Interacting with digital audio effects through a haptic knob with programmable resistance

INTERACTING WITH DIGITAL AUDIO EFFECTS THROUGH A HAPTIC KNOB WITH PROGRAMMABLE RESISTANCE

Yuri De Pra

Department of Computer Science,
Mathematics and Physics
University of Udine
Udine, Italy
yuri.depra@uniud.it

Federico Fontana

Department of Computer Science,
Mathematics and Physics
University of Udine
Udine, Italy
federico.fontana@uniud.it

Stefano Papetti

Institute for Computer Music and Sound Technology
Zurich Univeristy of Arts
Zurich, Switzerland
stefano.papetti@zhdk.ch

ABSTRACT

Live music performances and music production often involve the manipulation of several parameters during sound generation, processing, and mixing. In hardware layouts, those parameters are usually controlled using knobs, sliders and buttons. When these layouts are virtualized, the use of physical (e.g. MIDI) controllers can make interaction easier and reduce the cognitive load associated to sound manipulation. The addition of haptic feedback can further improve such interaction by facilitating the detection of the nature (continuous / discrete) and value of a parameter. To this end, we have realized an endless-knob controller prototype with programmable resistance to rotation, able to render various haptic effects. Ten subjects assessed the effectiveness of the provided haptic feedback in a target-matching task where either visual-only or visual-haptic feedback was provided; the experiment reported significantly lower errors in presence of haptic feedback. Finally, the knob was configured as a multi-parametric controller for a real-time audio effect software written in Python, simulating the voltage-controlled filter aboard the EMS VCS3. The integration of the sound algorithm and the haptic knob is discussed, together with various haptic feedback effects in response to control actions.

1. INTRODUCTION

Vintage analog synthesizers and effects are increasingly fascinating musicians and producers around the world, so that a continuous effort is being made by the audio industry to reproduce those machines as software replicas. Besides providing their sought-after sound, these replicas must also reproduce the original layouts made of elements such as knobs, sliders and buttons. While hardware reproductions can copy the physical layout, software simulators usually provide a graphical user interface (GUI). In spite of the growing diffusion of touchscreens, one of the most appreciated features of hardware audio technology consists in the tangibility of the available controls. This feature improves user interaction for two main reasons: first, physical controls allow precise parameters manipulation and, secondly, they can be effectively operated also when visual attention is focusing elsewhere, contrarily to what can be done with GUIs, e.g. on touchscreens. In this regard, the benefits of additional haptic feedback on touchscreens have been recognized for audio mixing [1].

Virtual Studio Technology (VST) plugin containers allow software controls to be mapped onto external hardware via DAW-level automation, MIDI or Open Sound Control (OSC) protocols. Hardware MIDI controllers usually distribute a predefined number of sliders, knobs and pads on static layouts. Obviously, such layouts cannot offer one-to-one control mappings suitable for every software product. More advanced controllers offer two-way communication to provide visual feedback via e.g. LED or LCD screens displaying the current assignment and state of parameters, whose value may not correspond to the actual position of a physical control. Controllers with motorized elements (e.g. faders) also exist, whose position depend on the value of the mapped parameters [2]. Although such elements are just meant to change their position (e.g. according to pre-recorded envelope tracks), they may be also used interactively to provide haptic feedback [3]. Similarly to sliding faders, rotary knobs could also be motorized to enable haptic feedback.

In this paper we deal with *resistive* (i.e. passive) force feedback as a means to mark-up specific positions in controllers. A common example is offered by the left/right balance slider/pot found on home stereo amplifiers: through physically marking the mid position with a point-wise change of the resistance operating in both sliding/rotating directions, it was far easier for the hi-fi listener to reset the stereophony to the center position. Standard knobs provide constant resistance to turning, and sometimes add detents emphasizing regular scale points or specific positions depending on the encoder's mechanical design. Our research goal is to freely program the resistance offered by such physical controls.

A haptic knob was designed with programmable resistive force based on electromagnetic technology, which can render different haptic effects. The proposed solution results in a significant reduction of costs, weight, space and power consumption as compared to existing products (see Sec. 2). We present the design of various haptic effects and their possible binding with physical controls. Furthermore, we report on the results of a user study consisting in an abstract target-matching task which compared visual-only and visual-haptic feedback conditions to assess the effect on performance of the proposed resistive feedback. Finally, as an application, the haptic knob was used to control a digital model of the voltage-controlled filter (VCF) from the EMS VCS3 synthesizer.[1] To this end, we developed a software framework in Python implementing both haptic feedback effects and the filter algorithm.

[1] https://en.wikipedia.org/wiki/EMS_VCS_3

2. RELATED WORK

Rotary controllers have been researched from different perspectives: ergonomic studies have defined guidelines concerning knob sizes and shapes [4, 5], while other studies focused on user interaction aspects [6]. Park *et al.* found that haptic knobs are more fun and interesting than standard input devices like mice and keyboards, moreover improving the focus of the user on the main task [7]. Experiments comparing physical and virtual knobs have highlighted that tangibility has positive effects on several aspects of interaction [8]. For instance, it was proved that tangible control allows for better performance in terms of error rate and interaction speed: interaction with physical knobs was found to be 20% faster than with their virtual counterparts and, additionally, subjective performance remained unaltered also when one's visual attention focused on a different task [9]. These results are especially relevant when a knob is used as a multi-parameter controller, integrating browsing and selection of items through turn-and-click actions. Also for this reason, some recent smart gadgets incorporate or offer additional physical knobs: examples include Google Nest [10], an intelligent touchscreen-based thermostat, and the Bluetooth multimedia controllers Griffin PowerMate [11] and Microsoft Surface Dial [12].

In the last decade product designers have furthermore started to "brand" tactile cues [13]. Thus, haptic feedback design is going beyond a traditional process linking mechanical features (e.g. torque, number of resistive detents) to functional behavior [14, 15], and is progressively embracing the idea of giving a unique "feel" to a machine interface. For instance, the EMS VCS3 synthesizer became popular also for its unique joystick controller, at that time not found elsewhere in the market of electronic musical instruments. Such joystick, assignable through the synthesizer's own routing matrix, enabled an especially creative interaction with several sound processing components, including the VCF simulated in this work.

Force-feedback rotary controllers can be found in many contexts, including car dashboards, piloting systems, audio/video editors, robot controls, medical devices, household and professional appliances. Most of them make use of DC motors to generate force feedback [16, 15, 17]. The use of such motors enables complex actuation, such as bouncing effects, at affordable costs. Hybrid solutions that combine motors and brakes have been proposed as well [18]: while they allow the design of previously unavailable subtle effects, additional components increase complexity, cost and size of the hardware. Even more advanced and expensive solutions make use of magneto-rheological fluids to generate variable torques [19, 20]: in this case, magnetic fields variations are used to change the density of a fluid in which the knob shaft is immersed, allowing the precise control of torque up to end-stop simulation. Although variable-resistance knobs have been available for many years through the technologies mentioned above, their relatively high cost, complexity and encumbrance still limit their effective use in commercial products. Specifically concerning the music market, the Traktor Kontrol S4 is a DJ console offering a jog wheel which incorporates programmable haptic feedback technology [21]. A further interesting project is the Torquetuner [22], a rotary device with programmable force-feedback based on a DC motor that can render various haptic effects for use on digital musical instruments.

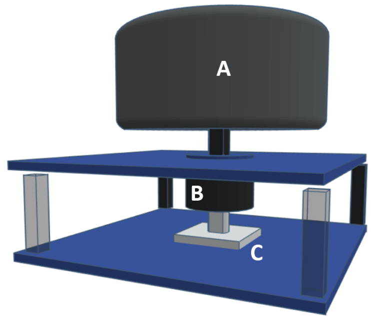

Figure 1: Schematic of the knob controller: the end-effector (A) is connected to the encoder (C) by means of the electromagnetic braking system (B).

3. DESIGN

Our programmable knob offers variable mechanical resistive force which, in response to active rotation imparted by the user, induces haptic illusions of active force feedback. Mechanical resistance is generated by an electromagnetic clutch used as a braking system. As described in [23], the other main hardware components are a rotary encoder and a microcontroller used to drive the electromagnetic braking system through one of its pulse-width modulation (PWM) output. To this end, a single microcontroller can manage multiple haptic knobs, depending on the available number of PWM outputs: for instance, the Arduino NANO that we used in our prototype can manage up to 6 programmable knobs. The cost of the knob hardly exceeds 40 Euros and, with large-scale manufacturing, it can be easily kept within 10 Euros, especially if multiple knobs are controlled with a single microcontroller. The specific hardware architecture and the control algorithms have been filed to the Italian Patent Office (IPO) [24].

Compared to existing solutions, the use of an electromagnetic braking system has advantages and drawbacks. The main advantage is the possibility to couple a standard knob with an encoder using the braking hardware as a connection point, as shown in Fig. 1. Other benefits are the low weight, thickness and cost compared to similar designs based on DC motors [16]. Power consumption is also reduced compared to DC motors, enabling battery-supplied portable solutions. The main shortcoming of our system is the lack of active force feedback, which limits the available haptic effects to perceived changes in torque, selection of detents, and definition of endpoints.

Although the knob provides resistance to rotation in both directions, the system takes advantage of the unidirectional shift performed by users during their action. Its control algorithm in fact determines the strength and duration of the resistive force based on the encoder position and the estimated rotation speed. The encoder position is sampled at fixed frequency rate and it is used to determine the speed of the rotation. The current prototype is equipped with a magnetic encoder having 4096 points per revolution, resulting in a resolution of 0.1 degrees per step.

The output voltage of the microcontroller (between 0 and V_{CC}) depends on the relative length of the PWM duty cycle. For example, if a constant torque is set, the output is always active with

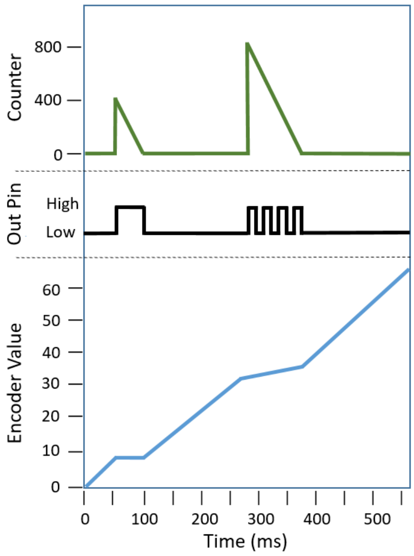

Figure 2: Temporal evolution of control variables for detents simulation: when the encoder position (blue line) reaches a detent, force feedback is activated by switching on the output signal (black line) until the counter (green line) is reset to zero. The PWM values of the effects are 100% and 50% of the duty cycle, respectively generating a proportional force.

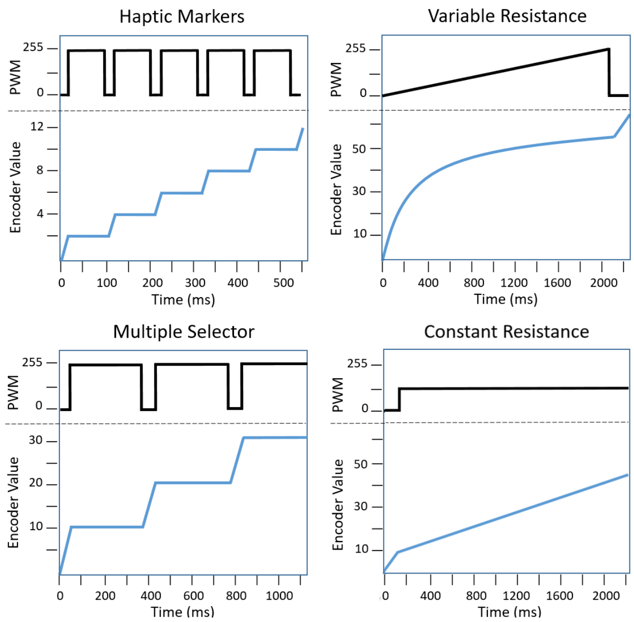

Figure 3: Example of different haptic feedback effects for a constant torque input.

a voltage proportional to the selected torque.

The available force-feedback effects include detents: once the encoder hits a position corresponding to a detent, the algorithm sets the PWM duty cycle and a counter depending on the programmed resistance and estimated rotation speed. The electromagnetic brake remains active (with a constant PWM value) until the counter reaches zero. The counter in its turn is decremented at every cycle. The deeper the detent, the longer the knob is blocked. Figure 2 shows the temporal evolution of the mentioned variables for two different detent effect examples. During the leftmost event, a soft detent stops hand movement for 50 ms, while during the second event the hand force is greater than the knob resistance, with consequently partial activation of the braking system slowing down the hand movement without stopping it.

Since a high frequency PWM control is needed to avoid the generation of audible noise, modulation was programmed using the low-level timer instructions offered by the Arduino microcontroller: for instance, using the 8-bit timer of Arduino NANO the highest PWM frequency is 16 MHz/255 = 62.7 kHz. Alternatively, it is possible to use methods such as those provided by the Arduino Motor Driver library,[2] which offers an abstraction of the timer for various microcontrollers.

Regarding input signals, the knob also features a button-press function resulting in 6 different control configurations: clockwise (CW) rotation, counterclockwise (CCW) rotation, short press, long press, press + CW rotation and press + CCW rotation.

The current prototype provides a metal cylindrical handle having diameter of 65 mm. The latter is hosted on a 3D-printed PLA structure (100x80x40 mm) embedding electronics and mechanical parts. However, these features can be adapted to the specific application (e.g. a smaller diameter of the knob if multiple controls must be provided by a device).

[2] `https://github.com/adafruit/Adafruit-Motor-Shield-library`

4. RESISTIVE EFFECT DESIGN

The resistive force feedback technology calls for a specific design of the haptic effects, too. Whilst the Torquetuner [22] makes available a series of haptic effects encoded as transfer functions between angle/velocity and torque, our knob needs to be programmed in the duration of the resistance to design many of the effects. For instance, the illusion of soft and hard detents is generated by locking the knob for different amounts of time. Some haptic feedback effects that can be designed with the proposed controller are listed below. The first two effects provide discrete information, whereas the third and fourth effect are continuous.

Haptic markers: movements along a graduated scale trigger resistance points simulating soft detents. The top-left plot in Fig. 3 shows the encoder position during 500 ms in presence of a constant torque input. The absence of resistance enables fast motion, resulting in the completion of multiple steps in a short time (steep segments); otherwise, the knob resistance almost stops hand movement for 100 ms (flat segments). This effect has a visual counterpart corresponding to a graduated slider or knob.

Multiple selector: movements across markers generate a resistive feedback simulating hard detents. The bottom-left plot in Fig. 3 shows the corresponding feedback: hard detents are rendered by almost stopping hand movement for 350 ms; otherwise, no resistance is generated. This effect has a logical counterpart corresponding to e.g. a set of radio buttons or a rotary selector. A common practice is to

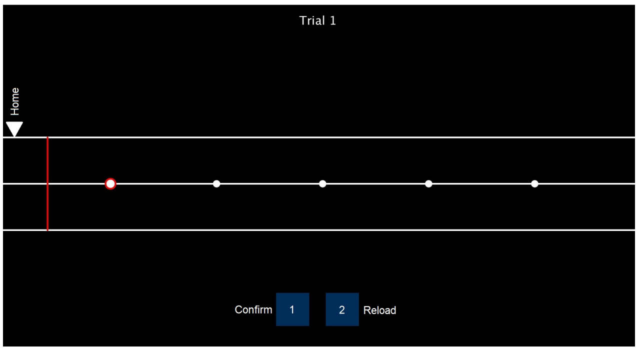

Figure 4: Graphical user interface used in the experiment.

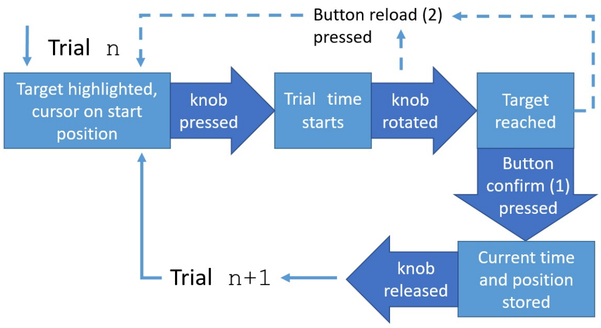

Figure 5: Test procedure.

use such controls to select source signals, waveforms, or to switch among audio channels.

Variable resistance: resistance is proportional to the encoded value (see the top-right plot of Fig. 3). This control can be used to linearly regulate effects' parameters (e.g. wet/dry, intensity, etc.). Additionally, sudden discontinuities of the resistance can be programmed to signal positions of the knob featuring some specific event (i.e. a binary switch following a parameter maximization, or a peculiar value of the same parameter).

Constant resistance: the resistance of the knob is constant, independent of the position (see the bottom-right plot of Fig. 3). Different values of resistance may be used to implement softer/harder knobs.

5. EVALUATION

For its general applicability to diverse contexts, not limited to the control of digital musical instruments or sound effects, the performance of the haptic knob was evaluated quantitatively through an abstract task rather than qualitatively in a musical production scenario. To this extent, we conducted a user study assessing the effect of haptic feedback on a target-matching task.

5.1. Setup and procedure

The haptic knob was connected to a computer running a GUI developed in Processing 3.5 (shown in Fig. 4) consisting of five

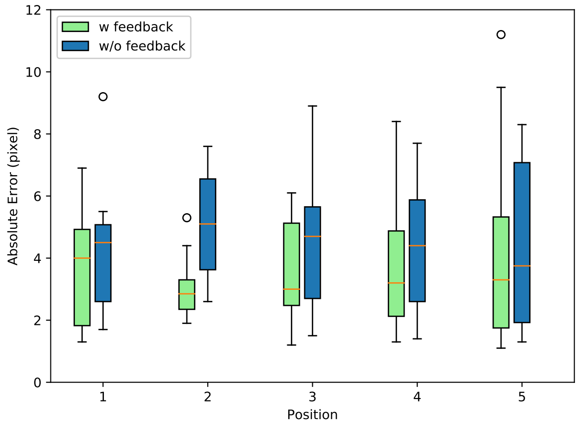

Figure 6: Box plots of the absolute error (mismatch) as a function of feedback condition and target position.

equally spaced round markers placed on a segment, a vertical red cursor, and two buttons for trial sequence control (1=Confirm and 2=Repeat, operated by the computer keyboard). The position of the vertical red cursor was controlled by the haptic knob with a ratio equal to 11.4 pixels per degree; the space between subsequent round markers, hence, amounted to about 320 pixels (i.e. 28 degrees).

The task consisted in rotating the haptic knob and move the vertical cursor until matching the center of the marker highlighted in red color. At each trial, a different marker could become red. Such task was performed under two conditions: i) Multiple selector resistance type (see Fig. 3) was conveyed every time a marker was crossed; ii) No haptic feedback was delivered. Five marker positions (Dot1–Dot5) were combined with such conditions (with or without haptic feedback) resulting in $5{\times}2{=}10$ factor combinations, each of which was repeated 10 times. The resulting 100 trials were randomly balanced.

Figure 5 describes the procedure followed in the test. At the beginning of each trial the cursor was positioned at the left side of the GUI ("Home" marker on the left side of Fig. 4); then, participants were asked to press the knob with their dominant hand and rotate it until matching the target. At that point, participants had to press the Confirm button (key 1) on the keyboard with their free hand. For each trial the execution time from knob press to Confirm and the mismatch, i.e. the distance between the target and the cursor position when Confirm was pressed, were recorded.

Ten people (7 males, 3 females) aged between 24 and 57 (M = 39.9, SD = 10.3), all right-handed, participated in the study on a voluntary basis and were not paid. Prior to starting the test, a training session consisting of 10 trials was offered, in which each factor combination was presented once in random order.

5.2. Results

Figure 6 and 7 respectively show box plots of the absolute error (mismatch) and time-to-match, for all factor combinations. Tests on the data distributions with the D'Agostino method [25] confirmed no significant deviation from normality.

A two-way ANOVA was conducted to study the influence of the two independent variables (position, feedback) on mismatch.

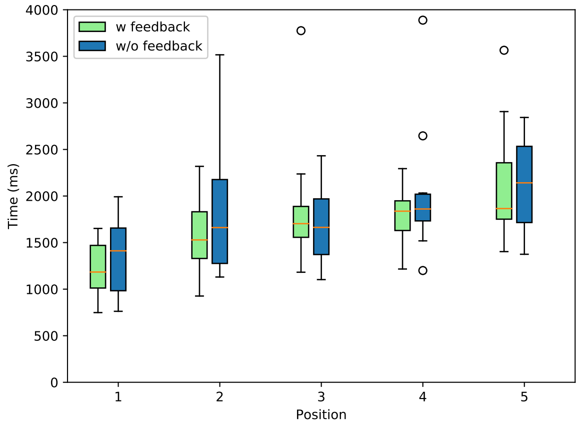

Figure 7: Box plots showing the time-to-match as a function of feedback condition and target position.

Position included five levels (1-5) and feedback consisted of two levels (with and without haptic feedback). Using a Greenhouse-Geisser correction for insphericity , the main effect for feedback yielded $F(1, 9) = 13.4$, $p < .005$, indicating a significant difference between trials in presence of haptic feedback ($M = 3.7$, $SD = 0.31$) and without it ($M = 4.54$, $SD = 0.31$). The main effect for position yielded $F(1.5, 13.6) = 0.2$, $p > .05$, indicating that the effect of position was not significant. The interaction effect was not significant too: $F(2.5, 22.5) = 2$, $p > .05$.

Similarly, a two-way ANOVA was performed to study the influence of the two independent variables (position, feedback) on the time to match the target. After a Greenhouse-Geisser correction for insphericity, the main effect for feedback yielded $F(1, 9) = 0.43$, $p > .05$, indicating that the effect for feedback was not significant. The main effect for position yielded $F(1.4, 12.8) = 22.3$, $p < .001$, indicating a significant difference between position 1 ($M = 1396$, $SD = 150$), position 2 ($M = 1720$, $SD = 145$), position 3 ($M = 1799$, $SD = 133$), position 4 ($M = 2049$, $SD = 184$) and position 5 ($M = 2223$, $SD = 170$). The interaction effect was not significant: $F(1.5, 13.3) = 1.4$, $p > .05$.

5.3. Discussion

Our results are in agreement with the studies reported in [26]: visual-haptic feedback yields significantly higher performance than visual-only condition.

The target matching was generally very precise, since it was always guided by visual feedback: as a matter of fact, absolute errors were always smaller than 1 degree (that is 11.4 pixels). However, Figure 6 shows a significant performance increase in terms of mismatch reduction. This result is not surprising, as resistive force appeared each time the cursor was crossing a marker, hence allowing the participants to stop their action as soon as they felt an increase in torque. On the other hand, the position of the marker was not significant, meaning that the number of traversed resistive markers did not affect matching accuracy.

As shown in Fig. 7, the time needed to complete a trial depended on the position, with longer trials associated to farther targets. While this may appear obvious at first, one must consider

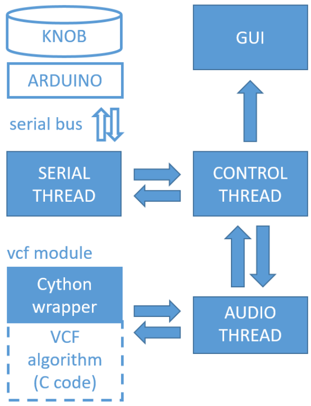

Figure 8: Python software architecture for interactive control of an audio effect with the haptic knob.

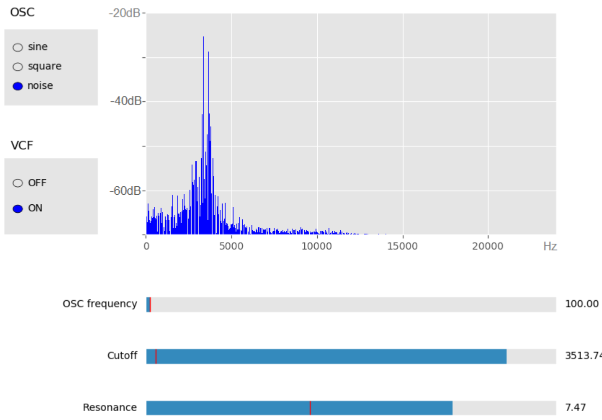

Figure 9: Audio effect GUI: Python-based visual interface displaying the available user controls and spectrum of the processed sound output.

that haptic feedback created a resistance to knob rotation, however this had no significant consequences on the time-to-match. What is more, the medians corresponding to factor combinations with haptic feedback were generally lower, suggesting that resistive force allowed the execution of slightly faster trials.

6. INTERACTIVE CONTROL OF A REAL-TIME DIGITAL AUDIO EFFECT

In the light of the experiment results, the haptic knob was configured as a multi-parametric input device to control different digital sound effect parameters. In particular, the device was mapped to a digital model of the EMS VCS3 voltage-controlled filter, previously presented in [27]. Given the computational complexity of the model, we updated the original Processing 3.5 implementation by developing a multi-threading software architecture in Python 3.6 and adopting the Python real-time audio programming approach described in [28].

Despite the fact that Python offers a fast learning curve, rapid prototyping tools, and that powerful libraries for scientific computing are available, it is still rarely used for real-time sound processing, mainly due to its dynamic and interpreted nature, therefore computationally inefficient. Our methodology [28] bypasses such inefficiency through a hybrid approach, in which external functions running native machine code are made available to the Python interpreter. This solution dramatically decreases the computation time of critical code sections (for instance the iterative computation of nonlinear functions at audio rate), often matching the real-time constraint on modern machines. The complete Python implementation of the EMS VCS3 VCF digital model, complemented by a computational benchmark, can be found in [28].

Figure 8 shows the implemented software architecture: the haptic device communicates with the digital effect through the Arduino microcontroller via serial connection. The related thread asynchronously receives commands and positional data, whereas the main control thread makes them available to the DSP algorithm, and in parallel updates the GUI shown in Fig. 9. The audio thread realizes a simple oscillator synthesizing a sine wave, a square wave, or white noise; in the meantime it processes sample-by-sample chunks of audio through the VCF algorithm. Such algorithm can be easily developed in Python and processed by the Numba library or can be developed in C code and, hence, made available to the Python interpreter through the so-called *cythonization* [28]. Two input modalities were made available through the knob:

- The **navigation mode**, allowing to select a control, is enabled by keeping the knob pressed while browsing over the controls through CW or CCW rotations;
- The **control mode** is enabled by simply rotating the knob CW or CCW once a control parameter has been selected, and allows to change its values.

With regard to the implemented haptic feedback, in navigation mode a hard detent (Multiple selector of Fig. 3) is generated each time the user jumps to a different control, while in control mode the following effects are produced: the selection of the sound source as well as the filter on/off switching are rendered by hard detents (visually displayed as radio buttons); the VCF cutoff frequency (visually represented by a slider) is rendered with a fixed low resistance; finally, the VCF resonance (visualized using a slider) is rendered with a variable resistance. The knob can be also used to select the fundamental frequency of the oscillator: in this case the knob actuates no feedback, just conveying its own physical resistance. All the controls visually represented by sliders furthermore provide a barrier effect corresponding to the sliders' endpoints

Figure 10 shows code excerpts of the Python threads forming the software architecture represented in Fig. 8. When a control is selected using the knob in navigation mode, the function `change_control(x)` is called, which populates the `selected_control` object attributes with the content of the argument `x`. The object `selected_control` embeds information such as the controlled parameter and its current value, the functions mapping the knob position into this value and, finally, information needed by the visual display and haptic effects. Thus, when the active instance of `selected_control` is changed, the GUI is updated and a consequent command is sent to the knob controller to configure the corresponding haptic effect. Next, after some processing by the `serial_read()` thread, the selected control is set with positional data coming from the knob.

```
import vcf  #cython compiled module
...
CHUNK_SIZE = 512 #buffer size
ampl = 0.7       #signal amplitude

#thread to read asynchronous messages
def serial_read(ser):
    while getattr(t, "serialActive", True):
        #read data from haptic knob
        s = ser.readline()
        ...
        selected_ctrl.val(encoder_val)

#define audio callback
def audio_callback(...):
    global y
    #generate samples
    y = generate_source(CHUNK_SIZE, ampl)
    #process samples
    if vcf_enabled:
        y = vcf.process_vcf(y)
    #update displayed FFT
    fft_data = display(scipy.fft(y))
    line_plot.set_ydata(fft_data)
    #convert to PCM format
    pcm = double_to_pcm(y)
    return (pcm, pyaudio.paContinue)

#change parameter to control
def change_control(x):
    selected_ctrl = x
    ...
    #send control data to haptic knob
    ser.write(selected_ctrl.haptic_effect)

#serial thread for haptic knob communication
ser = serial.Serial(port, baud, timeout=1)
t = threading.Thread(target=serial_read,
                                    args=(ser,))
t.start()

#audio thread
p = pyaudio.PyAudio()
s = p.open(..., stream_callback=audio_callback)
s.start_stream()

while True:
    wait_for_exit_signal()

# stop audio
s.stop_stream()
s.close()
p.terminate()
# stop serial thread
t.join()
```

Figure 10: Threads forming the Python software architecture.

The `audio_callback` function, called by the stream object `s` of the PyAudio library, generates and processes audio signal buffers according to user parameters setting the sound source and the effect properties. The function `process_vcf(y)` realizes the VCF and belongs to the library `vcf`, independently compiled and then imported as shown at the beginning of the code example. The simple GUI, generated using Matplotlib widgets, is regularly updated by the audio thread concerning the FFT window, and by the control thread concerning the visual elements forming the controls.

7. CONCLUSIONS

A low-cost knob controller was prototyped, offering programmable resistive-feedback and affording four different haptic effects. The effectiveness of the available haptic effects was assessed in a user study implementing a visually based target-matching task. A significant reduction of the error size was found when haptic effects were added. As a test application, the digital model of a voltage-controlled filter was interfaced with our haptic controller through a Python software architecture integrating real-time sound processing and interactive control.

The prototype demonstrates the concrete possibility to push the design of digital audio generation and processing beyond sound accuracy and quality, by including programmable hardware behaviors. In the analog domain, hardware controls contributed to the unique character of sound devices, and likely also to configurations that were finally preferred by musicians and engineers while crafting their sound. To this end, we are planning to test how knob rotation optimally scales with changes of the parameters, and how to fine-tune the haptic feedback for an optimal control of the same parameters.

As future work, we plan to improve the communication with the knob controller by implementing also MIDI/OSC protocols, and to specify a communication protocol encoding haptic information for the controller from the sound application: this way, an audio effect could respond to a parameter configuration with specific haptic feedback in addition to sound.

Regarding the software architecture, we plan to develop a structured framework hosting multiple digital effects, endowed of advanced GUIs realized using the Qt framework inside a Python environment.

8. ACKNOWLEDGMENTS

This research was partially supported by project HAPTEEV funded by the Swiss National Science Foundation (grant 178972). Yuri De Pra's Ph.D. is funded with a scholarship of The Research Hub by Electrolux Professional SpA.

9. REFERENCES

[1] Sebastian Merchel, Ercan Altinsoy, and Maik Stamm, "Tactile music instrument recognition for audio mixers," in *Proc. 128th AES Convention*, London, UK, May 22-25 2010, AES.

[2] Behringer, "BCR2000 MIDI controller," accessed March 25, 2021, Available at https://www.behringer.com/product.html?modelCode=P0245.

[3] Edgar Berdahl and Alexandros Kontogeorgakopoulos, "The firefader: Simple, open-source, and reconfigurable haptic force feedback for musicians," *Computer Music J.*, vol. 37, no. 1, pp. 23–34, 2013.

[4] Kjell Ivar Øvergård, Knut Inge Fostervold, Hans Vanhauwaert Bjelland, and Thomas Hoff, "Knobology in use: an experimental evaluation of ergonomics recommendations," *Ergonomics*, vol. 50, no. 5, pp. 694–705, 2007.

[5] Poh Kiat Ng, Yue Hang Tan, Kian Siong Jee, Li Wah Thong, Jian Ai Yeow, and Chiew Yean Ng, "Ergonomic knob design validation for improved musculoskeletal comfort," *Modern Applied Science*, vol. 10, no. 1, pp. 76, 2016.

[6] Kathrin Krieger, Alexandra Moringen, and Helge Ritter, "Number of fingers and grasping orientation influence human performance during haptic rotation," in *IEEE World Haptics Conf. (WHC)*. IEEE, 2019, pp. 79–84.

[7] Wanjoo Park, Laehyun Kim, Hyunchul Cho, and Sehyung Park, "Dial-based game interface with multi-modal feedback," in *Int. Conf. on Entertainment Computing*. Springer, 2010, pp. 389–396.

[8] Sylvain Pauchet, Catherine Letondal, Jean-Luc Vinot, Mickaël Causse, Mathieu Cousy, Valentin Becquet, and Guillaume Crouzet, "Gazeform: dynamic gaze-adaptive touch surface for eyes-free interaction in airliner cockpits," in *Proc. of the Designing Interactive Systems Conf. (DIS)*. ACM, 2018, pp. 1193–1205.

[9] Simon Voelker, Kjell Ivar Øvergård, Chat Wacharamanotham, and Jan Borchers, "Knobology revisited: A comparison of user performance between tangible and virtual rotary knobs," in *Proc. of the Int. Conf. on Interactive Tabletops & Surfaces (ISS)*. ACM, 2015, pp. 35–38.

[10] John B Filson, Eric B Daniels, Adam Mittleman, Sierra L Nelmes, and Yoky Matsuoka, "Dynamic distributed-sensor thermostat network for forecasting external events," Dec. 31 2013, U.S. Patent 8,620,841.

[11] Griffin Technology, "PowerMate multimedia control knob," accessed March 25, 2021, Available at https://www.iclarified.com/i1112/.

[12] Microsoft, "Surface Dial," accessed March 25, 2021, Available at https://www.microsoft.com/en-us/p/surface-dial/925r551sktgn.

[13] Charles Spence and Alberto Gallace, "Multisensory design: Reaching out to touch the consumer," *Psychology & Marketing*, vol. 28, no. 3, pp. 267–308, 2011.

[14] Laehyun Kim, Manchul Han, Sang Kyun Shin, and Se Hyung Park, "A haptic dial system for multimodal prototyping," in *Int. Conf. on Artificial Reality and Telexistence (ICAT)*, 2008.

[15] Sunghwan Shin, In Lee, Hojin Lee, Gabjong Han, Kyungpyo Hong, Sunghoon Yim, Jongwon Lee, YoungJin Park, Byeong Ki Kang, Dae Ho Ryoo, et al., "Haptic simulation of refrigerator door," in *IEEE Haptics Symposium (HAPTICS)*. IEEE, 2012, pp. 147–154.

[16] Nicolai Beni, Marco Grottoli, Francesco Ferrise, and Monica Bordegoni, "Rapid prototyping of low cost 1 dof haptic interfaces," in *IEEE Haptics Symposium (HAPTICS)*. IEEE, 2014, pp. 479–483.

[17] Tan Ping Hua, Yeong Che Fai, Ricky Yap, and Eileen Su Lee Ming, "Development of a low cost haptic knob," *Int. J. of Mechanical, Aerospace, Industrial, Mechatronic and Manufacturing Engineering*, vol. 5, no. 10, 2011.

[18] Dominique Chapuis, Xavier Michel, Roger Gassert, Chee-Meng Chew, Etienne Burdet, and Hannes Bleuler, "A haptic knob with a hybrid ultrasonic motor and powder clutch actuator," in *Joint EuroHaptics Conf. and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems (WHC)*. IEEE, 2007, pp. 200–205.

[19] Marie Aude Vitrani, Jason Nikitczuk, Guillaume Morel, and Costantinos Mavroidis, "Torque control of electrorheological fluidic resistive actuators for haptic vehicular instrument controls," in *Proc. of the IEEE Int. Conf. on Robotics & Automation*, 2006.

[20] Stefan Battlogg, "Magnetorheological transmission device," July 28 2015, U.S. Patent 9,091,309.

[21] Native Instruments, "Traktor Kontrol S4," accessed March 25, 2021, Available at https://www.native-instruments.com /en/products/traktor/dj-controllers/traktor-kontrol-s4/.

[22] Mathias Kirkegaard, C Frisson, and MM Wanderley, "Torquetuner: A self contained module for designing rotary haptic force feedback for digital musical instruments," in *Proc. of the 2020 Int. Conf. on New Interfaces for Musical Expression, NIME*, 2020.

[23] Yuri De Pra, Federico Fontana, Stefano Papetti, and Michele Simonato, "A low-cost endless knob controller with programmable resistive force feedback for multimedia production," in *Proc. of the 17th Sound and Music Computing Conf. (SMC 2020)*, 2020.

[24] Yuri De Pra and Federico Fontana, "Haptic controller with programmable resistive force," 2021, IPO 102021000009068, Status: Pending.

[25] Ralph B. D'Agostino, "An omnibus test of normality for moderate and large size samples," *Biometrika*, vol. 58, no. 2, pp. 341–348, 1971.

[26] Matthew S Prewett, Liuquin Yang, Frederick RB Stilson, Ashley A Gray, Michael D Coovert, Jennifer Burke, Elizabeth Redden, and Linda R Elliot, "The benefits of multimodal information: a meta-analysis comparing visual and visual-tactile feedback," in *Proc. of the 8th Int. Conf. on Multimodal interfaces*, 2006, pp. 333–338.

[27] Federico Fontana and Marco Civolani, "Modeling of the ems vcs3 voltage-controlled filter as a nonlinear filter network," *IEEE Trans. on Audio, Speech, and Language Processing*, vol. 18, no. 4, pp. 760–772, May 2010.

[28] Yuri De Pra and Federico Fontana, "Programming real-time sound in python," *Applied Sciences*, vol. 10, no. 12, pp. 4214, 2020.

[29] Yuri De Pra, Federico Fontana, and Michele Simonato, "Development of real-time audio applications using python," in *Machine sounds, Sound machines - XXII Colloquio di Informatica Musicale*, Udine, Italy, 2018.