TorqueTuner: A self contained module for designing rotary haptic force feedback for digital musical instruments

Mathias Kirkegaard, Mathias Bredholt, Christian Frisson, Marcelo M. Wanderley
Input Devices and Musical Interaction Lab - IDMIL
Centre for Interdisciplinary Research in Music Media and Technology - CIRMMT
McGill University, Canada
{mathias.kirkegaard;mathias.bredholt}@mail.mcgill.ca
{christian.frisson;marcelo.wanderley}@mcgill.ca

Figure 1: TorqueTuner as a rotary force feedback joint between two T-Sticks

ABSTRACT

TorqueTuner is an embedded module that allows Digital Musical Instrument (DMI) designers to map sensors to parameters of haptic effects and dynamically modify rotary force feedback in real-time. We embedded inside TorqueTuner a collection of haptic effects (Wall, Magnet, Detents, Spring, Friction, Spin, Free) and a bi-directional interface through libmapper, a software library for making connections between data signals on a shared network. To increase affordability and portability of force-feedback implementations in DMI design, we designed our platform to be wireless, self-contained and built from commercially available components. To provide examples of modularity and portability, we integrated TorqueTuner into a standalone haptic knob and into an existing DMI, the T-Stick. We implemented 3 musical applications (Pitch wheel, Turntable and Exciter), by mapping sensors to sound synthesis in audio programming environment SuperCollider. While the original goal was to simulate the haptic feedback associated with turning a knob, we found that the platform allows for further expanding interaction possibilities in application scenarios where rotary control is familiar.

Author Keywords

digital musical instruments, haptic feedback, force-feedback interaction, sound synthesis, mapping, embedded systems

CCS Concepts

•**Applied computing** → **Sound and music computing;** Performing arts; •**Human-centered computing** → **Haptic devices;**

NIME'20, July 21-25, 2020, Royal Birmingham Conservatoire, Birmingham City University, Birmingham, United Kingdom.

1. INTRODUCTION

Haptic feedback is considered an important dimension of the interaction with musical instruments, as it can facilitate a performer-instrument intimacy and lead to a feeling of embodiment [28]. While this is an inherent mechanical feature of most acoustic instruments, many researchers have explored ways of synthesizing the haptic feedback, for digital musical instruments (DMI) where there is no mechanical coupling between gesture and sound [14, 17, 19, 12]. In this paper, we review related work on rotary and self-contained force-feedback (Section 2). We contribute the implementation of an open hardware module for rotary haptics with embedded effects library and bi-directional mapping interface (Section 3). We validate our module with 2 integrations in DMIs (Section 4) and 3 musical applications (Section 5).

2. RELATED WORK

We review related work on force feedback interaction, with a focus on rotary and self-contained haptics.

2.1 Force feedback

Haptic feedback is usually divided into two main categories, *force feedback* and *tactile feedback*. Force feedback relies on the kinesthetic sense (position, movement and force on body limbs), whereas tactile feedback is sensed through mechanoreceptors in the skin. For acoustic instruments the haptic feedback is usually inherited directly from the excitation mechanism, and this has been imitated with synthesized force feedback in the design of virtual instrument controllers such as a force feedback keyboard [14] and a virtual bowed string [19]. Tactile feedback has been used in musical applications for displaying textures [17], tempo [22] and vibrations of virtual resonant bodies [12]. Tactile feedback can be implemented with vibration motors small enough to fit in a smartphone, but larger and more powerful actuators are usually needed for force feedback in order to obtain a sufficient *z-width*, i.e. the range of displayable forces [27]. This increases both mechanical and electrical complexity, and is likely one explanation of why tactile feedback has gained the most popularity, in addition to the cost of force feedback devices remaining high. Examples of research that seek to accommodate this issue include Verplank's force feedback device The Plank [26] constructed from an old hard drive, Berdahl and Kontogeorgakopoulos's affordable and open-source device, Firefader [2] based on linear actuators, and open-source force feedback developments kits Hapkit [16] and by Haply [8]. A very common user input in electronic music is rotary control, usually in the form of knobs [4], but none of these devices are designed specifically for 1-DOF rotary control. Our main goal with TorqueTuner is to use affordable components and open-source technologies, to provide powerful rotary force feedback.

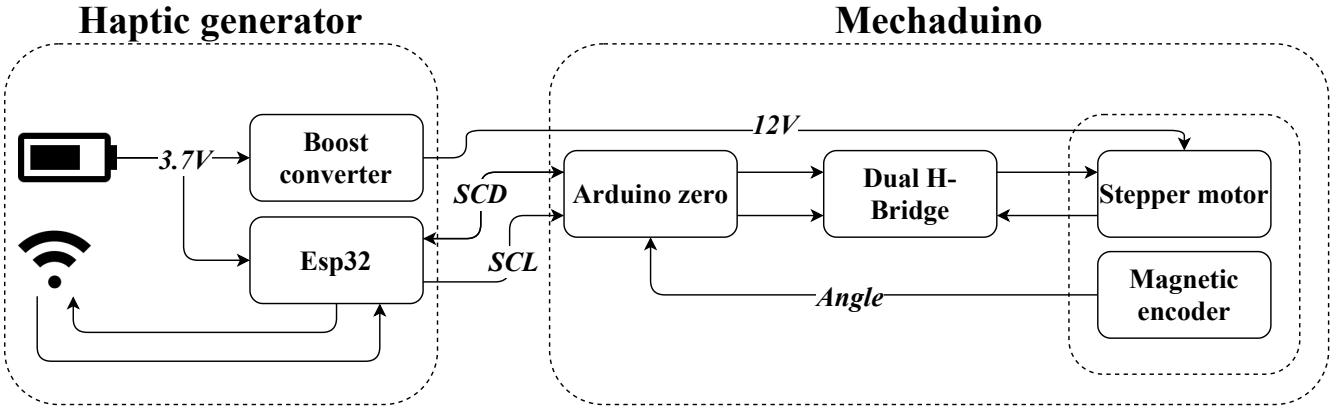


Figure 2: Block diagram of TorqueTuner hardware platform

2.2 Rotary force feedback

Swindells explored the mechanical and emotional aspects of interaction with physical rotary controls to inform the design of a force feedback knob that simulates detents [21]. Chu implemented a collection of haptic effects for such a device, and explores its application to audio navigation [6]. Their main finding is how simulated springs can be used to lock a user input into a center position. Oosterhout et al. explored the perceptual links between the shape and the haptic feedback of a knob [25], and expanded their design into a shape changing knob [24]. Beamish et al. used rotary force feedback in the design of a DMI, a turntable for stable velocity playback and a collection of additional haptic effects including haptic beat markers and scratching with a spring [1]. Our goal is to use their guidelines in our design of rotary haptic effects, and extend it by making them available for DMI designers and artist in a low-cost and open-source in module.

2.3 Self contained haptics

The control of haptic feedback is usually implemented by a haptic generator that maps information from sensors and sound synthesis to actuators [9]. In many existing applications, the haptic generator is a piece of software running on a desktop/laptop, that bridges a musical application and a haptic device. Berdahl and Smith's Synth-A-Modeller [3] allows artists to design bi-directional (control in, feedback out) synthesis applications based on virtual mechanical objects, and compile it to Faust DSP code, which can be controlled with haptic devices. A similar approach is taken in Sinclair's work on Dimple [20], an interactive software system for developing virtual haptic environments with an interface through the Open Sound Control (OSC) protocol. All though these solutions allow for rendering of complex haptic environments, we find that relying on a desktop/laptop introduces a limitation on longevity and portability since compatibility maintenance is required. A main goal of the design of TorqueTuner is to embed the haptic rendering on a micro controller, in order to implement a self contained force feedback module that DMI designers can easily integrate into new instruments.

3. IMPLEMENTATION

The implementation of TorqueTuner is threefold: a hardware platform, a collection of haptic effects, and a wireless mapping interface. In this section we describe aspects of the implementation important for the goal of a self-contained rotary force feedback module, and present 2 exemplar integrations of TorqueTuner.

3.1 Hardware design

The hardware platform consists of a servo implemented with the *Mechaduino* [23] platform, a haptic generator embedded on a ESP32 microcontroller unit (MCU) and a LiPo-based power-supply (see Figure 2). The Mechaduino handles timing sensitive tasks on an interrupt basis, including the calculation of velocity, acceleration and the motor control loop, while the ESP32 handles haptic calculations and WiFi connectivity. We implement a bi-directional full speed I^2C interface for the internal communication. Additionally, we implement a checksum algorithm to enable discarding of erroneous data.

3.1.1 Rotary haptics with Mechaduino servomotor

Mechaduino is an open-source Arduino compatible platform that implements an industrial servo with a low cost stepper motor and a hall effect based magnetic encoder. It is designed for the standardized National Electrical Manufacturers Association (NEMA) 17 stepper motor frame-size, which is commonly used in 3D printers and CNC machines, and thus commercially available at a low price. The motor we used for TorqueTuner has a holding torque of 45 Ncm and a rated current draw of 2A. The Mechaduino platform utilizes the stepper motor with known step angle in an open loop configuration, to calibrate the magnetic encoder and bring the angular resolution down to $\pm 0.1^\circ$ corresponding to 3600 PPR.

3.1.2 Embedded control with ESP32

The ESP32 is a system on a chip (SOC) microcontroller with WiFi and Bluetooth. The ESP32 integrates a 240 MHz dual core microprocessor with a floating point unit. The TinyPico breakout of ESP32 has 4 MB of PSRAM memory, which makes it capable of storing all the lookup tables containing the haptic effects.

3.1.3 Portable power supply

We use a one cell LiPo battery to power the ESP32 and a boost converter to step up the voltage from 3.7 V to 12 V for driving the stepper motor. LiPo batteries have become very common in portable electronics due to their small size and very high discharge rate. Drawbacks include the necessity of protecting against over discharging and stabilizing the voltage output as it drops during discharge. The TinyPico features a LiPo battery charging circuit that implements under-voltage protection for the battery and constant voltage output. We have included a block diagram of the final hardware setup in Figure 2.

3.2 Embedded haptic effects

We implement the collection of haptic effects as transfer functions that relate angle or velocity to torque, and store them as look-up tables on the ESP32. In this way, we can generalize the control inputs for every haptic effect as `scale` (output gain), `stretch` (input gain) and `offset` (input offset) of the associated transfer functions. Figure 3 shows the transfer functions for 4 of the haptic effects. Additionally we introduce the control input `damping` which define the energy dissipation of the system. This can be both negative and positive, allowing for designing *passive displays*, where reactive forces (the ones displayed) equal the applied forces, and *active displays* where the displayed forces might be larger. Positive values corresponds to a damped system, and negative values corresponds to adding energy. This can be used to prevent or introduce oscillations respectively. An overview of the control inputs is given in Table 1.

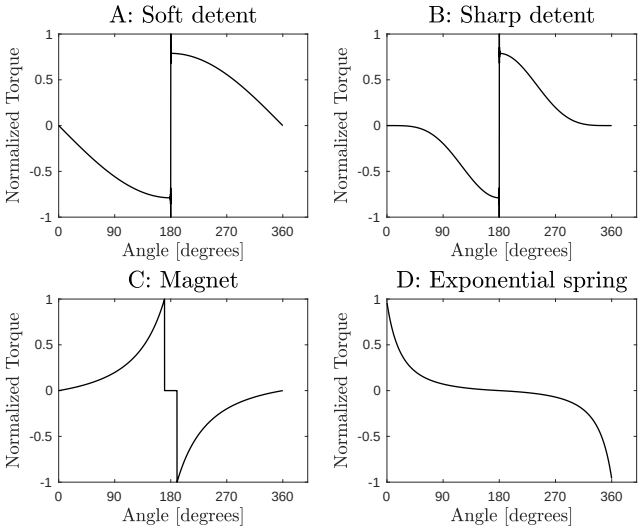


Figure 3: Transfer functions for angle dependent haptic effects. The sign of the torque represents the direction, such that a negative torque is towards a decrease in angle.

A haptic update procedure running a 1 kHz calculates the controller set points, torque and velocity, based on sensor data from the servo, control inputs from libmapper and the chosen haptic effect. The overall data flow of the haptic update procedure is shown in Figure 4, and the implementation of haptic effects is explained below.

3.2.1 Wall

Colgate et al. defined a virtual wall with stiffness and damping [7] as:

$$\begin{aligned}&\textbf{if } (\theta \geq \theta_w) \ \{\\ &\quad F_t = K(\theta - \theta_w) - B \cdot \dot{\theta}\\ &\}\\ &\textbf{else } \{\\ &\quad F_t = 0\\ &\}\end{aligned}$$

F_t : Torque
θ : Encoder angle
$\dot{\theta}$: Angular velocity.
θ_w : Angular position of the wall
K : Angle-dependent wall stiffness.
B : Damping factor.

The virtual wall is used as haptic feedback on the range of the knob by assigning θ_{wall} to the desired minimum and maximum angle, the default values are $\theta_{min} = 0°$ and $\theta_{max} = 360°$.

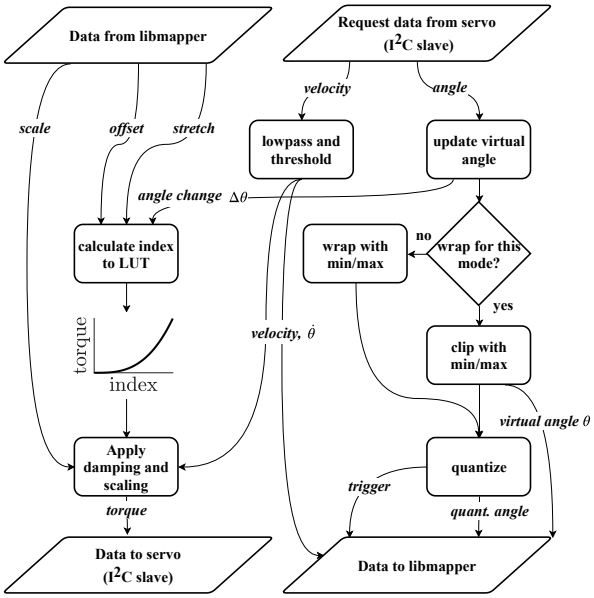


Figure 4: Diagram showing the data flow and -processing of the main update function of the Haptic Generator

3.2.2 Magnet

An exponential increase in torque, as the distance to to a virtual magnet linearly decreases. A negative `Scale` input could be applied to change the polarity of the magnet. The position of the magnet is by default in the middle, 180°, as seen in Figure 3, but could be changed with `offset`. The flat (zeroed) area around the magnet position, is implemented to assure stability.

3.2.3 Detents

Exponentially increasing torque opposing the turning direction, followed by a transient change in torque direction at the position of the detent, effectively resulting in a haptic click. 2 examples are shown in Figure 3 (A and B).

3.2.4 Friction

A linear relation between torque and angular velocity.

$$F_t = \dot{\theta} * \mu \tag{1}$$

where μ is a friction coefficient. In this mode, the control input `scale` maps to μ. The velocity is calculated from position through backwards differentiation, and low-pass filtered in order to remove noise.

3.2.5 Spin

A PID controller was hand-tuned while carrying the load of a custom designed knob, and used to maintain a constant target velocity. In this mode, the `Scale` input sets the controller effort, which corresponds to the maximum torque applied to reach target velocity. This is an example of active feedback.

3.2.6 Free

An unmapped mode in which the `Scale` parameter controls the torque directly. This allows for freely defining the haptic behavior through mappings in libmapper, either by mapping internal parameters (map to itself), to external controllers (sensors of other DMI's), to sound related parameters or to any combination of these. This mode could also be used for user defined curves.

3.3 Bi-directional mapping interface

We generate a set of output control signals based on the angle estimate obtained from the magnetic encoder. The set includes first and second order derivatives (velocity and acceleration), a virtual angle which can be clipped or wrapped to a desired range, a quantized angle and a trigger signal. The two last mentioned are both synchronized to the position of detents by default.

Input signal	Description
`TargetVelocity`	Set-point for velocity controller.
`Scale`	Gain of transfer function output.
`Stretch`	Gain of transfer function input.
`Offset`	Offset of transfer function input.
`Mode`	Selects haptic effect.
`Damping`	Ratio of energy dissipation.
Output signal	**Description**
`Angle`	Set-point for velocity controller.
`QuantAngle`	Quantization steps = number of detents.
`Velocity`	Gain of transfer function input.
`Acceleration`	Offset of transfer function input.
`Trigger`	A short pulse at every quantization step.

Table 1: Table of inputs and outputs exposed in libmapper, and used for interaction with music software

3.3.1 Mapping library for embedded platforms

libmapper is a library for making connections between data signals on a shared network using OSC [11]. The library is written in C and is supported on macOS, Windows, and Linux. In order to compile libmapper for ESP32, its dependencies need to be compiled first. libmapper has two dependencies, *liblo* and *zlib*. liblo is a lightweight library for OSC communication. zlib is a library for data compression. The zlib library has no dependencies, and we could compile it with no modifications. liblo relies on POSIX sockets and POSIX threads (pthreads) for creating TCP/UDP sockets and servers. The ESP32 platform provides the ESP-IDF (Espressif IoT Development Framework), which is an API for developing embedded applications. The ESP-IDF includes the library *lwip*, a lightweight TCP/IP stack. The API of lwip is made to replicate the behaviour of POSIX sockets, such that the library can serve as a compatibility layer for using POSIX sockets on embedded systems. As the ESP-IDF is based on the FreeRTOS operating system, the ESP-IDF supports multitasking with threads (called *tasks* in FreeRTOS). Conveniently, ESP-IDF includes a pthread library that translates the FreeRTOS API into the POSIX threads API, allowing software that uses pthreads to run on the ESP32. These abstractions provide most of the implementations needed for the port of libmapper. We contributed two minor but required changes to liblo. First, in the function which translate incoming data packets into OSC messages, we unified the memory allocation implementations between desktop (Windows, macOS, and Linux) and embedded (ESP32) platforms that return different pointers values. Second, libmapper relies on loop-back of multicast messages i.e. sending all outgoing messages to itself, for assigning a correct device ID. We specifically enabled the socket option `IP_MULTICAST_LOOP` for ESP32, which is enabled by default on macOS and Linux.

To facilitate the use of libmapper for embedded platforms, we chose to port libmapper from an ESP-IDF project into an Arduino library. Arduino libraries are distributed as source and header files along with a text file containing metadata for the library. This means that the library is compiled together with the user's project, every time the user presses the "Compile" button in the Arduino IDE. As there are quite a few compiler variables that need to be set up correctly for libmapper to compile, using this approach would require a lot of changes to the libmapper source code. According to the *Arduino IDE 1.5: Library specification*, a library can alternatively be distributed as precompiled. This has the benefit that a Makefile can be written to generate the Arduino library as a collection of header files and a precompiled archive (.a) file. Another benefit is that it will speed up the compilation time for the user, as the library would already be compiled. For these reasons, this was the chosen approach for distributing the library. The libmapper Arduino library and all TorqueTuner related source code are available at `https://github.com/IDMIL/TorqueTuner`.

4. MODULAR INTEGRATIONS IN DMIS

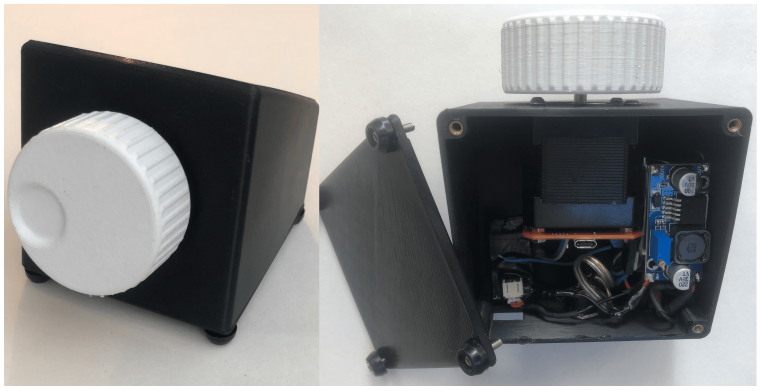
We produced two exemplar integrations of our TorqueTuner module, (1) a standalone knob and (2) an integration with an existing DMI, the T-stick. These are presented in Figure 5.

4.1 Standalone haptic knob

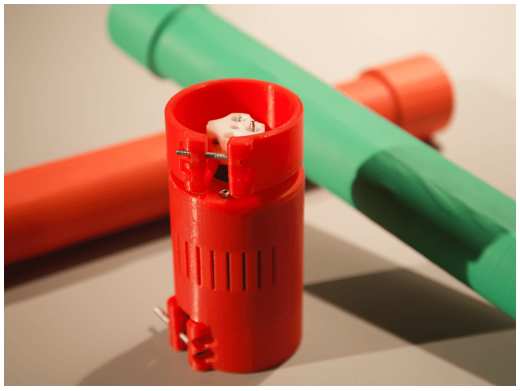
To create a standalone knob module, we 3d printed a knob and a mounting enclosure with a tilted front-panel. We used it for the design of the haptic effects, by making internal mappings in the `free` mode. We propose this as a platform for DMI designers to prototype rotary force feedback for new instruments. This is our rotary equivalent of the "Hello Wall" starter example of haptics tutorials in planar spaces [8]. We believe that our open-source and cost-effective integration will allow hapticians to easily replicate results from related research works on force-feedback rotary haptics [1, 21, 24, 25].

4.2 Rotary joint between 2 T-Sticks

The T-Stick is a DMI developed at Input Devices and Music Interaction Laboratory (IDMIL) 10 years ago, that has been continuously extended and maintained ever since. It is a stick build from PVC tube, and it includes pressure sensing, capacitive sensing, 6 axis position and movement sensing and triggering with a piezo mic. It uses a ESP32 internally for sensor acquisition and transmission of sensor data over a network, and is thereby libmapper compatible [13]. There is some inherent force feedback when playing the IMU of the T-stick with impulsive gestures such as shaking or swinging, as its mass poses a resistance against acceleration, but for slow changes in the orientation of the instrument, the performer relies mainly on visual feedback. As an example of portability of TorqueTuner, we designed an adapter for adding rotary force feedback, and a new twisting gesture, to the T-Stick. Figure 5b shows the 3D printed T-Stick adapter, and 2 T-sticks. I works like a force feedback joint between 2 T-sticks, one mounted on each end. See Figure 1. We find the `Spring` mode particularly useful in conveying feedback on the orientation of the instrument, specifically for *roll*.



(a) TorqueTuner as a standalone haptic knob



(b) TorqueTuner as T-stick adapter

Figure 5: Two examples of modular integrations of TorqueTuner in DMIs

5. MUSICAL APPLICATIONS

We present 3 example applications mapping TorqueTuner haptics to sound synthesis. We developed an extension *MapperUGen* adding libmapper support to audio programming environment SuperCollider.

5.1 Turntable

We designed a turntable-like application, where playback can be stopped by stalling the motor, and scratching can be performed by twisting it back and forth. A DC signal is mapped to input `Target Velocity`, in `Spin` mode, such that the motor rotates at constant speed. The output `Velocity`, is mapped to the playback speed of drum loop. The closed loop velocity controller tries to maintain the target velocity at all times, and therefore a torque is displayed when the user slows or stalls the motor. In [10] an energy continuum is discussed, i.e. the importance of a direct link between sound intensity and user energy input. In this mapping, an energy continuum is preserved, but the link is reversed, as energy is required from the user to silence the sound.

5.2 Pitch wheel

We designed an application reminiscent of the pitch wheel common in many keyboard synthesizers. When twisting, TorqueTuner displays an exponentially increasing force, when released it returns to idle. When using TorqueTuner in the Spring mode, there is a natural idle state, where the virtual spring is in equilibrium (neither compressed or stretched), corresponding to 180° on Figure 3D. In this mode, a saw wave is passed through a resonant low-pass filter, and the cutoff frequency is mapped to the angle. The exponential torque/angle relation serves as continuous feedback on the parameter state, and allows for accurate tuning of the cutoff frequency. This is reminiscent of the theremin + elastic band experiment in[15]. A second controller, acceleration data from the T-Stick, is mapped to `Scale`, allowing a shaking gesture on the T-Stick to be reflected as oscillations in torque. This causes small changes in the angle, and results in accurate control of vibrato-like modulation, which would be difficult with the T-Stick alone.

5.3 String plucker

We designed the string plucker application where detents serve as haptic feedback on virtual string plucking, and where twisting in-between detents produce vibrato-like modulation. In `Sharp Detent` mode, the `Trigger` output is mapped to the excitation input of a Karplus-Strong based plucked string model. The `Angle` output is mapped to the length of a delay-line, effectively changing the pitch of the plucked string. This mapping was explored when mounted to the T-stick, and the output of the pressure sensor was mapped to `Stretch`, such that detent density could be controlled by squeezing the T-Stick. A demonstration video of the 3 applications is available at: `https://vimeo.com/404592134`

6. DISCUSSION

An interview-based analysis on the challenges of haptic design is given in [18], and one finding is that hapticians find the implementation of demos/prototypes costly and complex but crucial in haptic experience design (HaXD). We believe that TorqueTuner, being self contained and cost effective, has potential of accommodating this issue. We also believe that having a wireless interface to common music software can assist also DMI's designers and artists in doing HaXD which is otherwise the work of hapticians, and therefore see a great potential in integrating libmapper into more music software. The appeal of TorqueTuner as a prototyping device for artists and non-programmers could be further enhanced with the implementation of a GUI for free hand curve editing, allowing for designing haptic effects, storing them as lookup tables and previewing them in real-time. We are currently working on implementing the standalone haptic knob, as a rotary force feedback block for Probatio, a modular toolkit for rapid prototyping postures and sensor combinations for DMI's. [5] The LiPo battery solution contributes to the modular and portable quality of TorqueTuner, and enables free endless rotation of both the motor-shaft and the body of TorqueTuner, as would not be possible with a wired power solution. While this eased the integration with the T-stick, and hopefully other future DMI's, it poses great limitations in terms of capacity and life-time. In power hungry applications such as the turntable, a fresh battery was discharged in 15 minutes. We suggest that future work focus on 1) optimizing for power losses in TorqueTuner and 2) limit the scope of haptic effects with power consumption in mind. We suggest a possible solution to 1) be the introduction a passivity observer, like the capacitive touch sensing in [2], that allows for disabling the actuator when there is no user interaction.

7. CONCLUSION

In this paper we described TorqueTuner, a self contained rotary force feedback module with an embedded collection of haptic effects. With the motivation of assisting DMI designers in prototyping force-feedback instruments, we built TorqueTuner from affordable components and open-source technologies, and designed a generalized and wireless control interface exposed via libmapper. We implemented a LiPo battery solution to make the module completely wireless, and thereby easy to integrate into existing DMI's. We produced 2 exemplar integrations that demonstrate the portability of the module, and made 3 musical applications by mapping the TorqueTuner to sound synthesis in SuperCollider. Wireless power and communication posed issues of capacity and life-time that limits the scope of displayable haptic effects. Optimizations on power loss should be explored in the future.

8. ACKNOWLEDGEMENTS

Christian Frisson is the recipient of Elevate grants IT12555 and IT12556 co-funded by Mitacs and Haply Robotics.

9. REFERENCES

[1] T. Beamish, K. MacLean, and S. Fels. Designing the haptic turntable for musical control. In *Haptic Interfaces for Virtual Environment and Teleoperator Systems*, page 24. IEEE Computer Society, 2003.

[2] E. Berdahl and A. Kontogeorgakopoulos. The firefader: Simple, open-source, and reconfigurable haptic force feedback for musicians. *Computer Music Journal*, 37(1):23–34, 2013.

[3] E. Berdahl and J. O. Smith. An introduction to the synth-a-modeller compiler: Modular and open-source sound synthesis using physical models. In *Linux Audio Conference*, 2012.

[4] K. Bjørn, J. Jarre, M. Metlay, and P. Nagel. *Push Turn Move: Interface Design in Electronic Music*. 2016.

[5] F. Calegario, M. Wanderley, S. Huot, G. Cabral, and G. Ramalho. A method and toolkit for digital musical instruments: Generating ideas and prototypes. *IEEE Multimedia*, 24(1):63–71, 2017.

[6] L. L. Chu. Using haptics for digital audio navigation. In *International Computer Music Conference (ICMC)*, 2002.

[7] J. E. Colgate, P. E. Grafing, M. C. Stanley, and G. Schenkel. Implementation of stiff virtual walls in force-reflecting interfaces. In *Virtual Reality Annual International Symposium*, page 202–208. IEEE, 1993.

[8] S. Ding and C. Gallacher. The haply development platform: A modular and open-sourced entry level haptic toolset. In *Extended Abstracts of the 2018 CHI Conference on Human Factors in Computing Systems*, 2018.

[9] M. Giordano, J. Sullivan, and M. Wanderley. *Design of Vibrotactile Feedback and Stimulation for Music Performance*, pages 193–214. Springer, 2018.

[10] A. Hunt, M. Wanderley, and M. Paradis. The importance of parameter mapping in electronic instrument design. *Journal of New Music Research*, 32(4):429–440, 2003.

[11] J. Malloch, S. Sinclair, and M. Wanderley. Libmapper: (a library for connecting things). In *CHI '13 Extended Abstracts on Human Factors in Computing Systems*, page 3087–3090, 2013.

[12] M. Marshall and M. Wanderley. Examining the effects of embedded vibrotactile feedback on the feel of a digital musical instrument. In *New Interfaces for Musical Expression (NIME)*, 2011.

[13] A. Nieva, J. Wang, J. Malloch, and M. Wanderley. The t-stick: Maintaining a 12 year-old digital musical instrument. In *New Interfaces for Musical Expression (NIME)*, pages 198–199, 2018.

[14] R. Oboe. A multi-instrument, force-feedback keyboard. *Computer Music Journal*, 30:38–52, 2006.

[15] S. O'Modhrain. *Playing by Feel: Incorporating Haptic Feedback into Computer-Based musical Instruments*. PhD thesis, Stanford University, 2000.

[16] M. Orta Martinez, C. M. Nunez, T. Liao, T. K. Morimoto, and A. Okamura. Evolution and analysis of hapkit: An open-source haptic device for educational applications. *IEEE Transactions on Haptics*, 2019.

[17] J. Rovan and V. Hayward. *Typology of tactile sounds and their synthesis in gesture-driven computer music performance*, pages 297–320. IRCAM – Centre Pompidou, 2000.

[18] O. S. Schneider, K. E. MacLean, C. Swindells, and K. S. Booth. Haptic experience design: What hapticians do and where they need help. *International Journal of Human-Computer Studies*, 107, 2017.

[19] S. Sinclair, G. P. Scavone, and M. M. Wanderley. Audio-haptic interaction with the digital waveguide bowed string. In *International Computer Music Conference (ICMC)*, 2009.

[20] S. Sinclair and M. Wanderley. A run-time programmable simulator to enable multi-modal interaction with rigid-body systems. *Interacting with Computers*, 21(1-2):54–63, 2009.

[21] C. E. Swindells. *Incorporating affect into the design of 1-D Rotary Physical Controls*. PhD thesis, University of British Columbia, 2007.

[22] Peterson Tuners. Peterson bodybeat sync. `https://www.petersontuners.com/products/bodybeatsync/`.

[23] Tropical Labs. Mechaduino. `https://tropical-labs.com/mechaduino/`.

[24] A. van Oosterhout, E. Hoggan, M. K. Rasmussen, and M. Bruns. Dynaknob: Combining haptic force feedback and shape change. In *Proceedings of the 2019 on Designing Interactive Systems Conference*, page 963–974, 2019.

[25] A. van Oosterhout, M. K. Rasmussen, E. Hoggan, and M. Bruns. Knobology 2.0: Giving shape to the haptic force feedback of interactive knobs. In *The 31st Annual ACM Symposium on User Interface Software and Technology Adjunct Proceedings*, page 197–199, 2018.

[26] B. Verplank, M. Gurevich, and M. Mathews. The plank: Designing a simple haptic controller. In *New Interfaces for Musical Expression (NIME)*, 2002.

[27] D. Weir and J. Colgate. Stability of haptic displays. *Haptic Rendering: Foundations, Algorithms, and Applications*, pages 123–156, 2008.

[28] G. Young, D. Murphy, and J. Weeter. *A Functional Analysis of Haptic Feedback in Digital Musical Instrument Interactions*, pages 95–122. Springer, 2018.