VibraForge: A Scalable Prototyping Toolkit For Creating Spatialized Vibrotactile Feedback Systems

Bingjian Huang
bingjian20@dgp.toronto.edu
Dynamic Graphics Project Lab,
University of Toronto
Toronto, Ontario, Canada

Siyi Ren
siyi.ren@mail.utoronto.ca
Division of Engineering Science,
University of Toronto
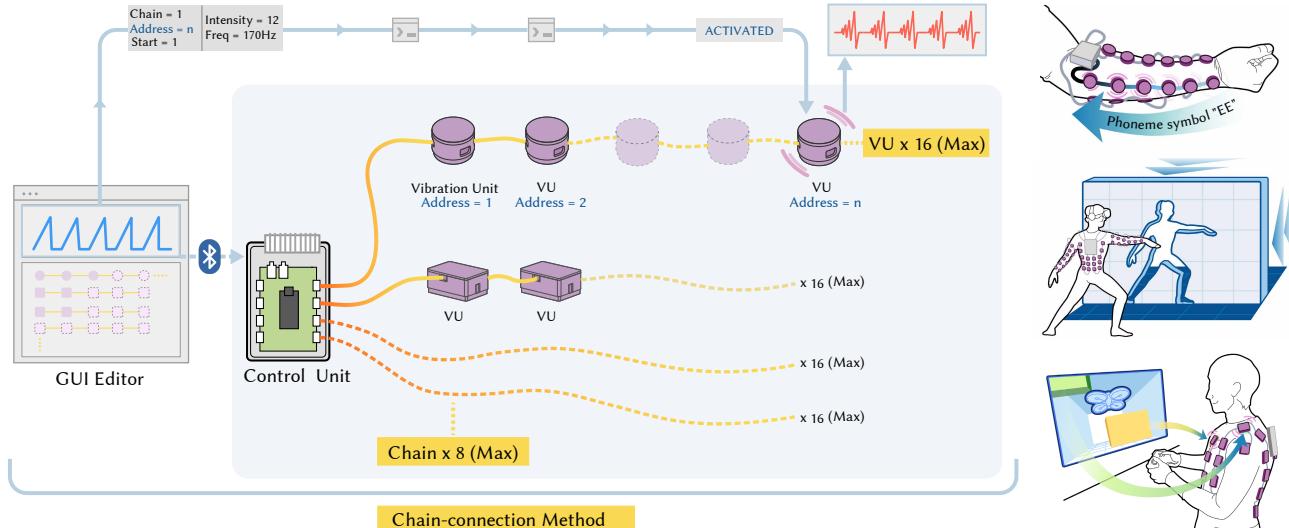Toronto, Ontario, Canada

Yuewen Luo
sofia.luo@mail.utoronto.ca
Division of Engineering Science,
University of Toronto
Toronto, Ontario, Canada

Qilong Cheng
qilong.cheng@mail.utoronto.ca
Mechanical Engineering, University
of Toronto
Toronto, Ontario, Canada

Hanfeng Cai
hanfeng.cai@mail.utoronto.ca
Department of Electrical and
Computer Engineering, University of
Toronto
Toronto, Ontario, Canada

Yeqi Sang
abel.sang@mail.utoronto.ca
Department of Mechanical &
Industrial Engineering, University of
Toronto
Toronto, Ontario, Canada

Mauricio Sousa
mauricio@dgp.toronto.edu
Dynamic Graphics Project Lab,
University of Toronto
Toronto, Ontario, Canada

Paul H. Dietz
dietz@cs.toronto.edu
Dynamic Graphics Project Lab,
University of Toronto
Toronto, Ontario, Canada

Daniel Wigdor
daniel@dgp.toronto.edu
Dynamic Graphics Project Lab,
University of Toronto
Toronto, Ontario, Canada

Figure 1: VibraForge is a scalable, open-source vibrotactile toolkit that supports the creation of spatialized vibrotactile feedback systems. Leveraging a chain-connection method, one control unit in the system can independently control up to 128 vibration units. Applications of the toolkit are shown on the right.

Abstract

Spatialized vibrotactile feedback systems deliver tactile information by placing multiple vibrotactile actuators on the body. As increasing numbers of actuators are required to adequately convey information in complicated applications, haptic designers find it difficult to

create such systems due to limited scalability of existing toolkits. We propose VibraForge, an open-source vibrotactile toolkit that supports up to 128 vibrotactile actuators. Each actuator is encapsulated within a self-contained vibration unit and driven by its own microcontroller. By leveraging a chain-connection method, each unit receives independent vibration commands from a control unit, with fine-grained control over intensity and frequency. We also designed a GUI Editor to expedite the authoring of spatial vibrotactile patterns. Technical evaluations show that vibration units reliably reproduce audio waveforms with low-latency and high-bandwidth data communication. Case studies of phonemic tactile display, virtual reality fitness training, and drone teleoperation demonstrate the potential usage of VibraForge within different domains.

CCS Concepts

• **Human-centered computing** → **Haptic devices**; **User interface toolkits**.

Keywords

Haptics, Vibrotactile, Toolkits, Wearable Devices

1 Introduction

Tactile feedback conveys sensory information to the human body through the sense of touch [8]. Vibrotactile feedback, which produces tactile sensations via actuators vibrating on the skin, has gained much popularity because it is compact, affordable, and accessible [6]. It has been widely adopted in various domains, such as virtual reality (VR) [4, 17, 52], robotics [1, 31, 57], rehabilitation [3, 23, 35], and accessibility [16, 37, 55].

To provide adequate feedback to users during complex tasks, spatialized vibrotactile feedback systems have been proposed [22, 24]. These systems utilize actuator positions as an additional channel to enhance information transmission. For example, Wang et al. designed a facial mask with 36 eccentric rotating mass (ERM) actuators to study the perception of localization and orientation of objects in VR [52]. The bHaptics TactSuit X40 [4] utilized 40 ERMs on a vest to create localized feedback for immersive VR gaming experiences. The bHaptics TactSuit was also used by Xia et al. to convey hydrodynamic flow intensity during submarine teleoperation [57]. Jung et al. designed a wireless haptic interface with 36 ERMs to convey detailed fingertip touch sensations on the upper arm for amputees [16]. Sanchez et al. created OpenVNAVI, a navigation aid system with 128 ERMs to guide visually impaired people [45]. A 64-actuator tactile jacket was also created by Lemmens et al. to enrich movie viewing experiences [20]. Similarly, West et al. created the Body:Suit:Score system with 60 ERMs to enhance musical experiences [53].

Despite the great potential of spatialized vibrotactile feedback systems, previous implementations of such systems revealed scalability and expressivity limitations. Systems that depend on direct connections between actuators and microcontroller units (MCUs) exhibit limited scalability, as the number of actuators is intrinsically restricted by the available GPIOs or PWM channels on a MCU [17, 37, 52]. Expanding the number of actuators requires multiple MCUs, resulting in exponential increases in system complexity and communication overhead. For systems that use custom drivers and data bus protocols to drive actuators indirectly, the multi-layer connections support more actuators but compromise the vibration expressivity [20, 21, 53]. These systems cannot render complex waveforms such as audio signals due to bandwidth limits, thereby limiting the types of actuators that can be used to simple actuators like ERMs. Moreover, while various toolkits have been developed to assist vibrotactile design, they typically employ direct or multi-layer connections and face the same limitations (e.g., TECHTILE [29], VITAKI [26], and VHP [7]).

To overcome these limitations, this work proposes VibraForge, an open-source vibrotactile toolkit that leverages a scalable, modular design to enable the rapid prototyping of vibrotactile systems. VibraForge consists of control units and vibration units. Control units are responsible for receiving vibration control commands from a Bluetooth server and sending them to vibration units. Vibration units are self-contained modules with an actuator driven by their own microcontroller. Vibration units are connected to a control unit via a chain-connection method, where a single GPIO pin on a control unit can control up to sixteen vibration units via a custom UART protocol. To better support spatial pattern authoring, we also designed an accompanying GUI Editor that enables haptic designers to intuitively author multi-actuator patterns.

Technical evaluations of VibraForge showed that the toolkit has a high bandwidth (200 Hz), low latency (16 ms), and can reliably render complex audio signals in real time. Through three case studies, we demonstrate that VibraForge not only supports the efficient replication of existing research prototypes such as phonemic displays [37], but also improves user performance in various applications such as virtual reality fitness gaming and robot teleoperation.

The contributions of this work are:

- an open-source hardware and software toolkit, VibraForge, that supports the design of spatialized vibrotactile feedback systems,
- technical evaluations of VibraForge, including power consumption, bandwidth, latency and acceleration, etc., and
- case studies demonstrating replicated research prototypes and novel applications enabled by VibraForge.

2 Related Work

Herein, we provide an overview of existing vibrotactile toolkits, highlighting their strengths and limitations. Then, we summarize previous spatialized vibrotactile feedback systems and discuss the benefits and drawbacks of their design strategies. Lastly, we summarize existing software for vibration pattern design.

2.1 Existing Vibrotactile Toolkits

Vibrotactile actuators such as eccentric rotation mass vibration motors (ERM), linear resonant actuators (LRA), voice coil actuators (VCA), or piezoelectric actuators (PIEZO) create mechanical vibrations via electromagnetic motions. Each actuator type and model has distinct voltage ratings, frequency responses, and vibration characteristics, making it challenging for designers to explore different configurations for wearable tactile devices. To assist the design process, various vibrotactile toolkits have been developed. These toolkits abstract low-level technical details and offer intuitive design interfaces, so researchers and designers can concentrate on creating vibration effects without being overwhelmed by the complexities of electronic, mechanical, and software design.

Vibrotactile prototyping toolkits can be categorized into those for education purposes and those for design purposes. Toolkits for education purposes often have simple compact widgets and limited output channels so they can teach novice users about the basics of haptic feedback and the design of simple vibration effects. For example, the TECHTILE toolkit was composed of a microphone, a signal amplifier, and two actuators [29]. Audio signals were captured by the microphone, amplified through the amplifier, and delivered to the actuators. The Stereohaptics toolkit used a similar principle, enabling users to create, record, and playback simple haptic waveforms, making haptic design accessible through familiar audio tools [15].

Toolkits for design purposes often have more sophisticated hardware and complex functionality that enable researchers and designers to explore different configurations of actuators and vibration patterns. For example, VITAKI was an ERM-based toolkit targeting virtual reality and video game applications [26]. It used shift registers and PWM drivers to control up to sixteen ERMs (i.e., the number of channels available on the PWM drivers). TactJam supported up to eight actuators and was intended for collaborative design [54]. With Syntacts, designers could design vibration effects in a GUI Editor and then send them through an audio amplifier to actuators for testing [33]. The Syntacts audio amplifier, however, only supported eight channels and was too bulky to wear on the body. To enable the creation of wearable devices, the VHP toolkit used a custom designed PCB and flexible PCB connectors so devices could fit to different body shapes [7]. The Nordic nRF52840 microcontrollers used in the toolkit had three PWM modules that generated 12 channels of PWM signals to drive the actuators.

While existing toolkits provide rich features to support vibrotactile design, they are limited by their scalability, particularly in the number of actuators they can support. This is because the actuators are directly controlled via GPIO pins or PWM channels on MCUs, with most offering a maximum of 16 channels. As vibrotactile feedback continues to be used for more complicated tasks and applications, these toolkits cannot meet the demands of emerging systems. Therefore, when designing VibraForge, we drew insights from previous spatialized vibrotactile feedback systems and propose a new connection method to overcome these scalability limitations.

2.2 Spatialized Vibrotactile System Design

Two connection methods have been often deployed When building spatialized vibrotactile systems (Figure 2). Some have tried maintaining a "direct connection" to each actuator by adding more controllers to a system and thus increasing the number of PWM channels. For example, Wang et al. used PWM outputs from four Arduino Mega2560 boards to drive their facial display [52]. Similarly, Kaul et al. used four Arduino Nanos to drive 17 actuators mounted on their HapticHead display [17]. Reed et al. used three custom-built audio amplifier boards to send individual waveforms to 24 actuators on a sleeve to deliver speech phonemic cues to deaf users [37]. Although complex waveforms could be sent to the actuators via direct connections, communication overhead and system complexity increased significantly when the system scaled up.
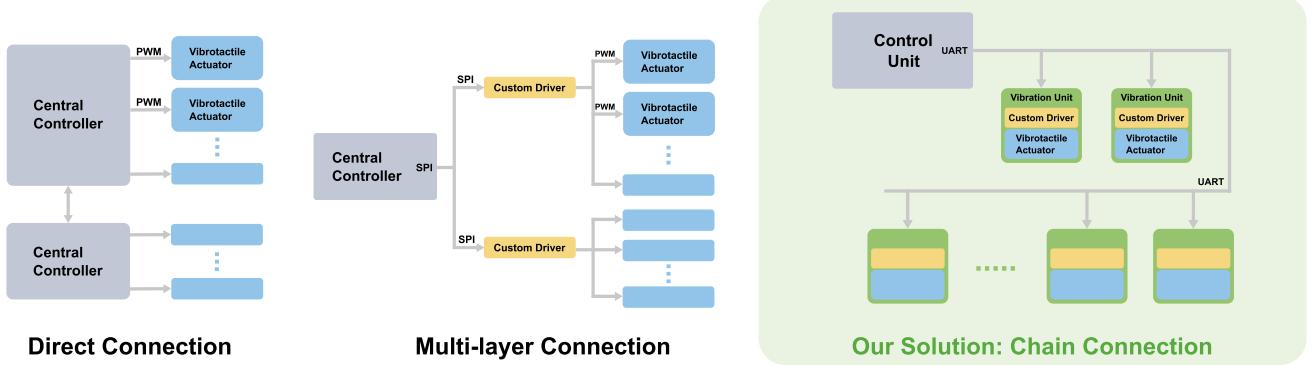
Others have explored using "multi-layer connections" that employ custom driver boards to interface between the microcontrollers and actuators [20, 21, 45, 53]. On the input side, these driver boards received vibration commands from a main controller board via data bus protocols such as I2C [45] or SPI [20, 53]. On the output side, they converted vibration commands to PWM signals and drove four [20], six [53], eight [45], or sixteen [21] actuators. By leveraging a multi-layer design, these systems were more scalable than previous solutions, however, the indirect control afforded by the data bus protocols made it impossible to process complex waveforms. This limited the expressivity of the actuators so only simple actuators like ERMs were used in these systems.

There is thus a trade-off between system scalability and actuator expressivity: a system that supports fine-grained control of individual actuators is difficult to scale, and vice versa. In this work, we propose a chain-connection method (Figure 2), where actuators are encapsulated into self-contained vibration units and driven by their own PCB drivers. Chains of vibration units are connected to a central MCU. Each PCB driver receives commands from the central MCU and generates PWM signals for its own actuator, thus supporting scalability. The high bandwidth and low latency control of vibration characteristics enables actuators to render complex waveforms, thus enhancing expressivity

2.3 Software for Vibration Pattern Design

Vibration pattern design is crucial in spatialized vibrotactile feedback systems. For a comprehensive review of previous design software and their public availability, please see [48]. Here we discuss the common interfaces used to design single-actuator waveforms and multi-actuator patterns.

For single-actuator design, one common method is to use predefined waveforms. Designers start with standard waveforms such as sine, square, or triangle, multiply them to generate new waveforms, and modify them using envelopes [33]. While this method provides authoring flexibility, it requires that designers have a deep understanding of how waveforms can be composed through combinations of standard signals. This makes it difficult for novice designers to use. Others have created editors that enable the direct manipulation of waveforms. With these editors, designers have access to two parallel windows to edit vibration amplitude and frequency. For example, in the Macaron Editor, designers could directly move keyframe dots to modify a waveform, making the design process more intuitive [40]. This method was later adopted by several industrial software tools, including Meta Haptics Studio [27].

Figure 2: Depiction of the direct connection (left) and multi-layer connection (middle) methods commonly used in spatialized vibrotactile systems and our proposed solution (right), which uses a chain connection method.

For multi-actuator authoring, the design focus shifts from waveform editing to pattern design. Designers compose vibration paths across multiple actuators on a canvas and the system converts the drawn paths to a series of vibration commands that are executed over time. For example, Schneider et al. utilized phantom sensation illusions to interpolate points between actuators to produce more continuous vibration sensations in Mango [39].

VibraForge's GUI Editor combines single-actuator waveform design with a multi-actuator canvas to support the efficient design for spatialized systems. It provides a library of standard waveforms and envelops for composing complex waveforms. It is also compatible with existing tools such as Syntacts [33] and VIREO [48] to support the reuse of workflows and prior knowledge. The canvas shows a virtual layout of actuators that replicates the physical chain connections so designers can use it as reference when designing spatial patterns.

3 VibraForge Design

As the discussion of prior work highlighted, there is a trade-off between actuator expressivity and system scalability. We aimed to mitigate this trade-off by using a novel chain-connection method with custom designed vibration units and control units. This section provides an overview of the VibraForge toolkit[1], including its design requirements, vibration units, control units, chain-connection method, signal segmentation algorithm, and vibration pattern GUI Editor.

3.1 Design Requirements

VibraForge's design requirements were drawn from design challenges discussed in prior work on wearable spatial vibrotactile design or were identified by the authors based on prior prototyping experiences.

From spatial vibrotactile design, three design requirements were proposed:

- *DR1: Scalability*. The toolkit should support large quantities of actuators. Designers should be able to use different types of actuators such as ERM, LRA, and VCA.

- *DR2: Fine-Grained Control*. The toolkit should provide fine-grained control of individual actuator vibration characteristics such as intensity and frequency to support expressivity. Regardless of the number of actuators employed in the system, the control mechanism should have a high bandwidth and low latency.

- *DR3: Usability*. The toolkit should provide an intuitive design interface for authoring vibrotactile spatial patterns. To leverage designers' prior design knowledge, the toolkit should be compatible with existing vibrotactile waveform editors such as Syntacts [33], VIREO [48], and Meta Haptics Studio [27].

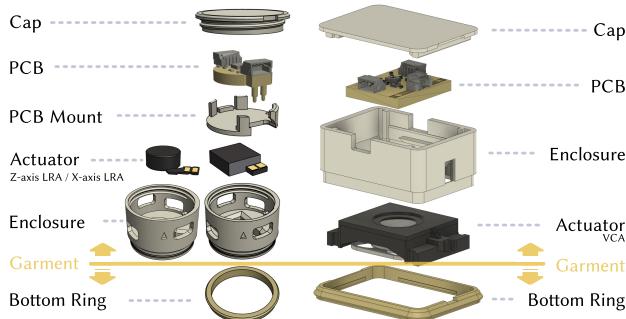From wearable design, two design requirements were proposed:

- *DR4: Flexibility*. The toolkit should enable designers to modify settings, reprogram behaviors, and physically rearrange components without difficulty. This requirement was derived from design challenges faced by West et al. when attempting to relocate or replace components that were sewn onto a garment [53].

- *DR5: Portability*. Devices built with the toolkit should be portable and powered by batteries, communicate with external devices via wireless protocols, and use lightweight components to ensure a comfortable wearing experience.

3.2 Vibration Units

The vibrations units were designed as self-contained modules that could drive actuators by themselves. Each vibration unit consisted of a vibrotactile actuator, a custom designed PCB, and 3D printed components (Figure 3). We designed three versions of the vibration unit to support a range of actuators (DR1).

3.2.1 Vibrotactile Actuators. An X-axis, square-shaped, 9.5 mm x 9.5 mm LRA manufactured by the Jinlong Machinery and Electronics Company (Model L959535) [25] was chosen for the vibration unit. Its resonant frequency was 170 Hz. When powered by 0.9 Vrms, it had a maximum current of 145 mArms and an acceleration of 0.65±0.12 Grms. The other two actuators were a Z-axis LRA manufactured by Jinlong (Model G0832) [25] and a VCA manufactured by PUI Audio, Inc. (Model HD-VA3222) [36]. The Z-axis LRA was cylindrical-shaped, with an 8 mm diameter and a height of 3.2 mm.
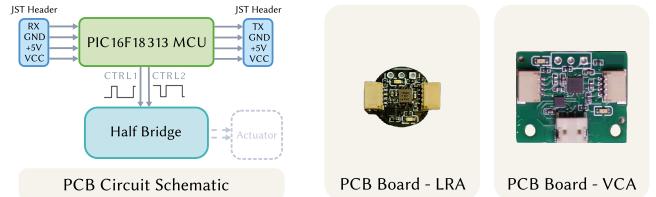

Figure 3: (Left) Exploded views of the small vibration unit for X-axis LRAs, including the cap, the PCB, the PCB mount, the enclosure, and the bottom ring. Z-axis LRAs shared most components except the enclosure. (Right) Exploded view of the large vibration unit for VCAs with a similar structure. Note that the bottom ring was placed underneath the garment to attach the unit.

Its resonant frequency was 235 Hz. When powered by 1.8 Vrms, it had a maximum current of 90 mArms and an acceleration of 0.50±0.08 Grms. The VCA was 32 mm x 22 mm and had a frequency response range from 80 - 500 Hz. When driven by a 133 Hz 1.5 Vrms signal, it had a maximum acceleration of 2.52 Gp-p.

These actuators were chosen because they cover a variety of vibration directions and intensities, thus satisfying the tactile perception requirements of different body parts (i.e., spatial acuity [9] and perceived intensity [49]). Since they all share the same PCB schematics and similar enclosure structures, in the following sections we only describe the vibration unit designed for the X-axis LRA. The other designs can be seen in Figure 3.

3.2.2 Printed Circuit Board Design. Due to bandwidth limitations, as the number of actuators increases it becomes difficult to drive them using real-time PWM signals sent from a central unit. Therefore, we allocated an MCU to each vibration unit to drive each actuator. Each vibration unit had its own driver PCB to receive UART commands from a central board and provide fine-grained control over vibration characteristics. The driver PCB had a PIC16F18313 MCU, a DRV8837 H-bridge motor driver, two 4-pin JST-SH headers, two pogo pins, and supportive components (Figure 4). The MCU was used to transmit commands and generate waveforms. The JST headers were used as input and output for the power and signal connections. The number of connector pins was minimized to four (i.e., actuator power, MCU power, ground, and data). The pogo pins were mounted on the bottom to ensure direct contact with the actuator pads.

When the MCU received a "start" signal, it sent a waveform to the H-bridge to modulate the actuator. We utilized the PWM generator inside the MCU for fine-grained control over the actuators' intensities, frequencies, and waveforms. Because there is no address system for each unit, designers only need to load their program into the MCU once. They can then plug-and-play each unit as desired. The communication protocol is further described in Section 3.4.

Figure 4: The circuit schematics and PCB board designs of vibration units.

3.2.3 3D Printed Enclosures. The 3D printed enclosures were designed to enclose each vibrotactile actuator and PCB board and provide a method to easily attach and detach each vibration unit to a garment (DR4). Similar to the size of a coin, the enclosure was compact and lightweight, enabling for a comfortable wearing experience (DR5).

The enclosure consisted of four parts: a cap, a PCB mount, an enclosure, and a bottom ring (Figure 3). The cap employed a screw-nut fastening system, which added top-down pressure to stabilize the PCB. The PCB mount offered protection for the PCB and aligned the pogo pins directly with the actuator pads. This improved the robustness of the vibration unit as no soldering was required during assembly. The enclosure served as a protective housing for all the parts. It had two holes for JST connectors, as well as two asymmetrically positioned vents for passive heat dissipation. The bottom ring was designed to be positioned beneath the garment to maintain direct contact with the skin without the need for any sewing (DR4).
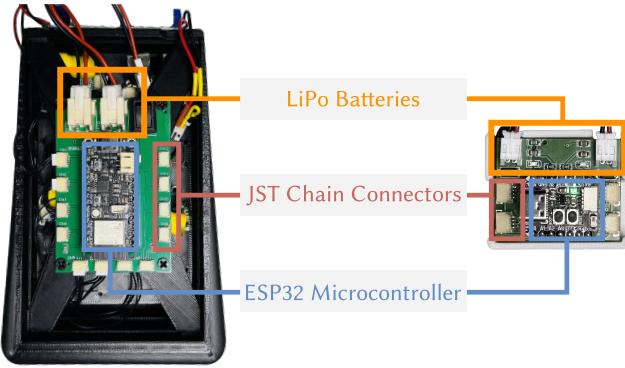
The same design can be easily adapted to other vibration unit structures for other types of actuators. In Figure 3, other vibration units that share the same PCB schematics and slightly different enclosure designs are shown.

3.3 Control Unit

The control unit was designed to be fully portable (DR6). It consisted of a PCB extension board with an ESP32 microcontroller (MCU), two LiPo batteries, and a 3D-printed enclosure (Figure 5). Multiple versions of control units were designed to accommodate different projects, ranging from simple arm bands to more spatialized body displays. They varied in the (a) number of chains supported, (b) model of the ESP32 board, and (c) battery capacity.

We designed a large control unit that supported up to 8 chains of vibration units and 16 units per chain, for a total of 128 vibration units. The extension board drew power from two serial 3.7 V 8200 mAh LiPo batteries. It stabilized the voltage at 5 V through two power regulators: a linear regulator that powered all the MCUs and a buck regulator that powered all actuators in the system. These separate power supplies ensured that MCUs operated on stable voltages without interference from the actuators' power consumption. The extension board had eight JST connectors. Each connector was connected to a GPIO pin on the ESP32 MCU and drove a chain of vibration units. The MCU was an Adafruit Feather ESP32 V2[2]. It was responsible for converting external Bluetooth commands to UART commands and sending them to the corresponding vibration unit.

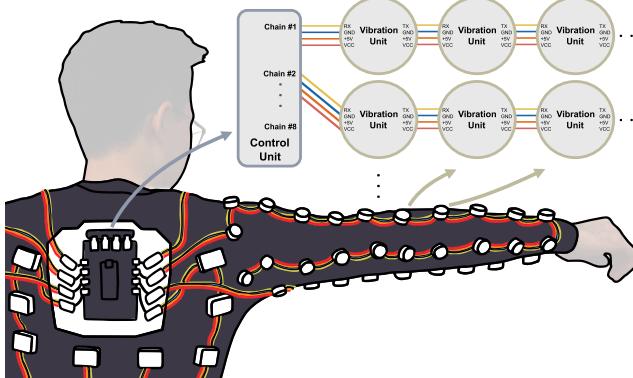[2]https://learn.adafruit.com/adafruit-esp32-feather-v2

Figure 5: The control unit, which contained an ESP32 MCU, an extension board, chain connectors, and LiPo batteries.

Since the number of required vibration units varies across different tasks, we also designed a small control unit with fewer supported chains. This compact unit design contained two 3.7 V 500 mAh LiPo batteries and an Adafruit QT Py ESP32-S3 [3]. This MCU had four general purpose GPIO pins that could be used for chain connections.

3.4 Chain-Connection Method and Data Communication

VibraForge differs from existing toolkits due to its use of a chain-connection method (Figure 6). This method enables the toolkit to address over 100 vibration units using a single control unit (DR1) and support the simultaneous control over the vibration characteristics of multiple actuators (DR2).
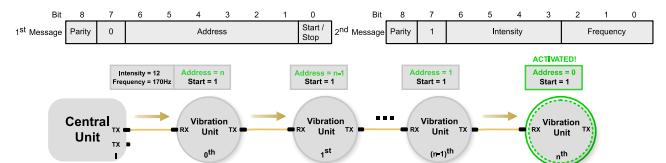

Figure 6: The multiple chain-connection method used within VibraForge, depicting the connections required for a given chain.

As explained in Section 3.3, each GPIO pin on the ESP32 MCU was routed to a JST connector on the extension board. Because vibration units shared the same JST connectors, they could be connected in chains and receive commands through the same GPIO pin.

There were two options for the data transmission design. The first option was to use data bus protocols such as CAN or I2C, where all the vibration units would receive the same command simultaneously and use the command's address to determine whether the command was intended for itself. While this option would allow commands to be transmitted with a minimal latency, the vibration units would need to be programmed with different addresses prior to use, which would be burdensome when working with multiple units. Instead, we decided to send commands through each chain using a serial protocol, where each unit would receive a command, process it, and resend it to the next unit. While this may increase the overall latency, it allowed each unit to operate with a "virtual address", enabling independent plug-and-play control without additional programming.
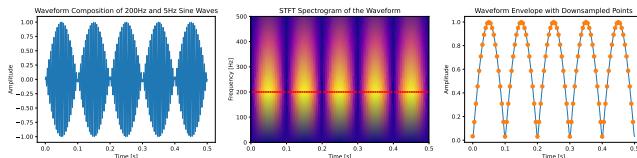

Figure 7: The two-byte message bits definition and an example of how vibration commands are transmitted along a chain using the "virtual address" system.

Vibration commands were transmitted via a UART protocol at 115.2 kHz with parity checks (Figure 7). Each command consisted of two bytes: the first byte encoded the address and start/stop status of the target vibration unit, and the second byte detailed the vibration characteristics. Upon receiving the first byte, a vibration unit checked the address. A non-zero address indicated the command was for subsequent units, prompting the unit to decrement the address and forward the message. This process continued until the address reached zero and signified the targeted unit, which then acted based on the start/stop bit: '1' (START) awaited the second byte, while '0' (STOP) disabled its actuator. When a unit was "signified", it waited for the second-byte message, which contained details about vibration characteristics, including the 16 intensity levels (evenly distributed from 0% to 100% duty cycle) and 8 frequencies (i.e., 123, 145, 170, 200, 235, 275, 322, and 384 Hz). The frequency levels were chosen so that each level was 1.18 times higher than the previous level. This followed Weber's factor for vibrotactile frequency just-noticeable differences [34].

3.5 Signal Segmentation Algorithm

To render complex waveforms such as audio signals on vibration units (DR2), we implemented a signal segmentation algorithm that converted high-frequency input signals to low-frequency serial commands (Figure 8).For instance, consider an input waveform of 44100 Hz audio signals that composed of 200 Hz and 5 Hz sine waves. High-frequency components (200 Hz) in the waveform were extracted using a short-time Fourier transform (STFT), which captured how the dominant frequency changed over time. Low-frequency components (5 Hz) were extracted using a Hilbert transform, which captured the envelope of the waveform. Both components were
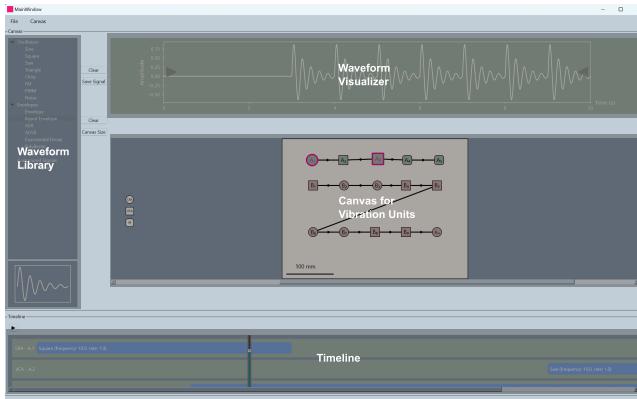
downsampled to 200 Hz and sent to vibration units. Since our frequency levels ranged from 123 Hz to 384 Hz, the segmentation threshold was set to 100 Hz. Frequency components above 100 Hz were rendered using frequency levels, while components below 100 Hz were rendered using intensity levels. The technical evaluations in Section 4 demonstrated that this algorithm could reliably render complex waveforms such as audio signals using serial command control.

Figure 8: The signal segmentation workflow, showing how low frequency and high frequency components were extracted from a complex waveform.

3.6 GUI Editor for Designing Spatialized Patterns

We designed a GUI Editor (Figure 9) to enable designers to create spatialized vibration patterns (DR3). This editor differed from previous editors in that it combined the waveform editing from single-actuator design software with the canvas design used for multi-actuator design. The key components of the GUI Editor were:

Figure 9: The layout of the GUI Editor, with its Waveform Library, Waveform Visualizer, Canvas, and Timeline.

- **Waveform Library**: The Waveform Library contained waveform templates such as oscillators and envelopes. Designers could use them to create a custom waveform and save it to the Waveform Library. In addition, designers could import waveforms in the JSON format from popular waveform editors such as Syntacts [33], VIREO [48], and Meta Haptics Studio [27] so they could reuse previous designs (DR3).
- **Waveform Visualizer**: The Waveform Visualizer served two purposes. When designers were creating a custom waveform, they could drag and drop waveform templates from

the Waveform Library onto the Waveform Visualizer and it would show the combined waveform. When designers were composing vibration patterns for individual vibration units, the Waveform Visualizer also show the composed waveform for the selected unit.
- **Canvas**: The Canvas enabled designers to drag-and-drop vibration units onto the canvas so they could be connected into chains that mimicked the physical layout of the vibration units. In addition, a "Create New Chain" function enabled designers to generate vibration units in batches and arrange them using grid layouts. When clicking on a unit, its corresponding waveform would show up in the Waveform Visualizer.
- **Timeline**: The Timeline served as a high-level summary of the vibration patterns. The blue blocks showed the range of time when the vibration unit was activated. By moving the slider left and right, the vibration units that were activated at that moment would be highlighted in the Canvas so designers could understand the vibration patterns over time. By clicking the "Play" button, the waveform would be converted into vibration commands with timestamps and sent to the control unit via Bluetooth.

In a typical spatial pattern design workflow, designers would begin by creating a new design and specify the number of chains and actuators to be used in their system. The GUI would then generate a default Canvas with the designated number of actuators. Designers would then freely adjust the layout to match their physical device. Then, designers would assign waveforms to each vibration unit and specify the start and end time. Each waveform could be designed from scratch using the waveform templates in the Waveform Library, or imported from other software. To review the spatialized patterns, designers would click on any vibration unit on the Canvas and the Waveform Visualizer would show the corresponding waveform. Alternatively, designers could see a high-level summary of how the vibration patterns of multiple units would change over time by moving the Timeline slider.

Additionally, we also provided a Python server with VibraForge APIs, so that vibration units can be directly activated and controlled via code (DR3). Later in Section 5, Python servers are integrated into external programming environments (e.g., Unity) to trigger real-time feedback for applications.

4 Technical Evaluation

In this section, we summarize various performance metrics of the VibraForge toolkit to demonstrate how it fulfills the aforementioned design requirements. All tests below used vibration units of X-axis LRAs and the small control unit with default level-7 intensity (50% duty cycle) and 170 Hz frequency, unless otherwise stated.

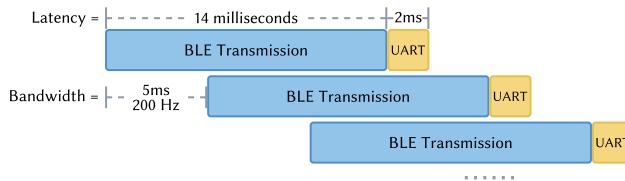### 4.1 Latency

Having fine-grained control (DR2) over the actuators in a vibrotactile system means that the data transmission of the system must have low latency and high bandwidth. To determine if the latency of VibraForge was below the human perceivable latency threshold of 45 ms [51], we measured the end-to-end latency of a PC to a control unit (i.e., the transmission latency of the Bluetooth Low

Energy protocol (BLE)), and of a control unit to a vibration unit (i.e., the latency of the custom UART protocol).

To measure the PC to control unit latency, a BLE write GATT command was sent and immediately followed by a BLE read GATT command. A Python timer recorded the write command sending time and read command receiving time and computed the difference as the latency. We repeated this measurement 100 times and found that the average round-trip transmission latency was 26.67±6.53 ms. Therefore, the one-way transmission latency was approximately 14 ms. For the control unit to vibration unit latency, repeated measurements using an oscilloscope showed that each command took 125 us to be transmitted from one unit to the next unit on a chain. Therefore, latency on the longest chain with 16 actuators would accumulate to 2 ms. Adding the two measurements together would thus result in an overall latency of approximately 16 ms, which would be virtually imperceptible by humans [51].

4.2 Bandwidth

The bandwidth of VibraForge was the number of commands that could be sent and processed per second by control units and vibration units. Bandwidth differs from latency in that asynchronous BLE commands could be continually sent without needing to wait for the previous command to be received (Figure 10). Therefore, bandwidth represented the refresh rate of the vibration commands for each actuator. Since BLE commands (millisecond level) would be processed much slower than UART commands (microsecond level), the bandwidth bottleneck of the system depended on the processing speed of BLE commands on the control unit.

Figure 10: Data transmission bandwidth breakdown.

To measure VibraForge's bandwidth, we wrote a Python testing program to repetitively send standard BLE commands without latency, and another program on the ESP32 (i.e., control unit) to receive the commands and print the receiving time to the serial port. The command processing time was computed as the difference between the receiving time for each command. We sent 1, 000 BLE commands with five UART commands in each of them, and computed an average processing time as 2.96 ± 1.20 ms. Therefore, for the actual operation of the toolkit, we set the BLE command sending delay to be 5 ms with some tolerance. This means that the control unit could simultaneously update five vibration units at 200 Hz refresh rate, satisfying the signal segmentation algorithm requirements.
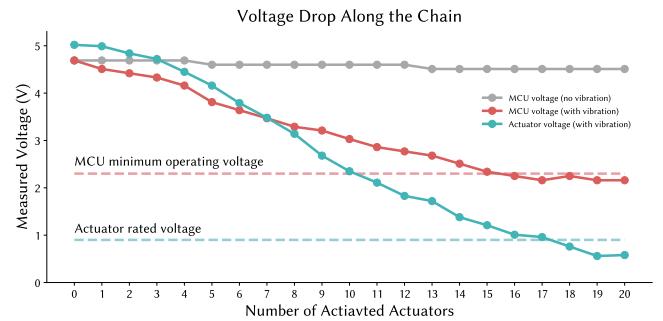
Although vibration units were not directly driven by continuous audio signals from the central MCU, the high bandwidth ensured that they could approximate continuous waveforms using discrete commands (DR2).

4.3 Power Consumption and Battery Life

Power consumption measurements would provide an estimation of how long wearable devices built using VibraForge would function before recharging was necessary (DR6). Power consumption was measured using a bench power supply with a real-time measurement of voltage, current, and power. When only the control unit was connected, it consumed 106 mA current. When multiple vibration units were connected to the control unit, each vibration unit consumed 2.5 mA when idle. When a vibration unit was activated, the actuator consumed 150 mA. Therefore, for a vibrotactile arm display with one control unit and two full chains of vibration units (16x2), the idle current would be 186 mA and a 500 mAh LiPo battery would last 2.69 hours. Even in extreme situations such as multimedia entertainment, running two actuators consistently would consume 486 mA, leading to 1.03 hours of operating time. For a larger system with four chains of vibration units (16x4), the idle current would be 266 mA, so a 8200 mAh LiPo battery would last 30.83 hours. In an extreme situation where eight actuators were continuously activated, the overall current would be 1460 mA and the battery would last 5.62 hours.

4.4 Maximum Number of Vibration Units On Each Chain

Here we described how the maximum number of units on each chain was determined (DR1). Due to incremental wire resistance along the chains, voltage drops will become more significant when additional vibration units are connected or when more units are activated simultaneously. To maintain stable operation, the MCU power line must sustain a minimum of 2.3 V (i.e., the MCU's minimum operational voltage) and the actuator power line should not fall below 0.9 V (i.e., the LRA-X actuator's rated voltage).

Figure 11: Voltage drops on the MCU power line and the actuator power line with an increasing number of actuators being activated.

We measured the voltage drop threshold by incrementally activating vibration units on the chain. An oscilloscope was used to measure the MCU power line and actuator power line at the last vibration unit of the chain. If the last unit maintained a stable voltage, then one could be certain that other units were stable as well. Figure 11 shows the relationship between voltage drop number of actuators activated. When the number of actuators surpassed 16, the MCU voltage dropped below the operating range. When it

surpassed 17, the actuator voltage dropped below its rated voltage. Therefore, control units could support up to 16 actuators on each chain.

4.5 Vibration Characteristics

To demonstrate that the toolkit can provide precise control over different vibration characteristics (DR2), we measured the acceleration of vibration units at various intensities and frequencies.

For intensity, we used an ADXL345 accelerometer for the measurements. Since most ADXL345 breakout boards are larger then the vibration unit itself, we designed a FlexPCB version and attached it between the skin and the vibration unit for precise measurements. A vibration unit was mounted on a sleeve and placed near the wrist of a human subject. During the testing, the vibration unit was activated for three seconds at each intensity level and the vibration frequency was kept constant at 170 Hz. The average acceleration was measured as the RMS value over the 1-second period of stable vibration. Acceleration was found to increase Monotonically as the intensity level increased (Figure 12 Left).

Similarly, for frequency, the vibration unit was placed near the wrist and activated for three seconds at each frequency level with level-3 intensity (25% duty cycle). To obtain the main frequency response, the acceleration data was processed using a fast Fourier transform. We found that the main frequency response was well aligned with the designated frequency at each level, thereby demonstrating the precise PWM signal control (Figure 12 Right).

4.6 Waveform Approximation

Because vibration units are controlled via discrete UART commands rather than continuous audio signals, one might worry that this would reduce the expressivity of the vibration waveform delivered through the units. Here we show that by utilizing high-bandwidth communication (DR2), VibraForge was capable of approximating audio signals and creating similar feedback. We used standard vibration waveforms from VibViz [41], converted them from WAV files to HAPTIC files through Meta Haptics Studio [27], and sent them to the GUI Editor. The editor automatically converted the haptic files to a sequence of vibration commands that were used to drive a vibration unit. We then measured the acceleration of the vibration unit and compared it with the original audio envelope. The results showed that they were closely approximated to each other (Figure 13).

5 Case Studies

We conducted three case studies to demonstrate the potential usage of VibraForge. The first case study showcased how VibraForge can be used to efficiently reproduce previous research prototypes. The second and third case studies showed how VibraForge can be used to build novel spatial vibrotactile systems that improve user performance in various applications.

5.1 Case Study 1: Replicating A Phonemic-Based Tactile Display

In Reed et al.'s work, a phonemic-based tactile display was designed to aid speech communication [37]. This tactile display used 24 actuators on the forearm to deliver unique tactile codes of 39

English phonemes to deaf and hard-of-hearing users. Phonemes were mapped to vibration patterns using frequency, waveform, amplitude, spatial location, and movement. Although the display was shown to be useful, the process of creating the display was challenging. To drive the 24 actuators independently, the researchers had to use a desktop audio device (i.e., a MOTU 24Ao) with three custom amplifier boards to provide output signals, which made the whole system bulky and unable to be used in real-world situations. In this section, we demonstrated how the VibraForge toolkit could be utilized to simplify the development process and enhance the portability of the display.
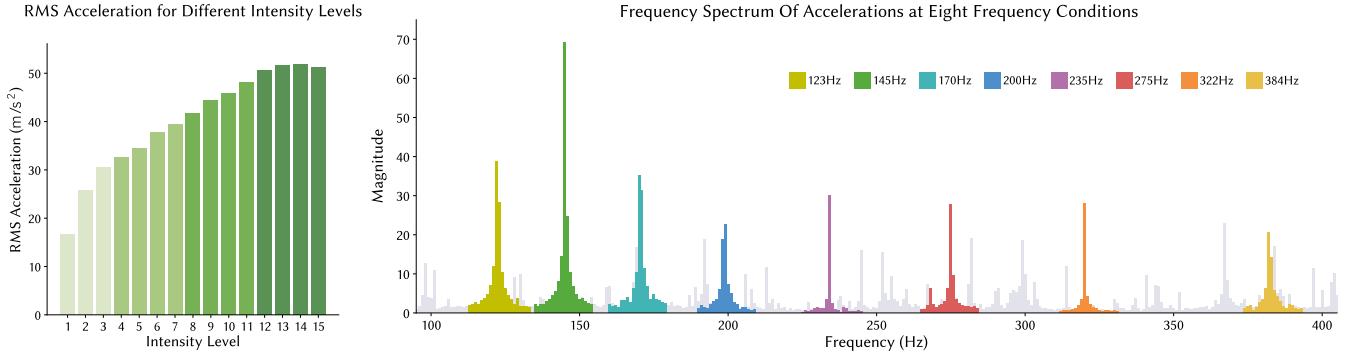
5.1.1 Vibrotactile System Development. The first step in replicating the phonemic tactile display was to determine the layout of the actuators and the chain connections. Since the original display used a 4x6 layout on the forearm, we decided to use four chains with six vibration units on each chain and a small control unit.

The second step was to assemble the display. The display required a sleeve garment and multiple modules from the toolkit (Figure 14). We first assembled the vibration units, then attached them to the garment and connected JST wires between them. The control unit was attached to the garment and four chains of vibration units were then connected to the control unit. The mapping between chains and connection ports followed the order of actuators described in the original paper. The whole assembly process took 16 minutes.
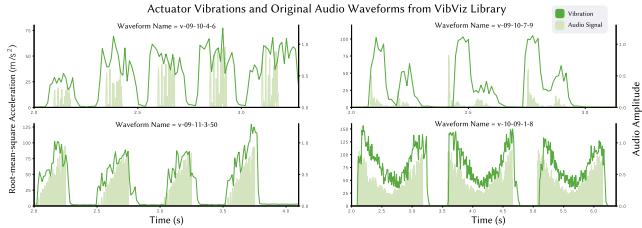
Once the hardware was built, the next step was to design the vibration patterns. Each phoneme needed to represented by a distinctive waveform that incorporated a high-frequency primary component and a low-frequency envelope, with a duration ranging from 100 to 480 ms. Consonants were designed as static patterns with multiple actuators being activated concurrently, whereas vowels were designed as dynamic patterns with actuators being activated in succession. The GUI Editor streamlined the process of creating the large variety of waveforms.

For example, to design the waveform for the consonant "V" (IPA symbol /v/), we first dragged a sine wave into the Waveform Visualizer and set the frequency to be 300 Hz (primary component). Then we multiplied it with another sine wave at 8 Hz (envelope). The combined waveform was then saved in the Waveform Library as "Consonant_B". Alternatively, when the envelope was not available in the Waveform Library (e.g., cos^2 for consonant "H"), we designed the waveform in Syntacts and imported the .CSV file into our GUI Editor. We then used the "create new chain" function to create a virtual layout with four chains of vibration units on the canvas. We selected each of the designated vibration units (i.e., i6, ii6, iii6, iv6), assigned the custom waveform, and set the time range from 0 to 400 ms. Finally, we clicked the "play" button to validate the design. The average time for an experienced haptic researcher on our research team to create one phoneme was one minute and five seconds. Therefore, the whole phoneme design process could be finished in 42 minutes. While the original paper did not mention the waveform design time, we believe the streamlined waveform creation process led to a significant reduction of design time.

5.1.2 Lessons Learned. The use of VibraForge to create the phonemic tactile display had several benefits. First, compared with the original device, our system was fully portable (DR5), with a small control unit and 24 vibration units directly attached to the sleeve.

Figure 12: The acceleration measurements at various intensity and frequency levels.

Figure 13: Comparison of actuator vibrations (green lines) and original audio waveforms (light green shaded areas) from the VibViz Library

Second, VibraForge simplified the overall development process. The hardware system was reduced to module assembly without soldering, and the waveform design was supported by the GUI Editor (DR3). Third, the flexible design (DR4) enabled personalization opportunities. The original paper mentioned that the inter-motor spacing of the actuators needed to be adjusted for different participants and that the process of adjusting the layout was rather tedious. With our system, because the vibration units could be easily attached and detached, researchers could easily adjust the positions even if the garment was worn on the arm, thus giving them much more flexibility.

Meanwhile, some challenges were found when using the toolkit. While designers could easily author patterns using the GUI Editor, it mainly served as a testing site, because the current design of GUI Editor only allows users to manually click the "play/stop" button to control the vibration sequences. This has constrained its use for real world interaction, where speech is segmented into phonemes and used to trigger corresponding patterns in real time. Right now the Python server does not support this kind of interaction so an additional API would be needed.

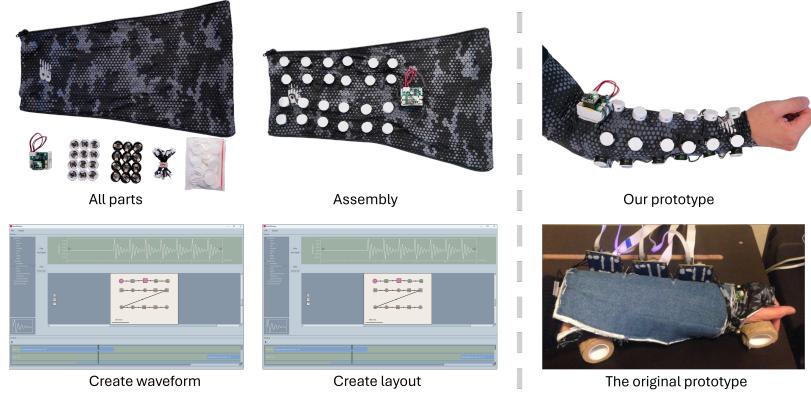## 5.2 Case Study 2: Fitness Gaming in Virtual Reality

Virtual Reality (VR) fitness applications like OhShape [18] and Supernatural [43] have gained popularity by turning traditional fitness routines into interactive, gamified experiences. However, users

often face challenges due to the absence of real-time, precise feedback about the physical positioning of their body [56]. While prior research has demonstrated the utility of vibrotactile feedback for movement guidance in rehabilitation applications [3, 35], systems used in these applications only deployed actuators on major joint locations such as the wrists or elbows, thus offering limited and coarse guidance. Inspired by OhShape [18] and the TV show "Hole in the Wall" [10], we built a VR fitness application in which participants were asked to adjust their body poses to mimic cutouts in virtual walls when their digital avatar moved through the wall (Figure 15). We used the VibraForge toolkit to construct an upper-body suit that would provide spatial vibrotactile feedback to improve training performance.
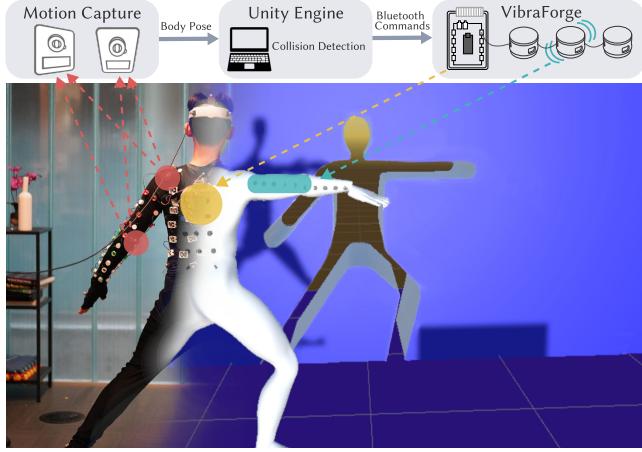
5.2.1 Vibrotactile System Development. The first step of our development was to determine the layout of vibration units on the body. We referenced prior work that measured spatial acuity [12] and determined that vibration units should be uniformly distributed on the arm with 4 cm of spacing between them and on the body with 8 cm of spacing between them. As a result, the suit had 36 vibration units on the body and 40 vibration units on each arm (Figure 15).

The second step was to mount the control units and vibration units on the garment, which was an exercise compression shirt. Since the total number of vibration units was within the upper limit of the large control unit, we only needed to use one control unit. It was worn on the upper back to help users maintain their balance during fitness training. Vibration units were mounted one-by-one and JST wires were used to connect them.

The third step was to integrate the hardware into the fitness game, which was built using the Unity Game Engine [47]. The physical layout of the vibration units on the suit was manually converted into a virtual layout and overlaid on the user's avatar, so that the virtual vibration units would move following the human body movement. Whenever a collision was detected between a virtual vibration unit and an obstacle, the system would determine which vibration units should be activated and send the vibration commands to the control unit through a Python server (Figure 15). Since collisions were binary events (i.e., collided / not collided), the vibration intensity was set to constant level (i.e., 7), as well as frequency (i.e., 170 Hz).

All parts | Assembly | Our prototype

Create waveform | Create layout | The original prototype

Figure 14: The development process of Case Study 1's phonemic tactile display re-implementation using VibraForge.

Figure 15: An overview of the pipeline used in Case Study 2's VR fitness game. Participant body movements were tracked by Vicon Motion Cameras and sent to a Unity scene for collision detection. Collision commands were then sent to a control unit on the participant's back through Bluetooth and converted to UART commands for the vibration units.

5.2.2 User Study Design. A preliminary study was conducted to evaluate the effectiveness of the vibrotactile suit on fitness training performance. The VR application was run on an Oculus Quest 2 headset [28] through a Quest Link connection to a PC (i.e., Alienware M15 Laptop with RTX 3060Ti GPU). Vicon Motion Capture cameras [50] were used to track the body poses of each participant and the poses were then mapped to the virtual avatar in the Unity scene. Nine participants (i.e., 4 Males and 5 Females; mean age = 25 years, SD = 3 years) were recruited to participate in the study, which lasted approximately one and a half hours, and each received $40 compensation. The study was approved by our institutional research ethics board and conducted in a university lab space.

During the study, each participant would experience two conditions, i.e., no feedback and vibrotactile feedback. In the no feedback condition, participants only saw the incoming virtual walls and their own avatar and had to rely on proprioception to adjust their body position. In the vibrotactile feedback condition, participants would feel vibrations at the body positions where collisions occurred between vibration units and walls. For each condition, 24 pairs of walls with different cutouts were presented in sequence. Two walls in each pair were identical. The first wall served as the "preview wall" to help the participant adjust their body pose, and the second wall served as the "test wall" to measure any body position changes. We measured the number of collisions occurring during each condition. Our hypotheses were that (1) vibrotactile feedback would cause fewer collisions than no feedback, and (2) test walls would have fewer collisions than preview walls.

5.2.3 Results. The average number of collisions between vibration units and walls was computed for each feedback condition. Then, a repeated measures ANOVA was conducted using SPSS to determine if feedback condition had an effect on the number of collisions that occurred. If there were significant differences between each feedback condition, pair-wise comparisons with a Bonferroni correction were then performed.

The RM-ANOVA determined that both feedback type ($F(1, 8) = 6.450, p < 0.05$) and wall type ($F(1, 8) = 5.582, p < 0.05$) had significant effects on the number of collisions. No interaction effect was found (($F(1, 8) = 3.936, p = 0.083$). For feedback type, the vibrotactile feedback (M=26.289, std=9.897) led to fewer collisions than the no feedback type (M=31.489, std=7.162, $p < 0.05$). For wall type, the test walls (M=27.552, std=10) led to fewer collisions than the preview walls (M=30.226, std=7.694, $p < 0.05$). Therefore, we concluded that both hypotheses were accepted.

5.2.4 Lessons Learned. The benefits of using VibraForge to build this high-density vibrotactile suit were multi-fold. First, scalability (DR1) enabled us to support a large number of actuators and provide highly localized feedback to guide body movement. Second, the modular and flexible design (DR4) made the system assembly process as simple as press-fitting the vibration units onto the garment and connecting wires between them. The entire process took less than 30 minutes. Compared with prior work that used sewing techniques [20, 53], our solution was more flexible and efficient. Third, the low latency and high bandwidth communication (DR2) helped deliver almost real-time vibrotactile feedback to users when

collisions occurred. Participants did not report any perceivable delays during the study.

Nonetheless, there were several challenges during the process. First, the integration of the vibrotactile system into the interactive game design was difficult. This was because we had to manually convert the physical layout of units on the body to a virtual layout in the Unity Engine and assign vibration units to their positions. This process would have been easier if we could detect the physical positions of the units and generate a virtual mapping automatically. Second, two participants reported that vibration units near the waist were not consistently perceived because body movement during fitness training would loosen the contact with garment. Our temporary solution was to tighten the suit so that movement was constrained. In the future, other methods would be needed to ensure that the vibration units maintained contact with the skin.

5.3 Case Study 3: Teleoperation Assistance for Unmanned Aerial Vehicle Operation

In this case study, we used VibraForge to design a wearable vibrotactile device to provide Unmanned Aerial Vehicle (UAV) operators with spatial information about obstacles. Current UAV teleoperation is dependent on visual feedback from onboard cameras, which presents significant challenges as a camera's limited field of view can hinder situation awareness and lead to potential collisions with obstacles. While there have been prior attempts to use haptic feedback to notify operators of nearby obstacles [5, 11, 57, 58], they required an operator to switch from using radio control (RC) controllers to hand-held haptic joysticks, only rendered force feedback in one direction at a time, or had fixed actuator layouts that could not represent lateral movements due to the absence of actuators on the side.

By creating a new vibrotactile feedback system using VibraForge, one could flexibly attach multiple vibration units to an operator's body that would render simultaneous spatial cues about surrounding obstacles while the operator was using an RC controller. This would enhance operator's collision avoidance ability and situational awareness in environments where visual feedback is limited.

5.3.1 Vibrotactile System Development. The first step of the development was to determine the layout of the vibration units on the body. Since there is no prior work that studied the mapping between tactile feedback positions on the body and human perception of 3D directions, we conducted a perceptual study with 10 participants to identify a mapping. The layout was initialized with 46 vibration units uniformly distributed on the body and optimized using user data. The final layout used 32 vibration units to achieve comprehensive coverage of the 3D space (Figure 16). More details on the perceptual study and layout optimization can be found in [13].

The second step was to affix control units and vibration units on the garment, which was a compression shirt similar to that used in Case Study 2. Thirty-two VCA vibration units were press-fit to the garment. A large control unit was placed on the operator's back. The vibration units were grouped into four chains, each covering the front left, front right, back left, and back right.

The third step was to integrate the hardware into the simulated environment, which was built using Microsoft AirSim [42] with Unreal Engine 4 and Python APIs. Since the mapping between vibration units and 3D directions were known from the perceptual study, when a potential collision was detected, the system would trigger the unit that had the nearest direction with the colliding object in the space. This avoided the need to reproduce the physical layout of the vibration units in the system. The risks of colliding with surrounding obstacles were computed using control barrier functions [2] and scaled to $[0, 15]$ to match the actuators' intensity levels. If the risk of an obstacle was above 0.5, the vibration unit in the corresponding direction would be activated by sending commands through a Python server.

5.3.2 User Study Design. A preliminary study was conducted to evaluate the effectiveness of the vibrotactile system on teleoperation performance. The simulation environment ran on an Alienware M15 laptop with RTX 3060Ti GPU. Twelve participants were recruited for the study (11 male, 1 female; mean age = 24 years, std = 3 years). Seven participants had previous experience operating quadrotors. Each study lasted 70 minutes and each participant received $40 as compensation. The study was approved by our institutional research ethics board.

During the study, each participant went through three feedback conditions in a randomized order: no feedback (**NA**), force shared control (**FSC**), and vibrotactile shared control (**VSC**). For the NA and VSC conditions, participants used an Xbox controller for input, which had control mechanism similar to an RC controller. The output device for the VSC condition was the vibrotactile system. For the FSC condition, a Novint Falcon haptic joystick [14] was used for both input and output, similar to previous studies [30, 38].
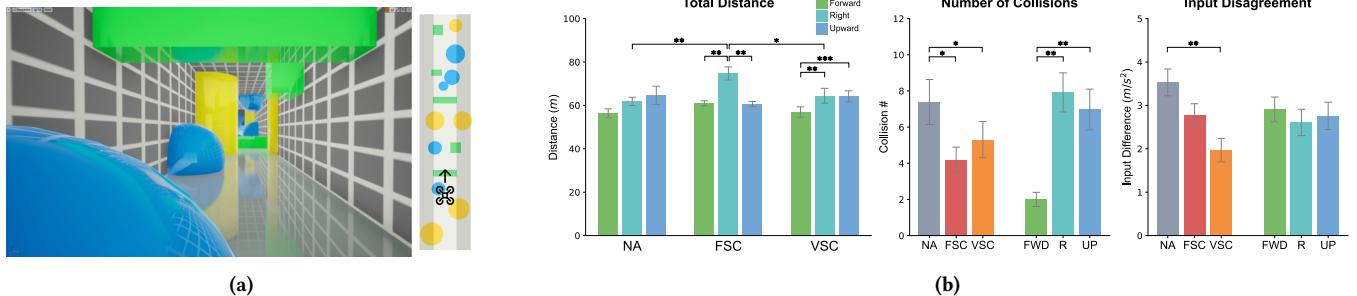
For each condition, the experimental scene was a $5m \times 5m \times 50m$ tunnel (Figure 17a) that required participants to steer quadrotors forward, right, or upward. Different flying directions were used to vary visual information capacity as obstacles are often outside the camera's field of view when flying right and upward. The tunnel contained fifteen objects that were randomly placed to avoid learning effects. Based on these conditions, we hypothesize that (a) vibrotactile feedback would enhance teleoperation performance compared to no feedback, and (b) the improvement would be more effective in the right and upward directions than the forward direction.

5.3.3 Analysis and Results. We evaluated three metrics: total distance travelled, number of collisions, and input disagreement (i.e., the deviation of user steering commands from the safe input range during flight). Using a two-way repeated measures ANOVA in SPSS, we assessed the effect of feedback condition and flying direction. Post-hoc pairwise comparisons with a Bonferroni correction were conducted for significant results (Figure 17b).

The RM-ANOVA determined that feedback condition had a significant effect on total distance travelled ($F(2, 22) = 3.585, p < 0.05$), as did with flying direction ($F(2, 22) = 17.868, p < 0.001$). The interaction between feedback condition and flying direction was also significant ($F(1.957, 21.527) = 4.916, p < 0.01$). The post-hoc analysis of the interaction effect revealed that FSC-R had longer distance than FSC-FWD ($p < 0.01$), FSC-UP ($p < 0.01$), NA-R ($p < 0.01$), and VSC-R ($p < 0.05$), while VSC-FWD had a significantly shorter distance than VSC-R ($p < 0.01$) and VSC-UP ($p < 0.001$).

(a)   (b)

Figure 16: (a) The vibrotactile device designed during Case Study 3 assists UAV operators with collision avoidance by delivering obstacle directions via multi-point vibrotactile feedback on the body. Operators not only see and feel visible obstacles (yellow), but also perceive obstacles outside their field of view (orange). (b) The layout of vibration units on the upper body. Red dots represent the initial layout with uniform spacing, while blue dots represent the final layout and are mapped to directions in the 3D space.

(a)   (b)

Figure 17: (a) A first-person view of the simulation environment used in Case Study 3's study, with randomly positioned obstacles and a top-down view on the right. (b) The results for the total distance travelled, number of collisions, and input disagreement metrics for the feedback conditions (NA, FSC, VSC) and flying directions (forward-FWD, right-R and upward-UP). The error bars represent the standard error of the mean (SEM), $^{*} = p < 0.05$, $^{**} = p < 0.01$, and $^{***} = p < 0.001$.

For the number of collisions, the RM-ANOVA determined that feedback condition had a significant effect ($F(2, 22) = 8.095, p < 0.01$), as did flying direction ($F(2, 22) = 15.653, p < 0.001$). The interaction between feedback condition and flying direction was also not found to be significant ($F(2.168, 23.846) = 1.910, p = 0.168$). The pairwise comparisons between feedback conditions revealed that NA caused more collisions than FSC ($p < 0.05$) and VSC ($p < 0.05$). The pairwise comparisons of flying directions revealed that flying forward caused fewer collisions than right ($p < 0.01$) and upward ($p < 0.01$).

For input disagreement, the RM-ANOVA determined that feedback condition had a significant effect ($F(2, 22) = 4.798, p < 0.05$), while flying direction did not ($F(1.254, 13.789) = 0.628, p = 0.476$). The interaction between feedback condition and flying direction was also not significant ($F(4, 44) = 1.566, p = 0.200$). The pairwise

comparison between feedback conditions revealed that there was less disagreement with VSC compared to NA ($p < 0.01$)

In summary, VSC caused fewer collisions and less input disagreement compared to NA, and achieved comparable performance with FSC. Additionally, the right and upward directions had more collisions than the forward condition, highlighting the increasing reliance users had on haptic feedback when under limited visual capacity. Thus, our hypotheses were accepted.

5.3.4 Lessons Learned. The process of using VibraForge to build the teleoperation vibrotactile system demonstrated several benefits. First, the flexible design (DR4) allowed us to iterate on the layout of vibration units rapidly during the perceptual study. From the initial 46 actuator layout to the final 32 actuator layout, we could easily adjust the positions of vibration units on the body and test various

spatial mappings. Second, the low latency vibration responses (DR2) also provided instant feedback for participants, which is essential in high speed teleoperation tasks. In addition, the varied intensities (DR2) provided more nuanced information for participants, helping them prioritize their responses towards obstacles with higher risks. Together, these benefits demonstrate the usability of the toolkit for a complex application.

We also encountered several design challenges. One issue was that participants reported that the same intensity level was perceived differently across body areas. Locations with bone contacts were perceived to be stronger, while those with more subcutaneous tissues were perceived to be weaker. This variability may affect feedback precision during critical applications, and require a deeper investigation into how this can be overcome. Another issue was encountered during the layout iteration. Because the final layout was relatively sparse, the original connection wires that were used were too short. Therefore, we had to solder and assemble new wires. This, however, helped us make the toolkit adaptable to more diverse future use cases.

6 Discussion

In this section, we discussed the implications drawn from case studies, including strengths and limitations of VibraForge. Additionally, from our first-hand experiences in case studies, we synthesized a generalized five-step pipeline for building spatialized vibrotactile systems. Potential applications of VibraForge were highlighted at the end.

6.1 Strengths of VibraForge

VibraForge aims to provide a "least resistance path" [19] towards creating spatialized vibrotactile systems, which was demonstrated through the three case studies.

VibraForge offers great scalability. Systems designed in the case studies are of different scales, ranging from sleeve-type phonemic display with 24 vibration units on the forearm, to body-scale system with over 100 vibration units spanning across the upper body. Compared existing toolkits that are often constrained intrinsically to less than 16 actuators, VibraForge's chain connection method enables designers to easily scale their system to more than 100 vibration units, significantly expanding the potential design space. Furthermore, adding a new unit to an existing system is also as simple as plug-and-play, without the need for any soldering or programming.

VibraForge offers great expressivity. Information of various complexity are adequately conveyed through spatial vibrotactile feedback in the case studies, ranging from simple on/off status for VR collision detection, to continuous risk levels in teleoperation and symbolic phoneme representation in the phonemic display. Compared with prior spatialized vibrotactile systems that are constrained to using ERMs and low-bandwidth control, our toolkit offers various actuator options and provides fine-grained control over vibration characteristics of each actuator. This enables vibration units to reliably reproduce audio waveforms in technical evaluations.

The GUI Editor of VibraForge features usability. In the phonemic display study, it effectively expedited the pattern authoring process

by reducing the average pattern authoring time to one minute. Compared with prior design interfaces, our editor highlighted the canvas and the timeline that provided substantial support for multi-actuator pattern design.

VibraForge also features portability and flexibility. Compared with the original phonemic display that was wired to a bulky desktop audio device, components used in our prototype were more portable and easy to wear, such as the small control unit that communicates via Bluetooth. In the UAV teleoperation study, flexibility ensured that vibration units could be frequently adjusted during the layout optimization perceptual study. Compared to using commercial devices that often have fixed feedback positions, VibraForge enables designers to easily adjust the positions of vibrations units, thus enabling rapid design iterations.

6.2 Limitations of VibraForge

Although VibraForge can enable designers to quickly prototype and evaluate new multi-actuator vibrotactile devices and experiences, the toolkit does not come without limitations. For example, the open-loop control used in the chain connection topology does pose some issues. First, when a control unit sends a vibration command, it is unclear if the command was successfully transmitted to the target vibration unit or not. This is especially important when sending a "stop" command, because the vibration unit won't stop vibrating if the commands was missed during the transmission process. Second, when multiple actuators were activated, there was a noticeable voltage drop along the chains. Although the technical evaluation showed that the supply voltage remained in the MCU operating range and actuator voltage range, it would still affect the users' subjective experience of the vibrations.

By adding extra command receiver pins on the control unit, one could instead use closed-loop control to solve these problems. With such a design, the vibration units and the control unit would form a loop, wherein the last vibration unit of the chain would be connected back to the control unit using a receiver pin. Whenever a vibration unit receives a command for itself, it would not only process the command and trigger the vibration, but also pass a confirmation message to the next unit. When the last unit in the chain receives this confirmation message, it would send the message back to the central unit, thus closing the loop. In cases where commands fail during the transmission process, the central unit would miss the confirmation message and resend the message. A closed loop connection would also reduce the voltage drop along the chain, as both ends of the chain would be connected to the power source. An LTspice simulation shows that an open-loop, eight-unit chain would result in voltage drop from 5 V to 3.1 V, while a closed-loop version would increase the voltage to 4.2 V.

Additionally, while the GUI Editor's waveform design was advantageous for the phonemic display pattern design, designers would face difficulties when designing more complex spatial patterns. For example, if a designer wanted to create the effect of actuators vibrating from a central point and expanding out in concentric circles, a designer would have to manually design the waveform, assign the waveform to each vibration unit, and specify the start and end time, which is very tedious. A better solution would be to use graphical

design metaphors and breakdown spatial patterns to low-level animations of point, line, and shapes. By leveraging phantom sensation illusions [39], designers would only need to draw the shapes and trajectories, and they would be automatically converted to vibration commands for each unit. In this way, complex patterns could be created with ease.

Our case studies also emphasized the need for an automation pipeline to detect the positions of vibration units on the body and convert them to virtual layout that can be later used in various programming environment. We believe this need can be fulfilled with computer vision and motion capture techniques in the future. Using RGB images of the body, computer vision methods could detect the human body outline, segment vibration units from the body, and record the units' positions. Similarly, retro-reflective markers could be mounted on top of each vibration unit to create a tracking object in the motion capture system. These positions could then be tracked in real-time and loaded into a development environment such as Unity Engine. In this way, designers could focus on the interaction design rather than manually creating the virtual layout.

6.3 Pipeline for Creating Spatialized Vibrotactile Systems

Synthesizing the processes followed during the three case studies, herein we summarize a generalized five-step pipeline for creating spatialized vibrotactile systems.

The first step is to determine **what** information to be delivered through vibrotactile feedback. For example, the purpose of the phonemic display was to convey vowels and consonants to deaf and hard-of-hearing people, so the types of vowels and consonants in words were the target information. In the drone teleoperation scenario, the key information was the potential risk of colliding with nearby obstacles and could be further divided into the directions of obstacles and their risk levels.

The second step is to decide **how** to convey the information. Since vibrations can be configured with different characteristics such as intensity, frequency, duration, position, and so on, the common practice is to breakdown the information and deliver it through multiple channels to maximize information transmission [46]. For example, vowels and consonants for the phonemic displays were distinguished by using moving patterns and static patterns, while different consonants were distinguished by actuator position, frequency modulation and vibration duration. As shown in prior research, providing information through multiple channels with fewer levels in each channel can enhance the overall information transfer for vibrotactile systems.

The third step is to decide **where** to install the system and derive the layout of vibration units. In cases where the information was spatially connected to the surrounding environment, the layout should have a good mapping between tactile positions and spatial direction. For example, in the VR fitness application, vibration units were mainly placed on the arm to provide localized feedback, indicating the arm pose differences between the user and the wall. In the UAV teleoperation, locations of vibrations were perceptually mapped to the directions of incoming obstacle. Perceptual limits should also be considered, e.g., spatial acuity indicated the upper

limits of layouts, as adding more actuators to the layout would not increase the information transmission.

The fourth step is to **build** the system. Control units and vibration units can be chosen based on the scale of the system. Learning from the building process of case studies, We recommend using the small control unit for localized systems such as sleeves and headbands, as well as applications that require short-term usage. Alternatively, the large control unit is suitable for long-term usage and body-scale systems, such as immersive virtual experience and robot teleoperation. When designing vibration patterns, if the information involves complex waveforms, designers can utilize the waveform library in the GUI Editor to expedite the authoring process, just like how we designed patterns for the phonemic display. If the information only requires changing levels of intensity or frequency, the Python server with APIs should be handy.

The final step is to **test** the system and **iterate** on the design. If the desired information cannot be transmitted as expected, the designer can reconfigure the system by changing the layout or experiment different characteristics for conveying the information.

6.4 Potential Applications for VibraForge

A toolkit such as VibraForge could benefit several domains. In VR, spatial vibrotactile feedback can enhance immersive social experiences. In VR social platforms, adding tactile interactions can help users express their emotions through social touch gestures [44]. If developers are creating devices for users with sensory issues in certain areas of their body, they can leverage the toolkit to quickly and easily adjust the layout of actuators to avoid discomfort.

During human-robot interaction, spatial feedback could be useful for assisting humanoid robot teleoperation. Humanoid robot training collects data from human operators performing tasks remotely. However, the absence of tactile feedback mechanisms makes it difficult for operators to execute daily tasks that are typically performed with ease using their own bodies [31]. Spatial tactile feedback can provide additional information about remote interaction, such as when a robot hits a wall or other obstacle. The choice of tactile feedback positions can also be configured to match the positions of sensors on the robot. This feedback could thus increase an operator's situational awareness and close the interaction loop.

Last but not least, we envision that VibraForge can be used to build custom prototypes for tactile perceptual studies, such as those measuring spatial acuity, information transfer, or other perceptual characteristics. For example, Park et al. performed perceptual studies to measure information transfer across five body positions [32]. These positions were not chosen by the researchers but instead were constrained by the design of the commercial system they used (i.e., tactile modules from bHaptics [4]). By using VibraForge, researchers could freely adjust the layout of vibration units on the body and adapt it to any test they want to conduct.

7 Conclusion

Acknowledging the limits with existing spatialized vibrotactile feedback systems and toolkits, we built the VibraForge toolkit. Our toolkit enables scalability via a modular design and the use of chain-connection data transmission methods. Technical evaluations validated the toolkit design, and case studies demonstrated the wide

range of experiences that could be authored. The lessons learned during the case studies contributed to our understanding of a generalized pipeline for creating spatialized vibrotactile systems, and highlighted potential research directions of toolkit refinement and future applications. The toolkit is in its final stages of preparation, and will be released to the public later this year. We are excited to see how researchers and designers will harness the toolkit to elevate their own research and push the boundary of design innovation when using spatialized vibrotactile feedback.

Acknowledgments

References

[1] Marco Aggravi, Florent Pausé, Paolo Robuffo Giordano, and Claudio Pacchierotti. 2018. Design and Evaluation of a Wearable Haptic Device for Skin Stretch, Pressure, and Vibrotactile Stimuli. *IEEE Robotics and Automation Letters* 3, 3 (2018), 2166–2173. https://doi.org/10.1109/LRA.2018.2810887

[2] Aaron D Ames, Samuel Coogan, Magnus Egerstedt, Gennaro Notomista, Koushil Sreenath, and Paulo Tabuada. 2019. Control barrier functions: Theory and applications. In *2019 18th European control conference (ECC)*. IEEE, 3420–3431.

[3] Karlin Bark, Emily Hyman, Frank Tan, Elizabeth Cha, Steven A Jax, Laurel J Buxbaum, and Katherine J Kuchenbecker. 2014. Effects of vibrotactile feedback on human learning of arm motions. *IEEE Transactions on Neural Systems and Rehabilitation Engineering* 23, 1 (2014), 51–63.

[4] bHaptics. 2024. bHaptics: Tactile Feedback for VR, Gaming, and Music. https://www.bhaptics.com/

[5] Adam M Brandt. 2009. Haptic Collision Avoidance for a Remotely Operated Quadrotor UAV in Indoor Environments. (2009).

[6] Seungmoon Choi and Katherine J. Kuchenbecker. 2013. Vibrotactile Display: Perception, Technology, and Applications. *Proc. IEEE* 101, 9 (2013), 2093–2104. https://doi.org/10.1109/JPROC.2012.2221071

[7] Artem Dementyev, Pascal Getreuer, Dimitri Kanevsky, Malcolm Slaney, and Richard F Lyon. 2021. VHP: vibrotactile haptics platform for on-body applications. In *The 34th Annu. ACM Symp. on User Interface Softw. and Technol.* 598–612.

[8] Abdulmotaleb El Saddik, Mauricio Orozco, Mohamad Eid, and Jongeun Cha. 2011. *Haptics technologies: Bringing touch to multimedia*. Springer Science & Business Media.

[9] Hesham Elsayed, Martin Weigel, Florian Müller, Martin Schmitz, Karola Marky, Sebastian Günther, Jan Riemann, and Max Mühlhäuser. 2020. Vibromap: Understanding the spacing of vibrotactile actuators across the body. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies* 4, 4 (2020), 1–16.

[10] Fox. 2008. Hole in the Wall (American game show). https://en.wikipedia.org/wiki/Hole_in_the_Wall_(American_game_show) Accessed: September, 2023.

[11] Xiaolei Hou, Pengfei Fang, and Yaohong Qu. 2017. Dynamic kinesthetic boundary for haptic teleoperation of unicycle type ground mobile robots. In *2017 36th Cine. Control Conf. (CCC)*. IEEE, 6246–6251.

[12] Bingjian Huang, Paul H. Dietz, and Daniel Wigdor. 2024. Investigating the Effects of Intensity and Frequency on Vibrotactile Spatial Acuity. *IEEE Transactions on Haptics* (2024), 1–13. https://doi.org/10.1109/TOH.2024.3350929

[13] Bingjian Huang, Zhecheng Wang, Qilong Cheng, Siyi Ren, Hanfeng Cai, Antonio Alvarez Valdivia, Karthik Mahadevan, and Daniel Wigdor. 2024. AeroHaptix: A Wearable Vibrotactile Feedback System for Enhancing Collision Avoidance in UAV Teleoperation. *arXiv preprint arXiv:2407.12105* (2024).

[14] Novint Technologies Inc. 2024. Novint Falcon. https://hapticshouse.com/pages/novints-falcon-haptic-device. Accessed: Feb 24, 2024.

[15] Ali Israr, Siyan Zhao, Kyna McIntosh, Zachary Schwemler, Adam Fritz, John Mars, Job Bedford, Christian Frisson, Ivan Huerta, Maggie Kosek, Babis Koniaris, and Kenny Mitchell. 2016. Stereohaptics: a haptic interaction toolkit for tangible virtual experiences. In *ACM SIGGRAPH 2016 Studio* (Anaheim, California) *(SIGGRAPH '16)*. Association for Computing Machinery, New York, NY, USA, Article 13, 57 pages. https://doi.org/10.1145/2929484.2970273

[16] Yei Hwan Jung, Jae-Young Yoo, Abraham Vázquez-Guardado, Jae-Hwan Kim, Jin-Tae Kim, Haiwen Luan, Minsu Park, Jaeman Lim, Hee-Sup Shin, Chun-Ju Su, et al. 2022. A wireless haptic interface for programmable patterns of touch across large areas of the skin. *Nature Electronics* 5, 6 (2022), 374–385.

[17] Oliver Beren Kaul and Michael Rohs. 2017. Haptichead: A spherical vibrotactile grid around the head for 3d guidance in virtual and augmented reality. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. 3729–3740.

[18] Odders Lab. 2019. OhShape. https://store.steampowered.com/app/1098100/OhShape/ Accessed: September, 2023.

[19] David Ledo, Steven Houben, Jo Vermeulen, Nicolai Marquardt, Lora Oehlberg, and Saul Greenberg. 2018. Evaluation Strategies for HCI Toolkit Research. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems* (Montreal QC, Canada) *(CHI '18)*. Association for Computing Machinery, New York, NY, USA, 1–17. https://doi.org/10.1145/3173574.3173610

[20] Paul Lemmens, Floris Crompvoets, Dirk Brokken, Jack Van Den Eerenbeemd, and Gert-Jan de Vries. 2009. A body-conforming tactile jacket to enrich movie viewing. In *World Haptics 2009-Third Joint EuroHaptics conference and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*. IEEE, 7–12.

[21] Yan LI, Yuki OBATA, Miyuki KUMAGAI, Marina ISHIKAWA, Moeki OWAKI, Natsuki FUKAMI, and Kiyoshi TOMIMATSU. 2013. A design study for the haptic vest as a navigation system. *International Journal of Asia Digital Art and Design Association* 17, 1 (2013), 10–17.

[22] Robert W. Lindeman, Robert Page, Yasuyuki Yanagida, and John L. Sibert. 2004. Towards full-body haptic feedback: the design and deployment of a spatialized vibrotactile feedback system. In *Proceedings of the ACM Symposium on Virtual Reality Software and Technology* (Hong Kong) *(VRST '04)*. Association for Computing Machinery, New York, NY, USA, 146–149. https://doi.org/10.1145/1077534.1077562

[23] Robert W Lindeman, Yasuyuki Yanagida, Kenichi Hosaka, and Shinji Abe. 2006. The TactaPack: A wireless sensor/actuator package for physical therapy applications. In *2006 14th Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*. IEEE, 337–341.

[24] Cephise Louison, Fabien Ferlay, and Daniel R Mestre. 2017. Spatialized vibrotactile feedback contributes to goal-directed movements in cluttered virtual environments. In *2017 IEEE Symp. on 3D User Interfaces (3DUI)*. IEEE, 99–102.

[25] Jinlong Machinery and Electronics. Co. 2022. LRA - Haptic Motors. http://en.kotl.com.cn/prod_view.aspx?TypeId=75&Id=252&Fid=t3:75:3

[26] Jonatan Martínez, Arturo S García, Miguel Oliver, José P Molina, and Pascual González. 2014. Vitaki: a vibrotactile prototyping toolkit for virtual reality and video games. *International Journal of Human-Computer Interaction* 30, 11 (2014), 855–871.

[27] Inc. Meta Platforms. 2023. Haptics SDK Studio for Meta Quest VR. https://developer.oculus.com/blog/haptics-sdk-studio-meta-quest-vr/ Accessed: 2024-08-26.

[28] Inc Meta Platforms. 2023. Meta Quest 2. https://www.meta.com/ca/quest/products/quest-2/?utm_source=www.google.com&utm_medium=oculusredirect

[29] Kouta Minamizawa, Yasuaki Kakehi, Masashi Nakatani, Soichiro Mihara, and Susumu Tachi. 2012. TECHTILE toolkit: a prototyping tool for design and education of haptic media. In *Proceedings of the 2012 Virtual Reality International Conference*. 1–2.

[30] Sammy Omari, Minh-Duc Hua, Guillaume Ducard, and Tarek Hamel. 2013. Bilateral haptic teleoperation of VTOL UAVs. In *2013 IEEE Int. Conf. on Robot. and Automat.* IEEE, 2393–2399.

[31] Claudio Pacchierotti and Domenico Prattichizzo. 2024. Cutaneous/Tactile Haptic Feedback in Robotic Teleoperation: Motivation, Survey, and Perspectives. *IEEE Transactions on Robotics* 40 (2024), 978–998. https://doi.org/10.1109/TRO.2023.3344027

[32] Jaejun Park, Junwoo Kim, Sangyoon Han, Chaeyong Park, Junseok Park, and Seungmoon Choi. 2023. Information Transfer of Full-Body Vibrotactile Stimuli: An Initial Study with One to Three Sequential Vibrations. In *2023 IEEE World Haptics Conference (WHC)*. IEEE, 41–47.

[33] Evan Pezent, Brandon Cambio, and Marcia K O'Malley. 2020. Syntacts: Opensource software and hardware for audio-controlled haptics. *IEEE Transactions on Haptics* 14, 1 (2020), 225–233.

[34] Helena Pongrac. 2006. Vibrotactile perception: Differential effects of frequency, amplitude, and acceleration. In *2006 ieee international workshop on haptic audio visual environments and their applications (have 2006)*. IEEE, 54–59.

[35] Deepa Prabhu, Muhammad Mehedi Hasan, Lisa Wise, Clare MacMahon, and Chris McCarthy. 2020. VibroSleeve: A wearable vibro-tactile feedback device for arm guidance. In *2020 42nd Annual International Conference of the IEEE Engineering in Medicine & Biology Society (EMBC)*. IEEE, 4909–4912.

[36] Inc. PUI Audio. 2023. HD-VA3222. https://puiaudio.com/product/haptics/hd-va3222 Accessed: September, 2023.

[37] Charlotte M Reed, Hong Z Tan, Zachary D Perez, E Courtenay Wilson, Frederico M Severgnini, Jaehong Jung, Juan S Martinez, Yang Jiao, Ali Israr, Frances Lau, et al. 2018. A phonemic-based tactile display for speech communication. *IEEE transactions on haptics* 12, 1 (2018), 2–17.

[38] Sergio Reyes, Hugo Romero, Sergio Salazar, Rogelio Lozano, and Omar Santos. 2015. Outdoor haptic teleoperation of a hexarotor UAV. In *2015 Int. Conf. on Unmanned Aircr. Syst. (ICUAS)*. IEEE, 972–979.

[39] Oliver S Schneider, Ali Israr, and Karon E MacLean. 2015. Tactile animation by direct manipulation of grid displays. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology*. 21–30.

[40] Oliver S Schneider and Karon E MacLean. 2016. Studying design process and example use with Macaron, a web-based vibrotactile effect editor. In *2016 IEEE Haptics Symposium (Haptics)*. IEEE, 52–58.

[41] Hasti Seifi, Kailun Zhang, and Karon E MacLean. 2015. VibViz: Organizing, visualizing and navigating vibration libraries. In *2015 IEEE World Haptics Conference (WHC)*. IEEE, 254–259.

[42] Shital Shah, Debadeepta Dey, Chris Lovett, and Ashish Kapoor. 2017. AirSim: High-Fidelity Visual and Physical Simulation for Autonomous Vehicles. In *Field and Service Robot*. arXiv:arXiv:1705.05065 https://arxiv.org/abs/1705.05065

[43] Supernatural. 2024. Supernatural - Virtual Reality Fitness. https://www.getsupernatural.com/. Accessed: 2024-09-02.

[44] Philipp Sykownik and Maic Masuch. 2020. The experience of social touch in multi-user virtual reality. In *Proceedings of the 26th ACM Symposium on Virtual Reality Software and Technology*. 1–11.

[45] David Antón Sánchez and René Bohne. 2015. OpenVNAVI: A Vibrotactile Navigation Aid for the Visually Impaired. https://hci.rwth-aachen.de/openvnavi Accessed: September, 2023.

[46] Hong Z Tan, Seungmoon Choi, Frances WY Lau, and Freddy Abnousi. 2020. Methodology for maximizing information transmission of haptic devices: A survey. *Proc. IEEE* 108, 6 (2020), 945–965.

[47] Unity Technologies. 2020. Unity Editor: What's New in 2020.3.31. https://unity.com/releases/editor/whats-new/2020.3.31

[48] Mihail Terenti and Radu-Daniel Vatavu. 2023. VIREO: Web-based Graphical Authoring of Vibrotactile Feedback for Interactions with Mobile and Wearable Devices. *International Journal of Human–Computer Interaction* 39, 20 (2023), 4162–4180.

[49] Ronald T Verrillo, Anthony J Fraioli, and Robert L Smith. 1969. Sensation magnitude of vibrotactile stimuli. *Perception & Psychophysics* 6, 6 (1969), 366–372.

[50] Vicon. 2023. Vicon Shogun Software System. https://www.vicon.com/software/shogun/ Accessed: September, 2023.

[51] Ingrid MLC Vogels. 2004. Detection of temporal delays in visual-haptic interfaces. *Human Factors* 46, 1 (2004), 118–134.

[52] K. Wang and et al. 2021. A Research on Sensing Localization and Orientation of Objects in VR with Facial Vibrotactile Display. In *Virtual, Augmented and Mixed Reality. HCII 2021. Lecture Notes in Computer Science*, J.Y.C. Chen and G. Fragomeni (Eds.), Vol. 12770. Springer, Cham. https://doi.org/10.1007/978-3-030-77599-5_17

[53] Travis J West, Alexandra Bachmayer, Sandeep Bhagwati, Joanna Berzowska, and Marcelo M Wanderley. 2019. The design of the body: suit: score, a full-body vibrotactile musical score. In *Human Interface and the Management of Information. Information in Intelligent Systems: Thematic Area, HIMI 2019, Held as Part of the 21st HCI International Conference, HCII 2019, Orlando, FL, USA, July 26-31, 2019, Proceedings, Part II 21*. Springer, 70–89.

[54] Dennis Wittchen, Katta Spiel, Bruno Fruchard, Donald Degraen, Oliver Schneider, Georg Freitag, and Paul Strohmeier. 2022. Tactjam: An end-to-end prototyping suite for collaborative design of on-body vibrotactile feedback. In *16th Int. Conf. on Tangible, Embedded, and Embodied Interact*. 1–13.

[55] Heidi JB Witteveen, Hans S Rietman, and Peter H Veltink. 2015. Vibrotactile grasping force and hand aperture feedback for myoelectric forearm prosthesis users. *Prosthetics and orthotics international* 39, 3 (2015), 204–212.

[56] Yihong Wu, Lingyun Yu, Jie Xu, Dazhen Deng, Jiachen Wang, Xiao Xie, Hui Zhang, and Yingcai Wu. 2023. AR-Enhanced Workouts: Exploring Visual Cues for At-Home Workout Videos in AR Environment. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*. 1–15.

[57] Pengxiang Xia, Kevin McSweeney, Feng Wen, Zhuoyuan Song, Michael Krieg, Shuai Li, Xiao Yu, Kent Crippen, Jonathan Adams, and Eric Jing Du. 2022. Virtual telepresence for the future of ROV teleoperations: opportunities and challenges. In *SNAME Offshore Symp*. SNAME, D011S001R001.

[58] Dawei Zhang, Guang Yang, and Rebecca P Khurshid. 2020. Haptic teleoperation of uavs through control barrier functions. *IEEE Trans. on Haptics* 13, 1 (2020), 109–115.