Waveform Autoencoding at the Edge of Perceivable Latency

Franco Caspe
Centre for Digital Music,
Queen Mary University of London
London, United Kingdom
f.s.caspe@qmul.ac.uk

Andrew McPherson
Dyson School of Design Engineering,
Imperial College London
London, United Kingdom
andrew.mcpherson@imperial.ac.uk

Mark Sandler
Centre for Digital Music,
Queen Mary University of London
London, United Kingdom
mark.sandler@qmul.ac.uk

Abstract

We introduce an audio plugin implementation of BRAVE, a waveform autoencoder presented recently, that affords Neural Audio Synthesis with low latency and jitter. As a redesign of the well-known RAVE model, BRAVE introduces a series of architectural modifications for supporting instrumental interaction with almost imperceptible latency (<10 ms) and jitter (~3 ms). By comparing both designs, we highlight key architectural differences between the models that impact their instrumental performance capability, arguing that no model fits all purposes, and calling for their careful selection for each interactive design. Finally, we discuss challenges and opportunities for leveraging low-latency waveform autoencoders to develop interactive systems, such as Digital Musical Instruments, that can foster control intimacy through enhanced responsiveness and space for nuance.

CCS Concepts

• **Applied computing** → **Sound and music computing**.

Keywords

Instrumental Performance, Timbre Transfer, Low Latency, Neural Audio Synthesis

ACM Reference Format:
Franco Caspe, Andrew McPherson, and Mark Sandler. 2025. Waveform Autoencoding at the Edge of Perceivable Latency. In *Proceedings of International Conference on New Interfaces for Musical Expression (NIME '25).* ACM, New York, NY, USA, 4 pages.

1 Introduction

Neural Audio Synthesis (NAS), a technique that uses neural networks to create data-driven synthesizers that can learn from audio corpora, has seen widespread adoption in the Digital Musical Instrument (DMI) design community. Among available algorithms, the RAVE models [2], seem to be a major choice for integration in interactive systems, with extensive applications that explore embodiment [15, 21], entanglement [25], agency [23] and physical interfaces [19], to name a few.

However, this raises a question: why do such varied applications converge on a single NAS algorithm? [1] We believe this may be due to its wide integration in audio production and creative coding platforms such as audio plugins, Max MSP, Pure Data or SuperCollider [3, 22], and also due to its easy training interface, which suggests an understandable path of least resistance [17] for DMI designers. At the same time, the reasons given by the authors revolve around the idea of training efficiency, the presence of latent spaces, and real-time inference. RAVE is not the only algorithm with such characteristics [9, 10, 26]: we argue that a deeper understanding of a few key technical characteristics of NAS, which we hereby explain within the context of RAVE v1, can help designers make more informed decisions on the most suitable algorithm for their specific use case.

For instance, a notable drawback of RAVE models is their latency in the order of hundreds of milliseconds [3, 6, 28], which makes it impractical for instrumental performance applications that require low action-to-sound latency and timing stability, crucial in DMIs for supporting control intimacy, time-keeping, and developing performance skills and personal style [14]. Addressing this, BRAVE [6], a recent re-design of RAVE v1, features suitable latency (< 10ms) and jitter (~3ms) [13], while preserving key characteristics—signal autoencoding (a.k.a. timbre transfer [2]) and latent space manipulation.

This paper introduces BRAVE to DMI designers by comparing it to the widely used RAVE v1 model. Leveraging this, we highlight important characteristics of waveform autoencoders that impact instrumental performance but are often invisible to DMI designers, buried in the complicated specifications of the algorithms. We then introduce an audio plugin [2] for real-time timbre transfer with BRAVE and conclude with insights on the challenges and possibilities for designing DMIs with audio autoencoders that operate near the limit of perceivable latency.

2 Technical Details

2.1 Real-time Inference and Latency

Instrumental performance typically requires low-latency algorithms that process short audio blocks at a high rate. This minimizes the waiting time between input and output, i.e., the buffering latency, but also limits the available processing time for a single inference pass [14]. Algorithms, including neural nets, are assessed for real-time operation by using the Real Time Factor (RTF), defined as the ratio between the time processing time t_p and the length of the input t_i, $RTF = t_p/t_i$. Therefore, a NAS model requires a $RTF < 1$ for a specific block size to be processed in real time.

However, even when using short input lengths, there may be additional sources of latency [6] related to other factors such as internal delay lines or slow dynamic responses. In the next section, we analyze how BRAVE tackles the latency sources found in RAVE.

[1] Although there are two main versions, with different architecture and slightly different audio quality, we consider them both inceptions of a single algorithm as they have similar capabilities in terms of latency, and learning efficiency.

[2] Supplemental material and download information can be found at http://fcaspe.github.io/BravePlugin.

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

2.2 Architectural Differences

BRAVE is a redesign of RAVE[3], aimed to support low-latency instrumental interaction. In both models, timbre transfer is accomplished by feeding an audio signal to a compressing encoder, which generates a latent vector of a shorter temporal length, but with a higher number of channels than the input. This vector is then processed by the decoder, which decompresses it, generating an audio signal back, with spectral and dynamic characteristics similar to that of the dataset used to train the model. Both models feature a similar variational, convolutional autoencoding architecture, modified in BRAVE to reduce latency. Figure 1 shows a simplified diagram of the main architectural differences. We explain them as follows:

Causal Training: Any algorithm that operates within a real-time constraint has to be causal. RAVE is typically trained in a non-causal fashion, with a look-ahead of about half a second. For real-time inference, the model is reconfigured as causal by adding delay lines within the convolutional layers which increases latency [3]. Conversely, BRAVE is always trained causally and does not require delay lines.

Lower Compression Ratio: The compression ratio determines how many audio samples are computed into a single latent timestep. RAVE, for instance, compresses 2048 audio samples into a single latent vector. A real-time scenario requires buffering at least that amount of samples before running a forward pass on the encoder. BRAVE reduces buffering latency with a smaller compression ratio of 128 samples. Furthermore, as events in the input are compressed with much finer granularity, the jitter of the response is improved.

Reduced Multi-Band Filter Length: RAVE employs a multi-band decomposition and re-composition Finite Impulse Response (FIR) filters for input and output respectively [16], with an inter-band attenuation of 100 dB. Relaxing this constraint down to 40 dB yields shorter filters, which reduces their group delay from about 512 to 128 samples, at the expense of a slight reduction in audio quality.

Removed Noise Generator: This generator in RAVE processes a noise signal using a time-variant filter controlled by the model, using FFT windows of 1024 samples. In BRAVE this would determine the minimum buffering latency for output. Therefore we do not implement it, as it just "slightly increases the reconstruction naturalness of noisy signals" [2, p.6].

Reduced number of parameters: BRAVE halves the channel width of all convolution layers, which yields a total number of parameters of 4.9 M, in comparison to those of RAVE's 17.6 M. Reducing the size of the layers does not inherently modify latency, but improves its RTF considerably in consumer CPUs, especially at short windows of 128 samples.

Compensated Receptive Field: Modifying the compression ratio can drastically reduce the receptive field of the model, i.e., the memory that stores the temporal context of the model, which guarantees continuity of the signal across audio blocks. BRAVE compensates for this by increasing the decoder's receptive field. This alters the dynamic properties of its latent space, as we discuss in Section 3.

BRAVE modifications effectively yield a causal waveform autoencoder with audio quality and timbre transfer capabilities similar to RAVE, but with better content preservation in terms of pitch and onsets due to the increased temporal resolution of its smaller compression ratio. Moreover, BRAVE can perform forward passes in a real-time, block-based fashion, with a latency of around 10 ms ± 3 ms when run at a sample rate of 44.1 kHz. Small latency variations may depend on the training data. We refer readers to the original BRAVE model publication for a thorough analysis of its audio quality [6] and a comparison with that of RAVE.

2.3 Real-time Implementation

Our model is compatible with existing tools that support RAVE in audio production and creative coding, which are based on the TorchScript runtime[4]. However, as stated in the original publication [6], BRAVE shows an RTF > 1 when running at the condition of minimum latency (128 samples) because it requires allocating memory at each inference pass. To address this, we develop a custom C++ inference engine for our model using RTNeural [7], a library for real-time DNN inference in audio applications. RTNeural supports causal, block-based inference on its convolutional layers, and allocates memory only once at model instantiation, which makes it suitable for inference at high frame rates. This allows BRAVE to run at an RTF of 0.3 on an Apple M1 Pro CPU.

We deploy this engine in a JUCE plugin and train on different open datasets of percussive and melodic instruments [4, 11]. The plugin features a simple set of controls for model selection, gain adjustment, and dry/wet controls, for mixing the transformed audio with the source. Its current GUI is shown in Figure 2.

3 Discussion: Interaction possibilities with low-latency waveform autoencoders

We believe our BRAVE implementation appears as a new design primitive in a field historically occupied by traditional low-latency Music Information Retrieval (MIR) algorithms for music performance analysis such as fundamental frequency (f_0) trackers [8] and onset detectors [27], which have been employed in a variety of music interactive systems with and without neural networks [5, 13, 18, 24].

By performing live timbre transfer with voice, playing guitar, or percussion, we find that BRAVE can render definite and ambiguous sound characteristics of the learned instruments, including ambiguous pitch and onsets. Keeping a degree of ambiguity is crucial for fostering intimate control relationships between musicians and instruments, where there is space for meaning-making [12]. Furthermore, ambiguity is an inherent characteristic of instrumental performance, and a source of headaches when working with f_0 trackers and onset detectors: consider the ancillary sounds made by hands sliding over the fretboard of a guitar, or subtle touch of a percussion instrument, or the bowing action on a violin that elicits high resonances. All these sounds are ambiguous for a reductionist frame of fundamental frequencies and amplitudes, and thus, when we execute these actions over such technical systems, we may perceive missed notes (actions we expect to generate sound but they do not), ghost notes (sounds that we do not execute voluntarily), or simply wrong pitch.

This is not to say that just by using a waveform autoencoder we are free from the representational problems that come with traditional MIR. Autoencoders learn a hidden representation from training data that, for our cases, generally aligns well with our expectations for instrumental performance. This however is not

[3]To improve readability, for the rest of the document we refer to the model RAVE v1, simply as RAVE.

[4]https://pytorch.org/docs/stable/jit.html

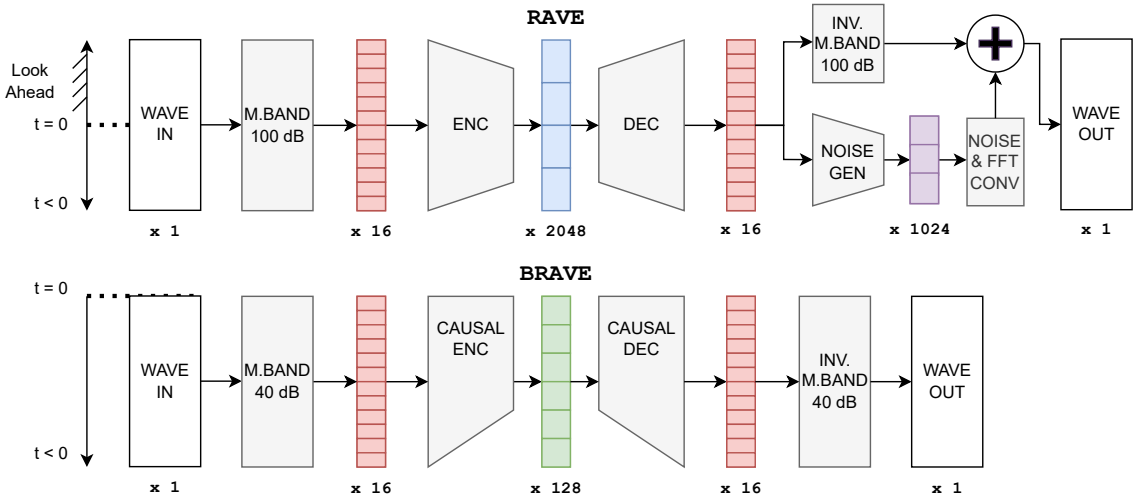

Figure 1: BRAVE achieves adequate latency (< 10 ms) and jitter (~ 3 ms) by removing RAVE's noise generator and using a smaller compression ratio, multiband attenuation, and causal training. The number of parameters is also reduced to improve its RTF. The compression ratios at different stages of the models are denoted in `monospace`.

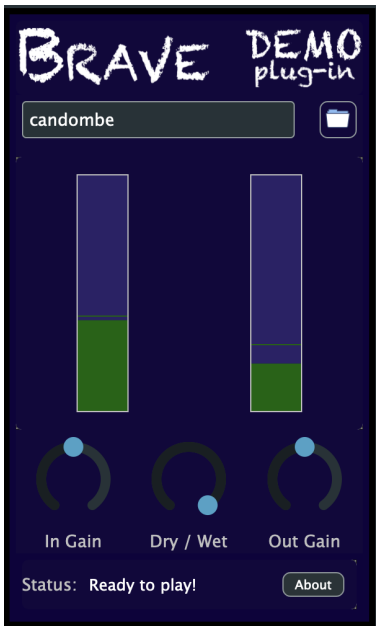

Figure 2: Plugin's GUI.

guaranteed: in agential realism terms, all representations entail an act of exclusion [1], and may be met with uncertainty or ambiguity by listeners and players even when designers employ them to stabilize their music performance systems [20]. In this case, a clear trade-off when using BRAVE is the loss of interoperability that classic representations of pitch and onsets offer, which for instance, could be used for driving a wide variety of synthesizers.

Finally, for the case of DMI designers considering latent representations and their manipulation, our model illustrates our argument against using RAVE as a one-size-fits-all tool. Not all latent spaces behave the same way: in RAVE, much of the dynamic information is encoded within each latent vector, resulting from the encoder's high compression ratio and receptive field. As a result, we observe that DMIs manipulating RAVE's latent space tend to produce sound dynamics tightly coupled to those of the controller input [25]. In contrast, due to the increase in the decoder's receptive field, BRAVE relies more on the *temporal trajectory* of a sequence of latent vectors for rendering the learned dynamics. This changes the properties of the latent space which could show different transient responses and more decoupled controller/sound dynamics if manipulated.

4 Conclusion

We hope this short paper can shed light on some of the crucial architectural characteristics that condition the instrumental performance capabilities of NAS algorithms when implemented in DMIs, arguing that there is no one-size-fits-all model. We emphasize that, in the rich landscape of NAS algorithms with potential for instrumental interaction, RAVE and BRAVE are just two options among many others. However, we expect our low-latency implementation of BRAVE can inspire researchers and makers to explore possibilities for novel and responsive musical interfaces that foster intimate control through its ambiguity-rendering possibilities and low-latency interaction.

5 Ethical Standards

This work did not involve experiments with musicians other than the first author, hence no institutional ethics approval was required. However, like all Artificial Intelligence research, our results and examples may reproduce and obfuscate biases in the choice of model architecture or training data. To foster transparency, we provide the training code and the audio plugin for download with models trained on open datasets. For more information please visit http://fcaspe.github.io/BravePlugin.

Acknowledgments

This work is supported by the Centre for Doctoral Training in Artificial Intelligence and Music, Engineering and Physical Sciences Research Council, UK Research and Innovation (EP/S022694/1). AM's contributions are supported by a UKRI Frontier Research

(Consolidator) Grant (EP/X023478/1, "RUDIMENTS") and by the Royal Academy of Engineering under the Research Chairs and Senior Research Fellowships scheme.

References

[1] Karen Barad. 2003. Posthumanist Performativity: Toward an Understanding of How Matter Comes to Matter. *Signs* 28, 3 (2003), 801–831. https://doi.org/10.1086/345321 jstor:10.1086/345321

[2] Antoine Caillon and Philippe Esling. 2021. RAVE: A Variational Autoencoder for Fast and High-Quality Neural Audio Synthesis. arXiv. arXiv:2111.05011 [cs, eess]

[3] Antoine Caillon and Philippe Esling. 2022. Streamable Neural Audio Synthesis With Non-Causal Convolutions. arXiv. arXiv:2204.07064 [cs, eess, stat]

[4] Lee Callender, Curtis Hawthorne, and Jesse Engel. 2020. Improving Perceptual Quality of Drum Transcription with the Expanded Groove MIDI Dataset. arXiv:2004.00188 [cs]

[5] Franco Caspe, Andrew McPherson, and Mark Sandler. 2023. FM Tone Transfer with Envelope Learning. In *Proceedings of the 18th International Audio Mostly Conference (AM '23)*. Association for Computing Machinery, New York, NY, USA, 116–123. https://doi.org/10.1145/3616195.3616196

[6] Franco Caspe, Jordie Shier, Mark Sandler, Charalampos Saitis, and Andrew McPherson. 2025. Designing Neural Synthesizers for Low-Latency Interaction. *Journal of the Audio Engineering Society* (In Press) (2025). https://doi.org/10.48550/arXiv.2503.11562

[7] Jatin Chowdhury. 2021. RTNeural: Fast Neural Inferencing for Real-Time Systems. https://doi.org/10.48550/arXiv.2106.03037 arXiv:2106.03037 [eess]

[8] Alain de Cheveigné and Hideki Kawahara. 2002. YIN, a Fundamental Frequency Estimator for Speech and Music. *The Journal of the Acoustical Society of America* 111, 4 (April 2002), 1917–1930. https://doi.org/10.1121/1.1458024

[9] Jesse Engel, Kumar Krishna Agrawal, Shuo Chen, Ishaan Gulrajani, Chris Donahue, and Adam Roberts. 2018. GANSynth: Adversarial Neural Audio Synthesis. In *International Conference on Learning Representations*.

[10] Jesse Engel, Lamtharn Hantrakul, Chenjie Gu, and Adam Roberts. 2020. DDSP: Differentiable Digital Signal Processing. In *8th International Conference on Learning Representations*. ICLR, Addis Ababa, Ethiopia.

[11] Dave Foster and Simon Dixon. 2021. Filosax: A Dataset of Annotated Jazz Saxophone Recordings. In *Proceedings of the 22nd International Society for Music Information Retrieval Conference*. ISMIR, Online.

[12] William W. Gaver, Jacob Beaver, and Steve Benford. 2003. Ambiguity as a Resource for Design. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '03)*. Association for Computing Machinery, New York, NY, USA, 233–240. https://doi.org/10.1145/642611.642653

[13] Robert H. Jack, Tony Stockman, and Andrew McPherson. 2016. Effect of Latency on Performer Interaction and Subjective Quality Assessment of a Digital Musical Instrument. In *Proceedings of the Audio Mostly 2016*. ACM, Norrköping Sweden, 116–123. https://doi.org/10.1145/2986416.2986428

[14] A. P. McPherson, R. H. Jack, G. Moro, and Brisbane Proceedings of the International Conference on New Interfaces for Musical Expression. 2016. Action-Sound Latency: Are Our Tools Fast Enough?. In *Proceedings of NIME'16*. NIME, Brisbane, Australia.

[15] Sarah Nabi, Philippe Esling, Geoffroy Peeters, and Frédéric Bevilacqua. 2024. Embodied Exploration of Deep Latent Spaces in Interactive Dance-Music Performance. In *Proceedings of the 9th International Conference on Movement and Computing*. ACM, Utrecht Netherlands, 1–9. https://doi.org/10.1145/3658852.3659072

[16] T.Q. Nguyen. 1994. Near-Perfect-Reconstruction Pseudo-QMF Banks. *IEEE Transactions on Signal Processing* 42, 1 (Jan. 1994), 65–76. https://doi.org/10.1109/78.258122

[17] Andrew Pickering. 1993. The Mangle of Practice: Agency and Emergence in the Sociology of Science. *Amer. J. Sociology* 99, 3 (Nov. 1993), 559–589. https://doi.org/10.1086/230316

[18] Cornelius Pöpel and Roger Dannenberg. 2005. Audio Signal Driven Sound Synthesis. In *Proceedings of the International Computer Music Conference*. Michigan Publishing, Barcelona, Spain.

[19] Nicola Privato, Giacomo Lepri, Thor Magnusson, and Einar Torfi Einarsson. 2024. Sketching Magnetic Interactions for Neural Synthesis. In *Proceedings of TENOR 2024*.

[20] Courtney N. Reed, Adan L. Benito, Franco Caspe, and Andrew P. McPherson. 2024. Shifting Ambiguity, Collapsing Indeterminacy: Designing with Data as Baradian Apparatus. *ACM Trans. Comput.-Hum. Interact.* 31, 6 (Dec. 2024), 73:1–73:41. https://doi.org/10.1145/3689043

[21] Hugo Scurto and Ludmila Postel. 2023. Soundwalking Deep Latent Spaces. In *Proceedings of the 23rd International Conference on New Interfaces for Musical Expression (NIME'23)*. Mexico City, Mexico.

[22] Nicholas Shaheed and Ge Wang. 2024. I Am Sitting in a (Latent) Room. In *Proceedings of the International Conference on New Interfaces for Musical Expression*. 333–338. https://doi.org/10.5281/zenodo.13904872

[23] 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*. 224–231. https://doi.org/10.5281/zenodo.11189164

[24] Jordie Shier, Charalampos Saitis, Andrew Robertson, and Andrew McPherson. 2024. Real-Time Timbre Remapping with Differentiable DSP. In *Proceedings of NIME 2024*. NIME, Utrecht, Netherlands.

[25] Steve Symons. 2024. The Perceptron: A Multi-player Entangled Instrument Based on Interpretive Mapping and Intra-action.. In *Audio Mostly 2024 - Explorations in Sonic Cultures*. ACM, Milan Italy, 385–391. https://doi.org/10.1145/3678299.3678338

[26] K. Tatar, D. Bisig, and P. Pasquier. 2021. Introducing Latent Timbre Synthesis. *Neural Computing and Applications* 33, 1 (Jan. 2021), 67–84. https://doi.org/10.1007/s00521-020-05424-2 arXiv:2006.00408

[27] Pierre Alexandre Tremblay, Owen Green, Gerard Roma, James Bradbury, Ted Moore, Jacob Hart, and Alex Harker. 2022. Fluid Corpus Manipulation Toolbox. (July 2022). https://doi.org/10.5281/zenodo.6834643

[28] Gabriel Vigliensoni and Rebecca Fiebrink. 2023. Steering Latent Audio Models through Interactive Machine Learning. In *Proceedings of the 14th International Conference on Computational Creativity (ICCC'23)*. Zenodo. https://doi.org/10.5281/zenodo.8087978