Turning music interaction into a multisensory game-like experience

Domenico Stefani, domenico.stefani@studenti.unitn.it, MAT: 203317
Francesco Trebo, francesco.trebo@studenti.unitn.it, MAT: 198899

Abstract

Getting into music can sometimes be challenging, and especially at the beginning when players have to become familiar with basic musical principles, the training can be boring and discourage people making them lose interest into the activity. Here comes the idea of a system which can introduce basic concepts like tempo or simple sound effects in a game-like experience for two people. What makes this project unique is the idea of combining two different interfaces, one for audio and one for video, which are operated by one player each. Only by being focused into the music and by paying attention to the changes in both sound and video, the two players can experience the best results. The evaluations seem to indicate that players, especially non-musicians, benefit from the game-like experience to improve their awareness of their performance with the music synthesizer.

I. Introduction

This project describes the ideas, development and first evaluation of an interactive musical device. The aim of this project is to understand whether the playful interaction between two people and a particularly devised musical device can help introducing non musician players to basic musical concepts like tempo or sound effects.

When speaking of getting into music playing, having to own an instrument has not been a blocking factor since mass production allowed to have a simple guitar or flute in almost every house, but the learning curve on many of these instruments can be as rewarding as steep. This often leads to difficulties when first starting to play an instrument and the player, especially young ones, can lose interest in the activity.

Our project proposes a game-like music experience, designed to be played by two people. The designed device allows the generation of music through its audio interface and matching visual effects on a screen through its video interface.

The audio interface features a simple step sequencer and audio synthesizer along with several controls that allow to modify features of the produced sound. The video interface is composed of a game controller and a program that mixes information coming from it and from the audio interface, generating visual effects that can be shown on a screen or a projector.

The game controller features an analog joystick, an accelerometer and a vibrating motor to provide haptic stimuli to the user. The hypothesis is that non-musician users, through the video interface, will benefit from having to listen actively to music, interact through the controls and receiving haptic stimuli and that the experience recorded can help having a playful and serene interaction with the more complex audio interface.

The area that explores playful music interaction is interesting because it includes a class of possible devices, software programs or instruments that can be educational for everyone but also recreational for musicians and experts

The projects itself is based on interactive sonic experience design and haptic perception (specifically SA-RA mechanoreceptors).

II. Related work

One of the works that inspired part of this project is the Dato Duo music synthesizer by Dato Musical Instruments [1] which is a very accessible “synth-for-two” designed with simplicity in mind and meant

to be played by two people. It features a synthesizer side that contains controls that shape the sound produced and a sequencer side that allows to act on the rhythm and the melody produced.

The proposed device is inspired to the successful two-people approach but diverges on the sensory modalities used, trading off the simplicity of a sound-only experience with a device that provides audio, video and haptic interaction in order to explore a different interaction, more similar to a video game.

A similar natural graphic approach, even if on a more reasonable scale for a music synthesizer, is used by Teenage Engineering in their OP-1 and Pocket Operator line-up [3] which include decorational but functional graphical elements to their interfaces.

On the software side there are many games and programs trying to teach music basics to young children but there are also good examples of how a game-like approach can be effective with music students that need to familiarize with more structured music theory concepts. One example is “Lord of the chords” [2] , a successful card game based on chords that was highly praised by music students and teachers.

All this products and projects are partially similar to the proposed device, some in the modalities and some in the motivations, but our approach has not been widely used yet and it is worth exploring.

III. Architecture design

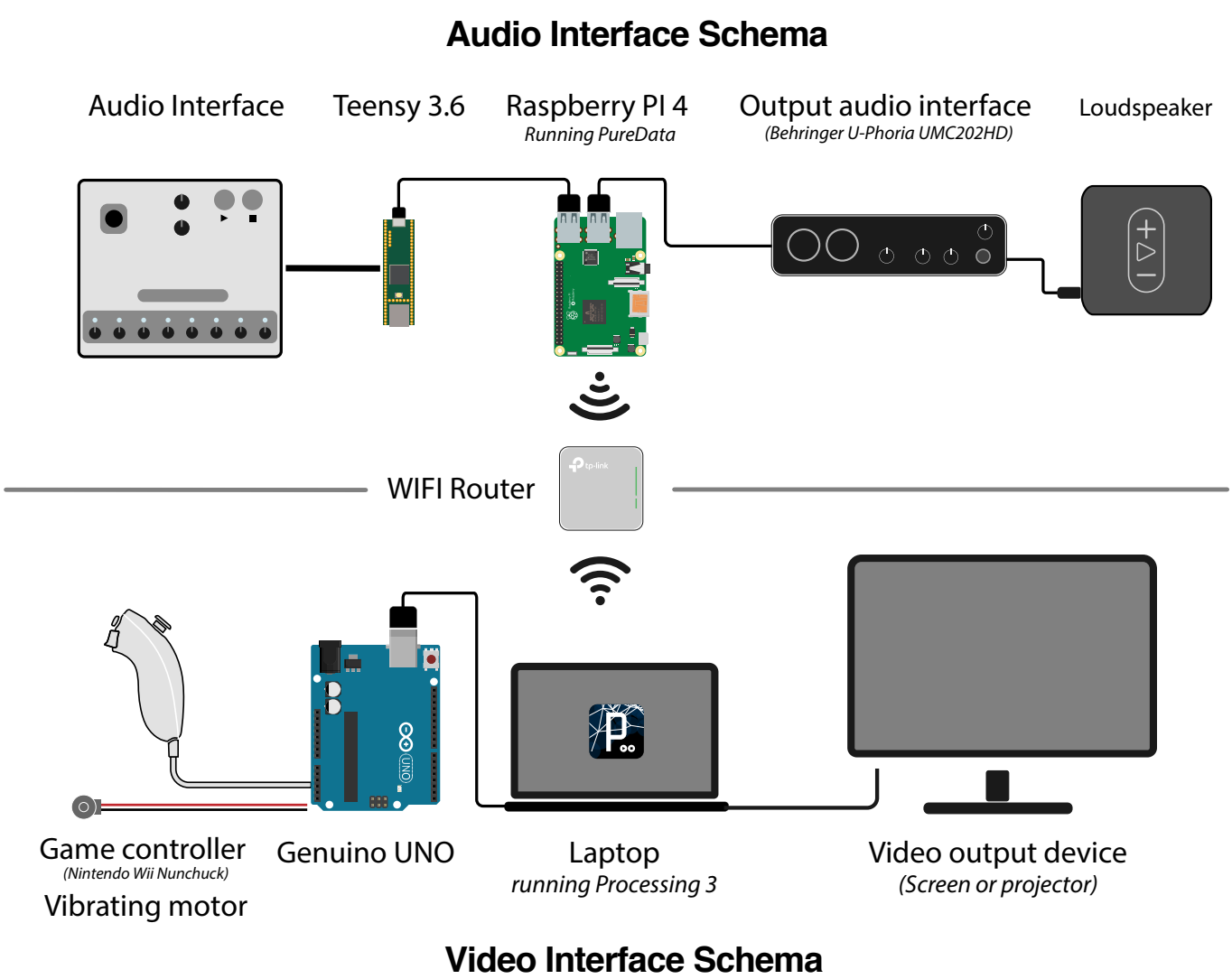

Fig. 1. Architecture design schema

The physical device is divided in an audio and a video section.

Audio Synthesizer

The Audio Interface is composed by a control interface with:

- 8 potentiometers for each note of the sequence played.
- 8 LEDs, one for each note potentiometer, indicating the current step of the sequence.
- 1 analog joystick controlling:
  - On the horizontal axis, the amount of detune between two superimposed waveshapes and the amount of delay feedback applied.
  - On the vertical axis, the shape of the sound waveform (morphing from a soft triangle wave to a square wave).
- 1 potentiometer controlling the cutoff frequency of a low-pass filter.
- 1 potentiometer controlling the tempo of the sequence.
- 1 Ribbon sensor controlling the note sustain through an envelope generator.
- 2 Buttons for Play and Pause.

Video Synthesizer

The video synthesizer consists in a laptop and a Nintendo Nunchuck Controller that features a three-axis accelerometer for motion-sensing and tilting and an analog joystick, connected through an Arduino UNO board.

The Arduino board converts the movements of the joystick into pulses, whose tempo is compared to the music BPM. A rumble motor, connected to the joystick, provides the user a tactile stimulus of the music tempo.

The laptop runs a Processing3 Script which reads messages from Arduino and from the Audio interface and uses both to generate visual effects on the screen. The laptop can also be connected to a bigger monitor or projector in order to produce a more immersive experience.

The choice of a good spot plays a critical role in the quality of the experience. A quiet room must be chosen in order to allow users to focus their attention on the generated sound. The presence of a big screen or projector could make the experience more immersive. Users of all ages must be willing to enjoy a game-like experience with the goal of achieving the best result possible, by keeping in mind that, being this the first prototype iteration, the interfaces must be used with particular care.

A. Usage model

The system is meant to be used by two people: one playing with the Audio interface and one with the Video interface.

Audio interface

Note that in-depth knowledge of the mapping between sensors and actions performed is not mandatory in order to enjoy the system, however knowing more can help exploring special sound features or effects. The audio components, as indexed in fig. 2 are:

1) Analog Joystick: It can be moved along the vertical and/or horizontal direction and it controls different sound features, namely:
   - Waveshape - vertical direction: The middle position indicates a 50% balance between 2 oscillators producing a Triangle wave and a Square wave.
     Moving the stick towards the top position increases gradually the triangle wave (soft) contribution, while moving it toward the bottom increases the contribution of the square wave (harsh)
   - Detune - Horizontal-Left direction: Moving the stick to the left increases gradually the frequency gap between the main oscillator and a second one (detuned oscillator). This is done at the same time for the two waveshapes (so 2 main oscillators, 2 detuned oscillators).

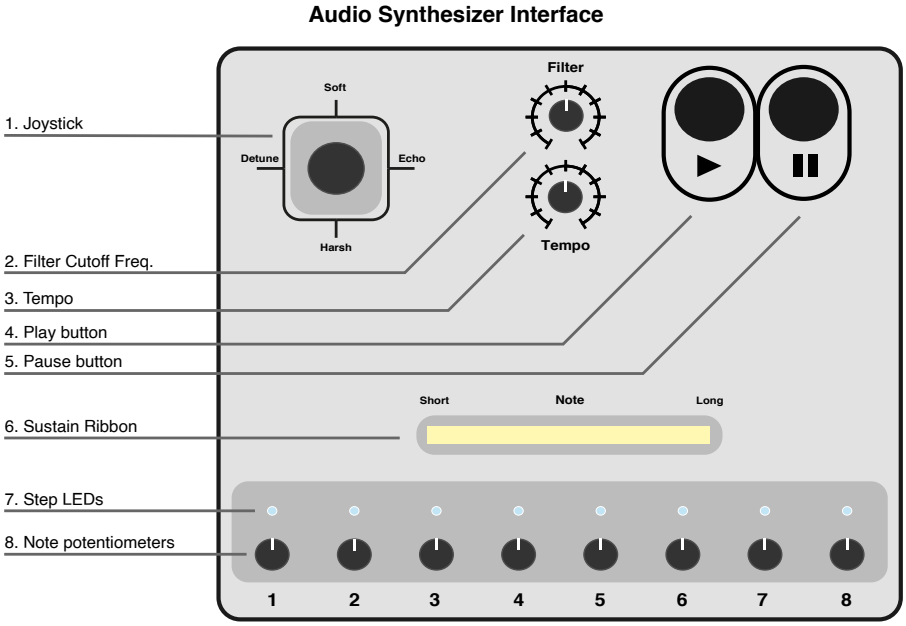

Fig. 2. Audio interface components

- Delay Feedback - Moving the stick to the right increases the amount of delay feedback added to the sound.

2) Filter Cut-off Frequency knob: It controls the cut-off frequency of a low-pass filter.
3) Tempo knob: It controls the speed of the note sequence.
4) Play button: When pressed, the note sequence starts playing.
5) Pause button: When pressed, the note sequence stops playing.
6) Sustain ribbon/soft-pot: This is used by sliding a finger over the sensitive area (white). Sliding the finger to the left shortens the note sustain while sliding it to the right makes it increase.
7) Step LEDs: The LED turned on indicates the current playing note.
8) Note potentiometers: The 8 potentiometers control the 8 notes of the played sequence, varying the sound from low pitched notes (far left) to high pitched ones (far right). The notes are quantized on a diatonic scale, so that it's harder to produce dissonant tunes.

All the controls modify the produced sound and some of them affect also visual features generated by the video interface.

Video Interface

The video interface is composed of a computer, eventually connected to an external monitor or projector, and the game controller. The controller (fig. 3) features:

1) Analog Joystick, which controls some visual effects;
2) 2 Buttons: The buttons are part of the controller, but they are not used for this application. Pressing either buttons has no effect.
3) Accelerometer: it measures the movements of the game controller. Shaking the controller vertically at the rhythm of the beat will improve the amount and the quality of visual effects, colours, etc.
4) Rumble Motor, which vibrates at the rhythm of the beat.
   Different combinations of the user actions can produce an increase or decrease of the variety, colourfulness and speed of the visuals.

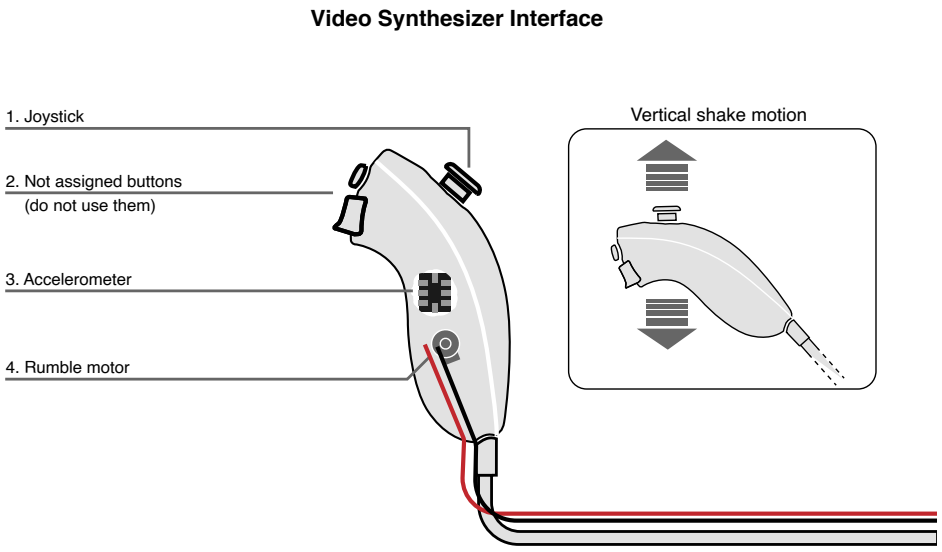

Fig. 3. Video interface components

IV. Implementation

The implementation details of the system are described separately for the *audio* and *video* interfaces.

Audio Interface

The audio interface consists in a custom designed desk with several sensors and LEDs, connected to a Teensy 3.6. The board manages sensor reading and sends/receives messages to/from a Single Board Computer (Raspberry PI 4) through serial communication.
The Raspberry PI runs a Miller Puckette's Pure Data patch, with different functions:

- Audio synthesis
- Serial Communication with the Teensy board
- OSC (Open Sound Control) Communication with the video interface, via a local Wi-Fi network.

The synthesizer structure (fig. 8) is the following:

A metronome is used to advance a 8 note **step sequencer**. The **note block** provides the frequency values obtained converting the sensor values read from the serial port, the **vco block** contains the 4 oscillators (2 detuned squarewave oscillators and 2 detuned trianglewave) and mixes the various contributions based on the detune and shape parameters, the **vcf block** contains the low-pass filters and uses the cutoff parameter, the **envelope block** contains the ADSR envelope generator triggered by the gate signal and the amplifier, the **effects block** contains the delay and uses the delay parameter for the feedback. The output device is a generic audio interface connected with a loudspeaker.

The Wi-Fi LAN is created using a Tp-Link wireless router and exploits the 5GHz frequency band when possible, for faster communication.

The Teensy board reads the analog voltages from the sensors and the digital signals from the two buttons. The analog values are filtered using a low-pass Butterworth filter and sent as a serial message when they are changed. The play button signal triggers the Pure Data patch that starts the metronome and sends a serial message to the board when each new step is triggered. The board turns on the led relative to the current step every time that a new one is triggered.

Video Interface

A Nintendo Nunchuck Controller has been connected to an Arduino UNO board via I2C communication. A specific library by G. Bianconi [4] has been used as reference, to decode the information provided by the controller. The Arduino controller has the task to compute the data from the accelerometer converting the movements of the controller into a numerical frequency of pulses. The obtained result in sent via a serial communication to the computer, where a Processing 3 sketch is running. Arduino receives from the computer a series of messages that are converted into short physical pulses of the rumble motor, which can be used as tempo reference from the user.

The motor has been strategically placed on the controller in order to touch a particular spot of the user hand: the frequency of the vibration impulse will not exceed 4Hz (240 bpm), so the position of the slowly adapting Merkel disk mechanoreceptors has been chosen, on the palm of the hand [7]. Moreover, the vibration frequency itself has higher frequency (130-180 Hz estimated) and the position has been chosen because it's also near to SA Ruffini cylinder mechanoreceptor that responds the best to vibration in the 15-400Hz frequency range.

The computer receives the tempo value by the audio interface via an OSC communication. By comparing that tempo with the one received by the Arduino, taking in account a given tolerance (Algorithm 1), it gives the user a reward or a penalty, which consist in an improvement or a decrease of the visual effects quality (colour, quantity, etc.). The program also indicates, by using a visual indicator, if the movement of the user is slower or faster than the audio one.

A challenging aspect of the hardware implementation of the visual interface was the number of asynchronous tasks to be performed by both the Arduino and the Processing sketch.

To keep the latency as low as possible, Serial communication of values between Arduino and the computer occurs only if the value changes and therefore has to be updated (Algorithm 2).

An interesting aspect to be taken into account, is the following: even if the tempo signal emitted by the audio interface has to pass through a series of steps before being converted into vibration (OSC between raspberry and computer, processing sketch, serial between computer and Arduino), the time required is shorter than the latency of the audio DAC, so a delay had to be implemented to synchronize audio with the tactile stimulus (Algorithms 3,4).

The graphical experience designed is composed by original ideas and some features inspired by existing projects, one of which being the 3D terrain Generation challenge implementation [5] created by Daniel Shiffman. It generates "peaks" or "mountains" using Perlin Noise [8], a procedural texture primitive, used to produce smooth and even natural appearing textures on computer generated surfaces. For the purpose of this graphical experience, it allows to have correlation in height among nearby peaks.

V. Evaluation

The tested *hypothesis* is that *users can benefit from having to listen actively to music, interact through the controls and receiving haptic stimuli (when using the video interface) and that the recorded experience can help in having a playful and serene interaction with the more complex audio interface*. An additional sub-hypothesis is that *the impact of this improvement is greater for non-musician participants*.

The game-like experience designed is an independent variable of the evaluation.
The participant selection is controlled by the researchers, thereby the mix of musicians and non-musician users can be defined as an independent variable, while the subjective intuitiveness and overall enjoyment of the experience are dependent variables. Testing was designed to be organized in sessions with 2 users per session where at least one participant was not a musician

Pilot Study

A pilot study was conducted with 4 participants and it consisted of the regular test, followed by an additional brief questionnaire about the experience and the questions asked. As suggested by participants, we decided to add more visual indicators on the video interface, to facilitate the user improvement.

Main Study

The main experiment was conducted with ten participants, nine males and one female, between 20 and 38 years of age (mean= 24.5, SD = 4.72) that gave their consent to take part to the evaluation session. 7 participants out of 10 were not musicians.

The test starts by allowing the two participants to take a sit and interact with one interface each, that from now on will be called the *main interface* of each player. After 5 minutes, if the participants are satisfied with the duration and the result of the first interaction, they swap interface and proceed by playing for 5 minutes more. Finally, the two users swap places one more time, so that both play again with their main interface, in order to record potential improvements in the interaction. Note that the audio interface was always assigned to a non-musician participant for the first (and third) interaction, in order to compare results after practice.

The *task* for the participant playing with the audio interface was to create interesting tunes, play with the interface and the controls while both listening to the music generated and looking at the video changing accordingly to their action.

The *task* for the participant playing with the video interface was to follow the rhythm of the music by shaking the controller accordingly and also use the analog control to modify the video.

After the experience, participants were asked to complete a questionnaire with the following questions:

- 2 questions regarding whether the participant is a musician or not and interest in music (using a *5 points Likert Scale*).
- 1 closed question asking what interface was firstly used by the participant.
- 3 questions evaluating emotional *valence, arousal and control*, designed using the *Self-Assessment Manikin* [6]
- 1 Open question about considerations on the *second interaction*.
- 3 questions using the *Self Assessment Manikin* [6] to evaluate the emotional response to the *third interaction*
- 1 Question with three *Visual Analog Scales* evaluating *intuitiveness*, *interest* and *pleasantness* of the final interaction.
- 1 *5-points Likert-scale* question, evaluating to what extent the improvement between the first and third interaction (if any), was to be attributed to the **experience with the alternative interface** (the interface used during the second interaction).
- 1 *5-points Likert-scale question*, evaluating to what extent the improvement between the first and third interaction (if any), was to be attributed to the **practice on the main interface**.
- 1 final open question asking to leave a comment about the experience and possible improvements.

Results only partially supported the first hypothesis, recording a mild improvement attributed to the experience across interfaces (avg. 3 out of 5 in the Likert scale, median:3 but high SD:1.78), while the contribution of the experience on the main interface itself was considered generally higher (avg: 4.3, median 4, smaller SD: 0.4)(fig. 4).

When comparing the contribution of the experience across interfaces for musicians and non-musicians, the averages are close but for the non-musician participants the variability is higher (fig. 5).

The emotional response satisfied our expectations and the comparison between first and third interaction is good too, with average valence decreasing ever so slightly but with *arousal and control increasing by a relevant amount* (fig. 6).

Finally, the bipolar attributes report good results with high interest from the participants in the interface, good comprehension and general pleasantness of the experience (fig. 7). Considerations on the answers to open questions and additional verbal comments expressed by the participants are discussed in the next section.

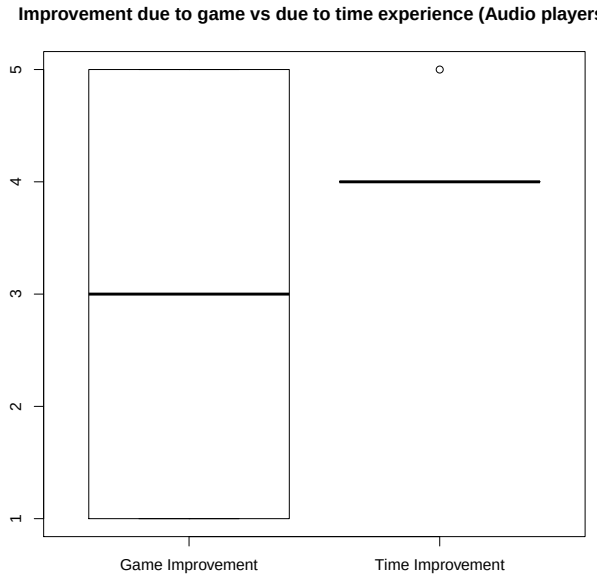

Fig. 4. Improvement attributed to the game experience vs improvement from practice with the main interface

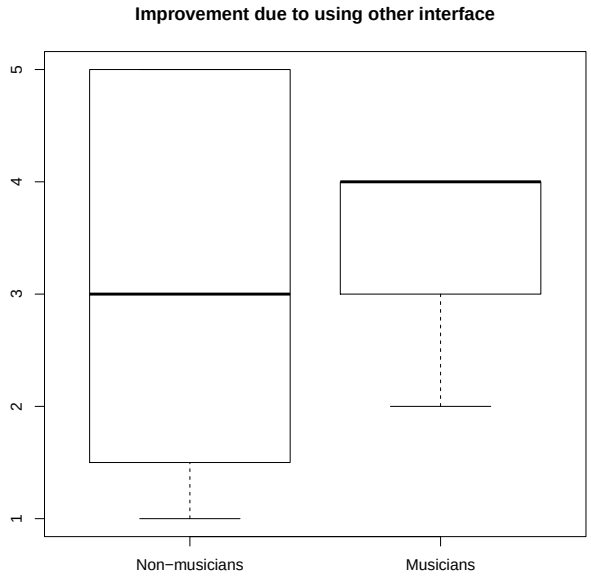

Fig. 5. Game improvement (due to experience with both interfaces) compared for musicians and non musicians participants

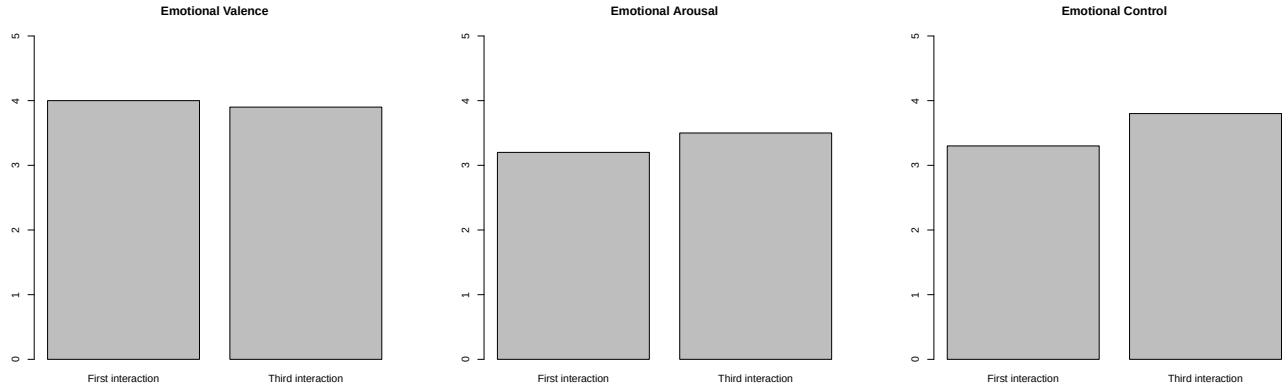

Fig. 6. Emotional response results compared between the first and third interaction

VI. Discussion and conclusions

A first general result is that the overall experience was considered pleasant, especially the interaction with the audio interface that offered many controls and direct feedback of their impact. One of the participants reported specifically that they "lost the sense of time because playing with the controls was a lot of fun" when playing with the audio interface, confirmed by the fact that, when asked to swap places as soon as the two participants were satisfied, they took 3-4 minutes to actually swap. The interaction with the Video interface was received as generally interesting too. One participant reported "this can be really useful to me as I'm not able to tap my foot following music rhythm".

The results show that the interaction across interfaces was useful, even if less than expected, in order to be engaged in the game experience and, more interestingly, non-musicians seemed to be more involved thanks to the game-like experience. This confirms the second sub-hypothesis and partially the main one, however a reliable analysis should include more test sessions.

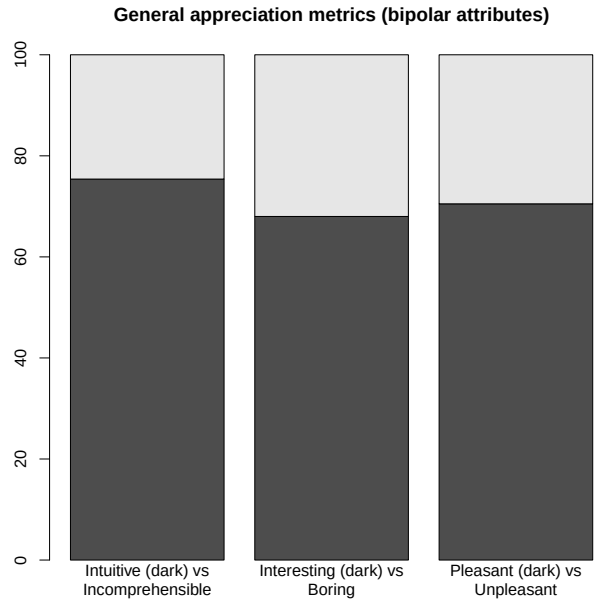

Fig. 7. Values recorded from three VAS

We have learned many lessons, one of all being that evaluation with users is complex and designing a questionnaire that provides informative results while still being rather intuitive to fill is not an easy task. If we had to proceed with a second design iteration, we would improve the controller of the video interface with a custom one, in order to improve the interaction with the analog joystick that, according to some users, was difficult to use while moving the controller constantly. On the audio interface we would replace the ribbon sensor with a more intuitive control (e.g. a physical slider) and further improve the sound design. The tune of the rumble motor could be improved, especially at high BPM: while it was considered an useful aid at slow speed, some users found it a bit confusing at higher tempo.

Another consideration is that the overall refinement of the experience tends to outweigh the concept, meaning that for users it's difficult to overlook the roughness of a prototype and consider the concept if the latter is not extremely novel.

a) Group members contributions: We both contributed to the idea of the project, the report and the evaluation study including preparing all the material needed. Francesco Trebo managed mainly the design and development of the video interface while Domenico Stefani designed and developed the audio interface, however we worked together to solve some issues and introduce some features to both interfaces.

REFERENCES

[1] Dato DUO from Dato Musical Instruments - *https://dato.mu/*
[2] Lord Of The Chords card game - *https://www.lordofthechords.com/*
[3] Teenage engineering - *https://teenage.engineering/products/synthesizers*
[4] G.Bianconi, ArduinoNunchuk project - *https://github.com/GabrielBianconi/arduino-nunchuk*
[5] Daniel Shiffman, 3D Terrain Generation with Perlin Noise in Processing - *https://thecodingtrain.com/CodingChallenges/011-perlinnoiseterrain.html*
[6] Bradley, M. M., & Lang, P. J. (1994). Measuring emotion: The self-assessment manikin and the semantic differential. Journal of Behavior Therapy and Experimental Psychiatry
[7] Gardner, E. P. & Martin, J. H. (2010) *Coding of Sensory Information* https://www.researchgate.net/publication/265246764_Coding_of_Sensory_Information
[8] Perlin, K. (1985) *An image synthesizer*, ACM SIGGRAPH Computer Graphics

VII. CODE APPENDIX

Algorithm 1: boolean isBpmGood(int bpm, int user_bpm)

```
float tolerance = bpm * relative_bpm_tollerance;
if (abs(bpm*2 - user_bpm) < tolerance) then
    return true;
end
if (abs(bpm - user_bpm) < tolerance) then
    return true;
end
if (abs(bpm/2 - user_bpm) < tolerance/2) then
    return true;
end
return false;
```

Algorithm 2:

```
if (nunchuk.analogX != prevXValue) then
    Serial.println("X-" + String(nunchuk.analogX));
    prevXValue = nunchuk.analogX;
end
```

Algorithm 3: serialEvent()

```
while Serial.available() do
    Serial.read();
    digitalWrite(motorPin, LOW);
    t_tempo_received = millis() + audioSync;
end
```

Algorithm 4: loop()

```
[...]
if (currentMillis > t_tempo_received) then
    digitalWrite(motorPin, HIGH);
end
```

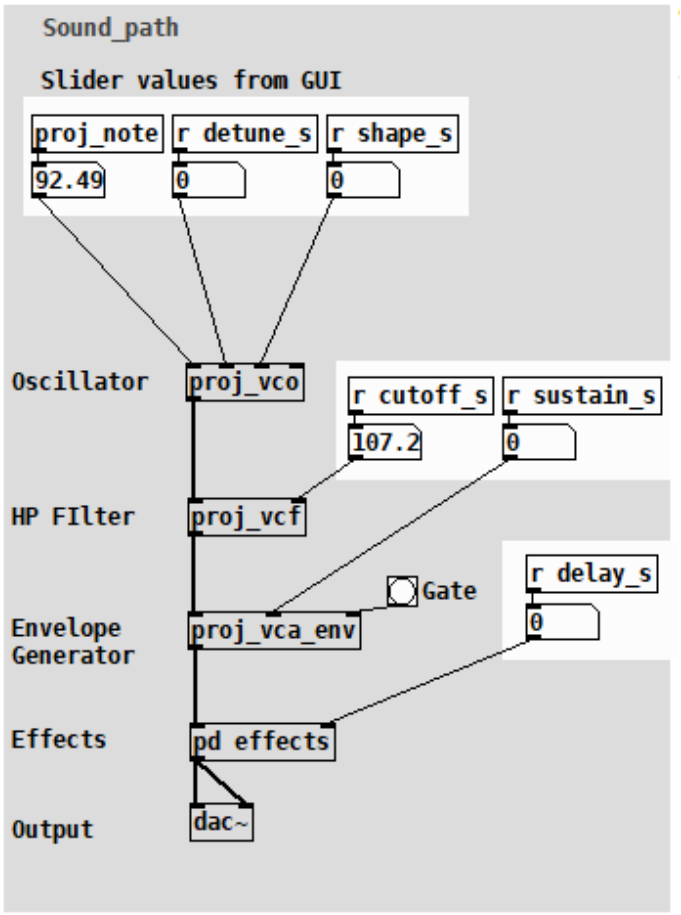

Fig. 8. A section of the Pure Data synthesizer patch representing the main blocks of the sound path.

The complete code, along with all the project material, is available on GitHub at `github.com/ftrebo/mis1920`