Two Sonification Methods for the MindCube

Fangzheng Liu*
fzliu@media.mit.edu
MIT Media Lab
Cambridge, MA, USA

Lancelot Blanchard*
lancelot@media.mit.edu.edu
MIT Media Lab
Cambridge, MA, USA

Don D. Haddad
ddh@mit.edu
MIT Media Lab
Cambridge, MA, USA

Joseph A. Paradiso
joep@media.mit.edu
MIT Media Lab
Cambridge, MA, USA

Abstract

In this work, we explore the musical interface potential of the MindCube, an interactive device designed to study emotions. Embedding diverse sensors and input devices, this interface resembles a fidget cube toy commonly used to help users relieve their stress and anxiety. As such, it is a particularly well-suited controller for musical systems that aim to help with emotion regulation. In this regard, we present two different mappings for the MindCube, with and without AI. With our generative AI mapping, we propose a way to infuse meaning within a latent space and techniques to navigate through it with an external controller. We discuss our results and propose directions for future work.

Keywords

MindCube, music controller, multi-sensor system, generative AI, emotion regulation

1 Introduction and Previous Work

Miniature and handheld music controllers can provide intuitive and expressive means for musical interaction [5, 23], often leveraging innovative designs.

The "Kibo" [1] is a MIDI controller featuring a simplified tangible user interface, designed entirely from wood. It comprises eight geometric extractable solids that users can manipulate to trigger note events and control various musical parameters. The device is an intuitive and tactile learning tool, aiming for enhanced music education. The "Accordiatron" [16] is another novel MIDI controller inspired by the traditional concertina. It translates the performer's gestures into MIDI data, allowing for flexible mapping to various musical parameters. The combination of discrete and continuous sensory outputs provides the subtle expressiveness necessary for interactive music performance. The "AirSticks" [24] is another gestural musical instrument that integrates Inertial Measurement Units (IMUs) to enable performers to trigger and manipulate sound events in real-time through expressive gestures. This wireless device captures both discrete actions, such as striking motions, and continuous movements, allowing for nuanced control over various musical parameters. The "AirSticks" successfully showed the power of commercial IMU for capturing striking and fluid motions in real time. With modern IMUs getting smaller, lower-noise, and lower power consumption, we can design more compact and efficient music controllers without sacrificing the fidelity of the captured motion data. The "CD-synth" [10] is a compact, wireless digital synthesizer designed for expressive musical performance. Its disc-shaped form allows performers to freely rotate and reorient the instrument, utilizing non-contact light sensing to modulate sound parameters. Equipped with sensors that detect rotation, orientation, touch, and proximity, the CD-Synth manipulates audio filters and effects applied to preset wavetables.

Some research uses off-the-shelf interactive systems to create musical interaction. Wong, Yuen & Choy [25], for example, use the Nintendo Wii Controller to develop an interactive music performance system. By employing analytical techniques to study motion data captured by the controller, the system maps detected gestures to musical expression. This approach leverages a low-cost and readily available game controller to create an engaging musical interface. Most of these interfaces, due to the large amount of data they produce through their sensors, are also interesting candidates to control and manipulate Machine Learning (ML) models and generative systems. ML has been extensively used to design music interfaces, mostly through the learning of explicit mappings between controls and sound characteristics [13–15, 18]. To enable further sonic exploration, *latent space exploration* has been proposed as a way to create an implicit or explicit mapping between the internal representation of an ML model and a set of chosen sound characteristics. Previous work [8, 9, 21, 22] has been mostly focused on the unconditional exploration of latent spaces for sound generation. To enable better control, some work has focused on guiding the latent space exploration with a given set of conditioning signals. Bitton et al. [4], for example, enable the sampling of a 3-dimensional latent space learnt by an Adversarial Auto-Encoder (AAE) to generate new musical samples that comply with a set of given characteristics (e.g., timbre or playing technique). Bretan et al. [6] use nearest neighbor search to automatically continue the musical input of a live performer. Only limited work has focused on real-time latent space exploration with strict conditioning for audio generation as we do here.

In this paper, we propose a way for the MindCube, a device we designed in previous work [19], to be used as an interactive music controller. Resembling a fidget cube toy commonly used for stress and anxiety relief, the MindCube offers a more compact form factor compared with the above-mentioned work–only $3.3cm \times 3.3cm \times 3.3cm$, which is significantly smaller than many existing controllers. Its design allows it to be comfortably held and operated with one hand, enhancing its portability and user-friendliness. Despite its small size, the MindCube is equipped with various interactive inputs, including tactile buttons, a rolling disk, a joystick, and a 9 DoF IMU, which can detect the controller's

*Both authors contributed equally to this research.

NIME '25, June 24–27,2025, Canberra, Australia

attitude in hand, providing a rich set of controls for musical expression.

2 Instrument Design

2.1 Hardware

The MindCube design resembles a fidget cube toy commonly used for stress and anxiety relief [2, 3, 11, 12]. A MindCube is a miniature (3.3cm × 3.3cm × 3.3cm) cubic interactive device that is easy to hold with one hand, which makes it ideal for playful interaction. Each side of the MindCube has various interactive inputs, including four tactile buttons, a small rolling disk, and a joystick, as shown in Figure 1.

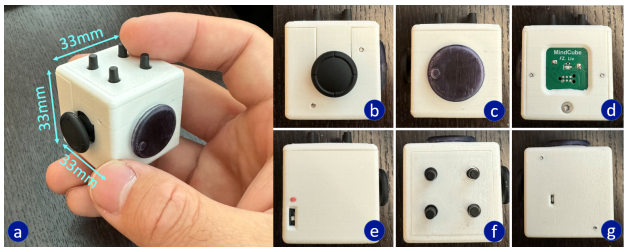

Figure 1: (a) A MindCube in hand and configurations of each side: (b) a joystick, (c) a rolling disk, (d) the charging indicator and programming port, (e) the power switch and linear vibration motor (on the inside), (f) tactile switches, (g) an LED indicator

The rolling disk is connected to a mouse scroll wheel encoder, and the SoC measures the pulses from it to detect rolling distance and directions. The inside of a MindCube is shown in the Figure. 2.

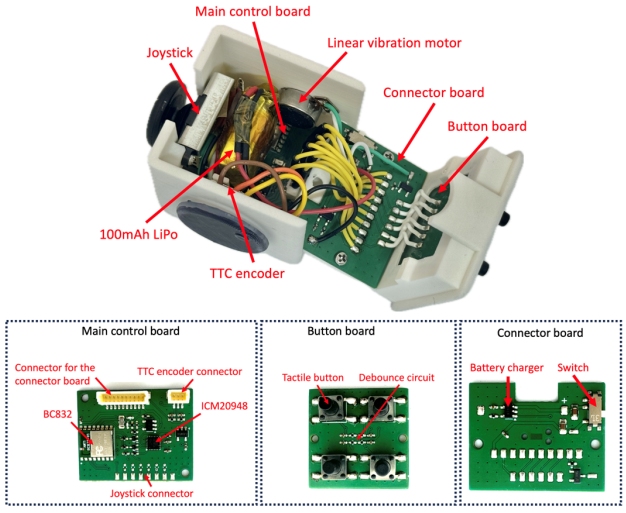

Figure 2: Three PCBs inside a MindCube.

The MindCube contains three PCB boards, each dedicated to a specific function. The main control board manages all control and communication processes. It is equipped with an nRF52832 (ARM Cortex-M4, Nordic) Bluetooth Low Energy (BLE) system-on-chip (SoC). Additionally, the board includes an ICM-20498 9-DoF IMU, which captures 3-axis accelerometer, gyroscope, and magnetometer data. This data enables real-time tracking of the MindCube's orientation while in the user's hand. Mounted on the opposite side of the main control board, the button board features four tactile buttons with debounce circuits to eliminate mechanical switch bouncing. The connector board serves as a bridge between the button board and the main control board. It also integrates a programming port for flashing firmware onto the SoC, a charging port for the Li-Po battery, and a slide switch to turn the MindCube on or off. To prevent accidental power-off, the switch handle is lower than the surface of the MindCube body. The system diagram in Figure 5 details the MindCube system structure.

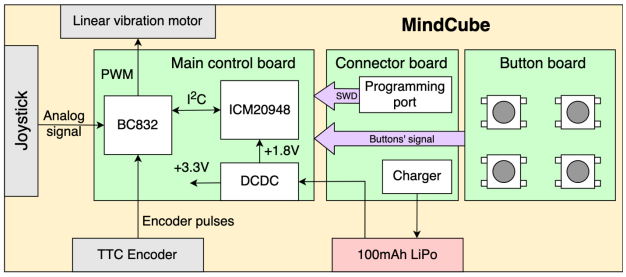

Figure 3: The system diagram of the MindCube.

The MindCube is powered by a 100 mAh Li-Po battery, providing up to more than three hours of battery life during continuous data transmission. Additionally, a linear vibration motor is mounted inside, which can be programmed to deliver various haptic feedback patterns. The motor is controlled via pulse-width modulation (PWM).

2.2 Firmware and communication

The MindCube firmware is developed using the Arduino framework. The SoC continuously reads sensor measurements, packages the data into MindCube packets, and transmits them via Bluetooth Low Energy (BLE) at a rate of 20 Hz. To ensure reliable and unambiguous packet framing, each packet is COBS (Consistent Overhead Byte Stuffing) encoded. A Python-based front-end application running on a MacBook receives the data over BLE, decodes the packets, and processes the information for various applications. One potential use case is analyzing the data to study users' real-time emotional states [26]. In this paper, we explore the data sonification applications.

In the following sections, we describe two musical mapping approaches utilizing the MindCube, with and without generative AI. The AI-driven approach explores the potential of using the MindCube's data to estimate the user's current emotional state and generates music as a proxy for emotion regulation. Although we do not formally prove here that data from the MindCube can detect a user's emotional status, our working hypothesis is that increased interaction with the MindCube may indicate heightened stress levels, while decreased interaction could suggest a more relaxed state. In response, our AI model generates calming music when high activity is detected and stimulating music to engage the user when low activity is detected. We aim to prove the MindCube's potential to accurately detect users' emotions in future work. To contrast with our AI approach, we also offer a non-AI musical mapping, where the MindCube facilitates expressive musical performances through a handcrafted modular synthesizer mapping.

3 AI-generated Music Mapping

In this AI-powered musical mapping, we collect the sensor data and generate loud, high-energy music when the user activity is low, and quiet, low-energy music when the user activity is high. This experimental mapping aims to engage the user continuously and hopes to be able to regulate the user's emotional state over time. An implementation of this mapping can be found on GitHub[1].

3.1 Model Architecture

To create an AI-based musical mapping for the MindCube, we make use of the RAVE (Realtime Audio Variational autoEncoder) model architecture [7]. RAVE is based on a Variational Auto Encoder (VAE) trained on accurately reconstructing audio files by encoding them to a latent distribution, before decoding them into audio files. The appeal of using this architecture within musical instruments stems from its capabilities to perform this autoencoding faster than real time. Following the VAE mathematical notation, we consider music as a continuous signal x sampled from an underlying data distribution $p_{\text{data}}(x)$. The RAVE model allows us to learn a latent representation of dimension 4 ($z \in \mathbb{R}^4$) that captures meaningful musical features while allowing for efficient reconstruction. The encoder and decoder of our VAE are neural networks that are modeled by $q_\phi(z|x)$ and $p_\theta(x|z)$.

The diverse sensors and input devices of the MindCube offer us the opportunity to explore the latent space of the model in a fun and interactive way. In order to explore this latent space in real time, we use Latent Diffusion [20] to generate latent codes that are then passed through the RAVE decoder to reconstruct audio[2]. Specifically, we model the latent space traversal as a stochastic process, where a latent variable $z_T \sim \mathcal{N}(0, I)$ undergoes a sequence of denoising steps following a learned reverse diffusion process. Our denoising process is denoted as $z_{t-1} = z_t + \epsilon_\psi(z_t, t)$, where ϵ_ψ is a neural network trained to predict and remove noise at each step t. The final latent code z_0 is then decoded using $p_\theta(x|z_0)$ to synthesize the corresponding audio. This approach enables smooth and structured navigation of the latent space, allowing the MindCube to generate expressive musical transformations in real time.

3.2 Model Mapping

We then have to generate latent codes that align with the inputs of the musical instrument, in order to create an enjoyable and coherent mapping. To do so, we make use of *Classifier-Free Guidance* (CFG) [17], a technique that allows us to modulate the generation process by conditioning on specific features. In our case, we train the model with CFG using the *Root-Mean-Square* (RMS) value as a conditioning signal, which serves as a proxy for the perceived loudness and energy of the generated audio. We use RMS as our main metric to generate contrasting high-energy and low-energy music, since loud, high-energy music generally exhibits a high RMS while quieter, low-energy music typically has a low RMS. We condition the Latent Diffusion Model with this RMS signal using cross attention layers.

During training, the model uses conditional dropout to learn both an unconditional distribution and a joint distribution over latent variables and their corresponding RMS values. At inference time, we can then use the following conditional score function:

$$\nabla_z \log p_\theta(z|c) = (1-\gamma)\nabla_z \log p_\theta(z) + \gamma \nabla_z \log p_\theta(z|c)$$

where c represents the RMS conditioning and γ is the guidance weight that controls the strength of the conditioning. When inferring latents, we compute a real-time RMS value from the sensor input and normalize it to a range between 0 and 1. This value is then used as the conditioning variable c in the CFG process, guiding the latent code generation towards outputs that match the desired spectral characteristics. The final latent code, as previously described, is passed through the RAVE decoder to synthesize audio, allowing for expressive and dynamic control over the instrument's sonic output.

An overview of our architecture can be seen in Figure 4.

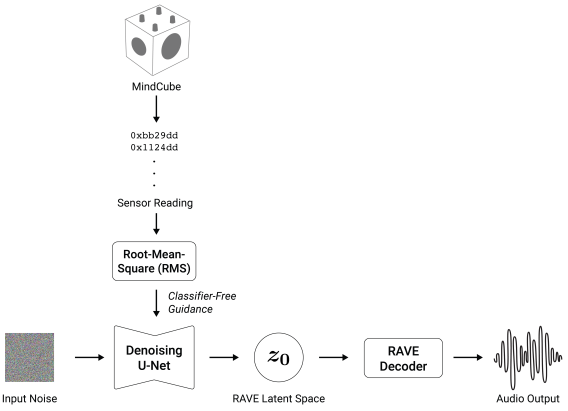

Figure 4: The architecture of the AI Music mapping for the MindCube.

3.3 Training

We train our RAVE model on the Free Music Archive (FMA) dataset, in particular the *"small"* subset, which contains 8,000 tracks of 30s of 8 balanced genres. On this subset of the dataset, the RMS value ranges between 0 and 0.8724. We train the model over 177 epochs until we observe the validation score going up. We use this RAVE model to encode our same dataset into latent codes with length 512 and use these latents to train our Latent Diffusion model. We then train our Latent Diffusion model with RMS as the embedding for CFG, for a total of 700 epochs.

3.4 Real-time Generation

We then need to embed our model into a real-time system to enable continuous music generation. This is a crucial but difficult step since both the latent diffusion and latent decoding processes are lengthy and introduce latency. To reduce this latency, we diffuse latents with a small length of 512, which, at a sampling rate of 44,100 Hz, gives us around 23 seconds of audio. Additionally, we only use 30 diffusion steps. On the M3 Max Macbook Pro that we used in our testing, the latent diffusion step took around 0.90 seconds while the latent decoding step took less than 0.15 seconds, giving us a total of around 1.05 seconds per generation. Our latency of 1.05 seconds for music generation means that we are forced to read the sensor input at a rate lower than $1/1.05 \approx 0.9524$ Hz. Although not optimal, this latency still enables a responsive interface since it allows for the user input to be considered in under a second on average.

[1] https://github.com/mitmedialab/mindcube-rave
[2] RAVE Latent Diffusion is implemented at https://github.com/moiseshorta/RAVE-Latent-Diffusion

Our real-time generation system therefore reads the data from the input sensors every second and adds it into a buffer. Every 1.05 seconds on average, we diffuse a new latent sequence of length 512 and perform Classifier-Free Guidance to condition the diffusion on the last sensor readings. To create an effective RMS condition, we use the following formula:

$$RMS_{cond} = \frac{1}{R} \cdot \sum_{i=1}^{16} w_i \cdot \sigma_i \,,$$

where σ_i refers to the standard deviation of sensor i, w_i is the weight for the reading of that sensor, and R is a normalization factor. In other terms, this calculates the weighted standard deviation of every sensor value over a moving window, normalized to fit between the conditioning values used during training. This allows us to get a sense of the recent *activity* of the MindCube, and calculate an adequate RMS value. Empirically, we observe that the accelerometer, the joystick, the buttons, and the encoder are the best indicators for manual activity. As such, we design a weight vector that favors these sensors over the others.

The latent diffusion model is also not designed to generate continuous music by default. To enable the generated music to flow naturally, we implement *outpainting* to enable the latent diffusion model to generate an adequate continuation for the previous piece of music. For every generation, we use the last few latent codes played to kickstart the diffusion and diffuse only the continuation, which we decode and play. This allows for smooth transitions between every diffused latent.

4 Music Mapping for expressive performance

We also explore another musical mapping by using the sensor data stream from the MindCube with VCV Rack, and this is accomplished through a structured pipeline. This pipeline comprises a Python-based TCP server for real-time data parsing, sensor fusion techniques to process raw IMU data, and the mapping of computed values to control virtual voltages within the modular synthesis environment. The goal is to develop a robust system that enables real-time, motion-driven modulation of synthesis parameters.

The MindCube streams sensor data—including accelerometer, gyroscope, magnetometer, joystick, encoder, and button states—via BLE to a Python TCP server implemented using the `bleak` library. This server listens for incoming byte data from the MindCube, parses it, and transmits the structured sensor data over a TCP socket to a custom-designed VCV Rack module, as shown in Figure 5. To derive meaningful control signals from the raw IMU data, a sensor fusion algorithm computes pitch and roll angles by combining accelerometer and gyroscope readings. The TCP communication follows a local server-client model, ensuring low-latency data transmission. Data packets are formatted as comma-separated values (CSV), containing all sensor readings in a predefined order, facilitating efficient parsing and utilization within the VCV Rack environment.

The custom-designed VCV Rack module, developed in C++, interfaces seamlessly with a Python server to control the synthesizer. Operating within a dedicated thread, it continuously reads the incoming data stream, parses the received CSV strings, and converts them into floating-point values representing sensor readings. This threading approach ensures smooth integration with VCV Rack's real-time processing engine. All sensor data is normalized to fit within the modular synthesis voltage range. Various mapping strategies have been explored. For instance,

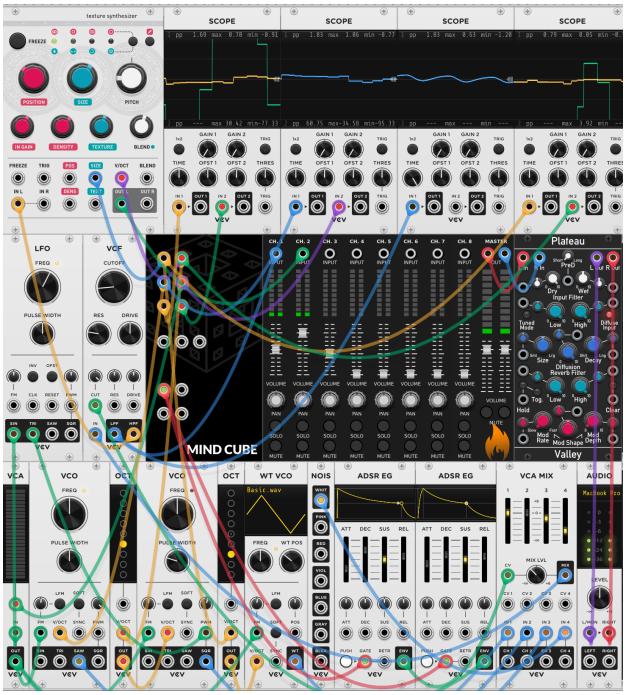

Figure 5: Live VCV Rack Patch that connects to the MindCube custom virtual module.

computed pitch and roll values are assigned to parameters such as filter cutoff frequency and LFO rate. Joystick inputs control stereo panning and modulation index, while button states are converted into gate signals to trigger envelope generators. Additionally, encoder inputs manage step sequencing and parameter selection. This system allows users to control modular synthesis in real time by converting natural movements into dynamic sound parameters.

As AI-driven music tools continue to evolve, integrating them within established modular synthesis environments becomes increasingly relevant. Our interface, built around the BLE-enabled MindCube and mapped into VCV Rack, exemplifies how embodied interaction can extend the patching paradigm. By blending real-time sensor input with the modular workflow, we envision future systems where AI-generated modulation and user-driven control coexist fluidly–bridging algorithmic composition with the hands-on ethos of modular synthesis and patchinwg.

5 Conclusion

In this paper, we explore the data sonification capabilities of the MindCube, a compact, handheld interactive device designed for expressive musical performance. Its small size and various sensing modalities make it a portable and user-friendly tool. Additionally, resembling a traditional fidget cube toy, the MindCube holds a potential for real-time emotion detection. By integrating generative AI-powered real-time music generation, we aim to facilitate emotion regulation, providing different musical responses based on user interaction patterns. In the future, we will utilize the MindCube for user studies, aiming to develop an accurate model that maps its data to various emotional states. This will enhance the ground truth for our generative AI-based musical emotion regulation system. We also aim to explore sonification

methodologies that can use both generative AI and modular synthesis.

6 Ethical Standards

There are no observed conflicts of interest. This research was conducted using discretionary funding for the hardware requirements and used lab-owned compute power for the training of the model. The Free Music Archive dataset used is distributed under the permissive CC BY 4.0 license, allowing us to use it for training and redistribution purposes.

References

[1] Mattia Davide Amico, Luca Andrea Ludovico, et al. 2020. Kibo: A MIDI controller with a tangible user interface for music education. In *Proceedings of the 12th International Conference on Computer Supported Education. 1: CSME*. SCITEPRESS, 613–619.

[2] Kathleen B Aspiranti and David M Hulac. 2022. Using fidget spinners to improve on-task classroom behavior for students with ADHD. *Behavior Analysis in Practice* 15, 2 (2022), 454–465.

[3] Lindsey Biel. 2017. Fidget toys or focus tools. *Autism File* 74 (2017), 12–13.

[4] Adrien Bitton, Philippe Esling, Antoine Caillon, and Martin Fouilleul. 2019. Assisted Sound Sample Generation with Musical Conditioning in Adversarial Auto-Encoders. In *Proceedings of the 22nd International Conference on Digital Audio Effects (DAFx-19),*. Birmingham, UK.

[5] Igor Bonifacic. 2020. Noise Machine is a tiny MIDI controller for creating music on the go. https://www.engadget.com/noise-machine-midi-controller-231326364.html

[6] Mason Bretan, Sageev Oore, Jesse Engel, Douglas Eck, and Larry Heck. 2017. Deep Music: Towards Musical Dialogue. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI'17)*. AAAI Press, 5081–5082. Place: San Francisco, California, USA.

[7] Antoine Caillon and Philippe Esling. 2021. RAVE: A variational autoencoder for fast and high-quality neural audio synthesis. arXiv:2111.05011 [cs.LG] https://arxiv.org/abs/2111.05011

[8] Guilherme Campos, Nuno Fonseca, Anibal Ferreira, and Matthew Davies. 2018. Generative Timbre Spaces: Regularizing Variational Auto-Encoders with perceptual Metrics. In *Proceedings of the 21st International Conference on Digital Audio Effects (DAFx-18),*. Aveiro, Portugal.

[9] Nutan Chen, Djalel Benbouzid, Francesco Ferroni, Mathis Nitschke, Luciano Pinna, and Patrick van der Smagt. 2022. Flat Latent Manifolds for Human-machine Co-creation of Music. https://arxiv.org/abs/2202.12243 _eprint: 2202.12243.

[10] Patrick Chwalek and Joe A Paradiso. 2019. CD-Synth: a Rotating, Untethered, Digital Synthesizer.. In *NIME*. 371–374.

[11] Suzanne B da Câmara, Rakshit Agrawal, and Katherine Isbister. 2018. Identifying children's fidget object preferences: toward exploring the impacts of fidgeting and fidget-friendly tangibles. In *Proceedings of the 2018 Designing Interactive Systems Conference*. 301–311.

[12] Matson Driesen, Joske Rijmen, An-Katrien Hulsbosch, Marina Danckaerts, Jan R Wiersema, and Saskia Van der Oord. 2023. Tools or Toys? The Effect of Fidget Spinners and Bouncy Bands on the Academic Performance in Children With Varying ADHD-Symptomatology. *Contemporary Educational Psychology* 75 (2023), 102214.

[13] Rebecca Fiebrink, Dan Trueman, and Perry R. Cook. 2009. A Meta-Instrument for Interactive, On-the-Fly Machine Learning. In *New Interfaces for Musical Expression*. https://api.semanticscholar.org/CorpusID:9059668

[14] Jules Françoise. 2013. Gesture–sound mapping by demonstration in interactive music systems. In *Proceedings of the 21st ACM International Conference on Multimedia* (Barcelona, Spain) *(MM '13)*. Association for Computing Machinery, New York, NY, USA, 1051–1054. https://doi.org/10.1145/2502081.2502214

[15] Ohad Fried and Rebecca Fiebrink. 2013. Cross-modal Sound Mapping Using Deep Learning. In *Proceedings of the International Conference on New Interfaces for Musical Expression*. Graduate School of Culture Technology, KAIST, Daejeon, Republic of Korea, 531–534. https://doi.org/10.5281/zenodo.1178528

[16] Michael Gurevich and Stephan von Muehlen. 2020. The Accordiatron: A MIDI controller for interactive music. *arXiv preprint arXiv:2010.01574* (2020).

[17] Jonathan Ho and Tim Salimans. 2021. Classifier-Free Diffusion Guidance. In *NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications*. https://openreview.net/forum?id=qw8AKxfYbI

[18] Cheng-Zhi Anna Huang, David Duvenaud, Kenneth C. Arnold, Brenton Partridge, Josiah W. Oberholtzer, and Krzysztof Z. Gajos. 2014. Active learning of intuitive control knobs for synthesizers using gaussian processes. In *Proceedings of the 19th International Conference on Intelligent User Interfaces* (Haifa, Israel) *(IUI '14)*. Association for Computing Machinery, New York, NY, USA, 115–124. https://doi.org/10.1145/2557500.2557544

[19] Fangzheng Liu, Don Derek Haddad, and Joe Paradiso. 2024. MindCube: an Interactive Device for Gauging Emotions. In *Adjunct Proceedings of the 37th Annual ACM Symposium on User Interface Software and Technology*. 1–2.

[20] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2022. High-Resolution Image Synthesis With Latent Diffusion Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 10684–10695.

[21] Hugo Scurto and Ludmila Postel. 2023. Soundwalking Deep Latent Spaces. In *Proceedings of the International Conference on New Interfaces for Musical Expression*, Miguel Ortiz and Adnan Marquez-Borbon (Eds.). Mexico City, Mexico, 232–235. https://doi.org/10.5281/zenodo.11189166 ISSN: 2220-4806.

[22] Victor Shepardson and Thor Magnusson. 2023. The Living Looper: Rethinking the Musical Loop as a Machine Action-Perception Loop. In *Proceedings of the International Conference on New Interfaces for Musical Expression*, Miguel Ortiz and Adnan Marquez-Borbon (Eds.). Mexico City, Mexico, Article 32, 8 pages. https://doi.org/10.5281/zenodo.11189164

[23] Giuseppe Torre, Kristina Andersen, and Frank Baldé. 2016. The Hands: The making of a digital musical instrument. *Computer Music Journal* 40, 2 (2016), 22–34.

[24] Sam Trolland, Alon Ilsar, Ciaran Frame, Jon McCormack, and Elliott Wilson. 2022. AirSticks 2.0: Instrument design for expressive gestural interaction. In *NIME 2022*. PubPub.

[25] Elaine L Wong, Wilson YF Yuen, and Clifford ST Choy. 2008. Designing wii controller: a powerful musical instrument in an interactive music performance system. In *Proceedings of the 6th International Conference on Advances in Mobile Computing and Multimedia*. 82–87.

[26] Kieran Woodward and Eiman Kanjo. 2020. ifidgetcube: Tangible fidgeting interfaces (tfis) to monitor and improve mental wellbeing. *IEEE Sensors Journal* 21, 13 (2020), 14300–14307.