency is that of trying to forecast a peak estimate value by detecting particularly fast changes in the derivative of the signal. It has been seen that for fast hit gestures the current sampling rate is not sufficient to perform this task. Another potentially effective solution to be tried is that of using a matched filter together with a properly computed threshold value, as proposed in [28].

### 3.3.2 Pressing-points position computation on the table perimeter

The 2-D coordinates of a center of pressure \( \mathbf{G} \) are computed in the following way:
3.3. ANALYSIS OF THE DIGITIZED SIGNALS

\[ G = \begin{cases} 
0 & \text{if } S_i^{\text{pressure}} \leq T_{\text{low}} \\
\sum_{i=1}^{n} S_i^{xy} S_i^{\text{pressure}} & \text{else}
\end{cases} \]

where \( n = 16 \) is the total number of sensors and \( S_i^{xy} \) are the normalized \( xy \) coordinates of the \( i \)-th sensor on the table. In our case these points are regularly spaced on a circumference. A lower threshold value \( T_{\text{low}} \) is used in order to minimize the influence of overall bit-errors and noise from the ADC. The single contact angle \( \alpha_{\text{touch}} \) is computed by mean of the four-quadrants arctangent function:

\[ \alpha_{\text{touch}} = \text{atan2} (g_x, g_y) \]  

(3.3)

The pressure graphs and the results of this computation for a press moving around the table perimeter are shown in figure 3.10 and 3.11. When more than a single user is touching the table perimeter, unique touch ID are generated and associated to each moving force peak which is tracked between following frames. In this case the computation of the contact angle is computed by restricting the considered sensors to those contributing to the force peak.

3.3.3 Hit position computation on the table perimeter

The \( G \) trajectory graph for some rapid hits is shown in figure 3.14. From this plot it is evident that the currently available temporal resolution does not manage to faithfully track fast hits. Some hit position approximation approaches based on \( G \)’s magnitude and average angle have been tried but it has been seen that higher sampling rates are necessary to achieve better precision.
CHAPTER 3. IMPLEMENTED SYSTEM

Figure 3.10: Pressures detected for a moving press along the table perimeter.

Figure 3.11: The resulting 2D information for the above data. In red $G$, in green the corresponding computed angular position. The blue lines show the corresponding $G - \alpha_{touch}$ pairs.
3.3. ANALYSIS OF THE DIGITIZED SIGNALS

Figure 3.12: The prototype application user interface showing the detected forces. On the left pressing on a single point, in the middle over a wider area. On the last image the overall magnitude of the applied force is shown together with the corresponding computed center of pressure (square) and angle (triangle).
Figure 3.13: Pressures detected for some fast hits on one side of the table. The gray dashed lines show the orientation of the sensors axes.

Figure 3.14: Resulting trajectories of $G$ (from blue to red in time).
Chapter 4

Test Application

An application has been created to test the implemented algorithms. This application allowed fine tuning of the algorithms, making them ready for incorporation with the existing Reactable software. An external Pure Data object has been created, together with a simple synthesis patch. Simple visual feedback has been provided by realizing a Processing* application.

4.1 Pure Data patch

A Pure Data patch has been created to allow the experimentation with real-time sonic feedback.

4.1.1 The [comport_fsr] Pure Data external

A Pure Data external was developed to free the Pure Data engine of the computational load and to allow for a more agile implementation of the sensor data processing algorithms described in chapter 3. This external has been developed by starting from the [comport] external and by adding the needed decoding and signal processing functionalities. The implemented Pure Data patch incorporating the object is shown in figure 4.1. The external outputs the following data:

- **Individual sensor values**
  - a stream of (SensorID, Sensorvalue) pairs
  - instantaneous messages for detected hits in the form of (sensor ID, peak value) pairs.

- **Compound values**

  Compound values are computed starting from all the sensor values. Each tracked continuous contact point is resolved by the external and is characterized by its unique ID, \((G_x, G_y), \alpha,\) and overall pressure. Experiments have shown that it is

---

*Processing is freely available at [http://processing.org/](http://processing.org/)
possible to detect at least 3 independent pressure points around the perimeter. A graph of the detected forces in such a case is visible in figure 4.2.

Figure 4.2: Pressure graph for 3 independent pressure points.

4.2 Latency Measurements

The gesture-to-sound response time of a system is determined by the sum of the latencies of all of its subcomponents. In our case the system cumulative latency is composed by:

---
4.2. LATENCY MEASUREMENTS

- Time needed for the sensor to record the user’s action (temporal sampling-rate)
- Transmission time between the sensor and the gesture interpretation unit (in our case the serial over USB connection, and storage in system memory of the sensed data)
- Processing time needed to interpret the gesture (peak detection)
- Audio synthesis latency (output audio buffers size, sound card drivers efficiency)
- Visual synthesis latency, constituted by:
  1. time needed to render the interface/output
  2. visualization device delay (see [33]).

Since we are just interested to measure the gesture to sound latency, a system analogous to the one proposed in [39] has been used. In this case the followed approach has been that of using a second computer to record both the sound of the hit on the surface and a click generated when the impact data was received. One of the result of such procedure is shown in figure 4.3, where it can be seen that the overall gesture-to-sound latency is around 50\( \text{ms} \). This delay time has been found to be acceptable when playing not too fast or complicated rhythmic patterns, however it becomes unacceptable when trying to perform faster and more complex rhythmic patterns. The reasons for such a big delay have been investigated. Part of this delay is due to the limitations of the sound card used in the tests, which can only achieve delays of about 15\( \text{ms} \) under Windows Vista, using the *ASIO4ALL* drivers†.

![Figure 4.3: Plot of the recorded gesture-to-sound latency.](image)

†The *ASIO4ALL* drivers are freely available at [www.asio4all.com/](http://www.asio4all.com/)
4.2.1 Serial over USB latency issues with the FTDI chip

The remaining delay time has been found to be caused by some limitations imposed by the FTDI chip used by the Arduino platform to send the serial data. This chip appears not to be optimized for low latency data transfer [23]. Due to this fact, the only viable solution seems to be that of using a better platform.

4.2.2 Decreasing the latency

A good candidate to replace the Arduino platform seems to be the PIC 18F2553 micro controller, produced by Microchip. This chip is currently used by some implementations of the uOSC protocol developed at CNMAT [35] and the reported latencies are in the range between 6 and 8\,ms. On the other hand the use of a higher quality sound card with high performance drivers should allow to reduce the synthesis latency up to less than 2\,ms, bringing the overall latency to a much more acceptable value (around 10\,ms).
Chapter 5

Proposed New Reactable Objects

New Reactable objects will enable the use of parameters coming from the sensors to control the already existing objects. Each new object is automatically associated with a sector of the table perimeter of which sensor values are then used. The new objects are connected to the traditional ones depending on proximity, or preferably by creating a sticky connection between them, so that the new object can be positioned without constraints on the table. The new objects can not be connected between themselves.

5.1 New Reactable Objects

Following here is a description of the behavior of each new object type in relation with the others. These new object are called: Envelope Trigger, Pitch Mallet, Pressure-Angle Follower, and Collective Control Object. Since these proposed objects interact in a new way with all of the already existing ones, a new shape should be used. The proposed shape and symbols identifying the new objects are shown in figure 5.1.

Figure 5.1: The proposed symbols for the new objects.
5.1.1 **Envelope Trigger**

The object rotation allows to change the note duration: from a whole note down to 16th note, in 7 steps. This duration is relative to the current performance BPM. A visual feedback shows the currently selected note duration. It is possible to toggle between an ADSR kind of envelope to a binary on-off envelope by pressing with two fingers nearby the object. The binary envelope is softened at the edges to allow for click-less sample playback. In the case that the interaction consists in a sustained pressure rather than in a simple hit, after-touch is implemented by generating triggering events at the selected tempo interval, and by changing the envelope amplitude in relation with the instantaneous pressure value. Dotted notes can be activated and deactivated by mean of a sensitive area located near to the object: in this way it becomes possible to obtain syncopation in the case of sustained triggering. The way this object interacts with the already existing ones is shown in table 5.1.

<table>
<thead>
<tr>
<th>Object</th>
<th>Impact</th>
<th>Pressure (after-touch)</th>
<th>Angle</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oscillator</td>
<td>amplitude envelope</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Sampler</td>
<td>trigger sample, control amplitude</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Loop</td>
<td>envelope amplitude (slicing)</td>
<td>envelope amplitude</td>
<td>-</td>
</tr>
<tr>
<td>Scratch</td>
<td>trigger start</td>
<td>envelope amplitude</td>
<td>-</td>
</tr>
<tr>
<td>LFOs</td>
<td>retrigger, envelope amplitude</td>
<td>envelope amplitude</td>
<td>-</td>
</tr>
<tr>
<td>Step Sequencer</td>
<td>trigger next note, note amplitude</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Filter</td>
<td>parameter envelope, peak parameter value</td>
<td>parameter value</td>
<td>-</td>
</tr>
</tbody>
</table>

5.1.2 **Pitch Mallet**

This object uses the more advanced feature of detecting the approximate hit magnitude and position. The width of the sensed tabletop perimeter depends on the object rotation and goes from a minimum of 1/16 of the perimeter to the full perimeter. This allows to focus the interaction area from a single sensor up to the whole table. This approach also allows for cooperative playing. Quantization of the estimated hit position can be enabled, or in alternative, the plain computed value can be used. The behavior of this object regarding after-touch handling is performed in
a way analogous to the *Envelope Trigger* object, with the difference that also the information regarding the angle of the contact point is used as a control parameter. The visual feedback associated with the object shows a corresponding virtual keyboard and a button allows to enable or disable quantization. The way this object interacts with the already existing ones is shown in table 5.2.

Table 5.2: Pitch Mallet - Control of other Reactable Objects

<table>
<thead>
<tr>
<th>Object</th>
<th>Impact</th>
<th>Pressure</th>
<th>Angle</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oscillator</td>
<td>envelope amplitude</td>
<td>envelope amplitude</td>
<td>pitch</td>
</tr>
<tr>
<td>Sampler</td>
<td>trigger start</td>
<td>envelope amplitude</td>
<td>pitch</td>
</tr>
<tr>
<td>Loop</td>
<td>trigger start</td>
<td>envelope amplitude</td>
<td>head position</td>
</tr>
<tr>
<td>Scratch</td>
<td>retrigger, amplitude</td>
<td>envelope amplitude</td>
<td>speed</td>
</tr>
<tr>
<td>LFOs</td>
<td>retrigger</td>
<td>LFO depth</td>
<td>LFO speed</td>
</tr>
<tr>
<td>Step Sequencer</td>
<td>step, note velocity</td>
<td>stepping speed</td>
<td>step: backward - same - forward</td>
</tr>
<tr>
<td>Filter</td>
<td>maximum parameter amplitude</td>
<td>parameter value</td>
<td>dry-wet crossfade</td>
</tr>
</tbody>
</table>

### 5.1.3 Pressure-Angle Follower

Differently from the *Envelope Trigger*, this object has no fixed envelope and works as a contact point pressure and angle follower. The pressure is mapped to different control parameters depending on the connected object, as specified in table 5.3.

Table 5.3: Pressure-Angle follower - Control of other Reactable Objects

<table>
<thead>
<tr>
<th>Object</th>
<th>Impact</th>
<th>Pressure</th>
<th>Angle</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oscillator</td>
<td>-</td>
<td>amplitude</td>
<td>pitch</td>
</tr>
<tr>
<td>Sampler</td>
<td>retrigger</td>
<td>amplitude</td>
<td>pitch</td>
</tr>
<tr>
<td>Loop</td>
<td>-</td>
<td>slow down - stop head</td>
<td>head position (scratch)</td>
</tr>
<tr>
<td>Scratch</td>
<td>-</td>
<td>speed</td>
<td>scratch time offset</td>
</tr>
<tr>
<td>LFOs</td>
<td>retrigger</td>
<td>LFO frequency</td>
<td>-</td>
</tr>
<tr>
<td>Step Sequencer</td>
<td>amplitude</td>
<td>step sequencer speed</td>
<td>-</td>
</tr>
<tr>
<td>Filter</td>
<td>-</td>
<td>cutoff</td>
<td>volume</td>
</tr>
</tbody>
</table>
5.1.4 Collective Control Object

This last object uses the total pressure value from all the sensors to control a high control-range sound synthesis algorithm. The resulting sound is not just the sonic sum of each performer’s contribute, but rather of the overall global interaction needed to wholly control the sound synthesis parameters. By rotating the object it is possible to switch between different custom synthesis algorithms.
Chapter 6

Conclusions and Future Work

The implemented hardware and software system opens the door to the development of a richer and more enticing musical experience when playing the Reactable musical instrument. The carried research has been based on a tight loop between the tuning of the implemented hardware and software, and personal testing of their effectiveness.

6.1 Using a more performing hardware solution

As previously said, while the use of the Arduino platform allowed for the implementation of the basic signal processing and control algorithms, limitations have been found due to the latency caused by the serial data transmission system adopted by the platform. For this reason better hardware alternatives will have to be tried. Possible solutions have already been found offering higher sensitivity, frame-rates, and lower latencies for a comparable price.

6.2 Incorporating the new Objects into the existing Reactable application

Once the above mentioned issues are fixed, the next step will be to implement the new Reactable objects as described in chapter 5. This step will require to gain an in-depth understanding of the whole Reactable software. Incorporating the new objects will require to modify the Pure Data synthesis system architecture, adapting the already existing objects to accept as input the new available information. Also the OpenGL visual synthesis engine and the Java connection manager will have to be modified accordingly.
Bibliography


BIBLIOGRAPHY


## List of Figures

2.1 A typical Reactable session. The connections carry the dynamic visual information about the waveforms which are traveling from a node to the other before of reaching the central sound output “sink”. .......................... 9
2.2 Schematic view of the Reactable system’s working. ......................... 9
2.3 The original concept of TAI as musical interface (from [6]). ............. 10
2.4 The **Percussion Tray** with four acoustic sensors connected to the Presto Kit processing module (from [5]). .................................................. 10
2.5 Representation of the FTIR technique’s working, as proposed by Han. .. 11
2.6 The pixel intensity grows as touch pressure grows. The left image is a zero-force touch, center is a light press, and right is a hard press (from [38]). 12
2.7 Composing elements of an Interlink FSR. ...................................... 12
2.8 The system developed by Schmidt. For their higher precision and weight tolerance, strain gauges are used. ................................. 13
2.9 Two of the IFSR-based touch-pad systems. ............................... 14
2.10 On the left the sensor multiplexing circuitry based on a set of shift registers and on the PIC24H micro-controller. On the right a schematic view of the implementation of the sensing array, based on orthogonal sets of conductors separated by an FSR material. ............................. 14
3.1 Implemented system architecture.................................................. 16
3.2 The used FSR (Interlink, Part No. 406 – 1.5” Square) and its relative force-resistance curve. .......................................................... 16
3.3 View of the sensing perimeter surface and detail of one of the sensing parts together with the expanded polyethylene elastomer. ............. 18
3.4 The voltage divider circuit (one of the two resistors is the FSR of which we want to know the resistance). ........................................ 18
3.5 The used buffered voltage divider circuit. ..................................... 19
3.6 The used Arduino 2009 board. .................................................. 20
3.7 The built prototype circuit together with the Arduino programmable microcontroller. ................................................................. 21
3.8 Diagram showing the software architecture of the implemented system. 21
3.9 Peak detection algorithm functioning. ........................................ 22
3.10 Pressures detected for a moving press along the table perimeter. ....... 24
3.11 The resulting 2D information for the above data. In red $G$, in green the corresponding computed angular position. The blue lines show the corresponding $G - \alpha_{touch}$ pairs. ........................................... 24

3.12 The prototype application user interface showing the detected forces. On the left pressing on a single point, in the middle over a wider area. On the last image the overall magnitude of the applied force is shown together with the corresponding computed center of pressure (square) and angle (triangle). ........................................... 25

3.13 Pressures detected for some fast hits on one side of the table. The gray dashed lines show the orientation of the sensors axes. ......................... 26

3.14 Resulting trajectories of $G$ (from blue to red in time). ......................... 26

4.1 View of the implemented Pure Data patch. The achieved frame rate of 120 frames per second per sensor is visible on the left of the patch. on the right the part of the patch used for data logging and further data analysis. ................................................................. 28

4.2 Pressure graph for 3 independent pressure points. ......................... 28

4.3 Plot of the recorded gesture-to-sound latency. ......................... 29

5.1 The proposed symbols for the new objects. ................................. 31
List of Tables

5.1 Envelope Trigger - Control of other Reactable Objects . . . . . . . . . . 32
5.2 Pitch Mallet - Control of other Reactable Objects . . . . . . . . . . . 33
5.3 Pressure-Angle follower - Control of other Reactable Objects . . . . . 33