PolyDDSP: A Lightweight and Polyphonic Differentiable Digital Signal Processing Library

Tom Baker, Ricardo Climent, and Ke Chen

University of Manchester
{tom.baker,ricardo.climent,ke.chen}@manchester.ac.uk

Abstract. This paper presents a work-in-progress DSP architecture[1] building from the basis of the Differentiable Digital Signal Processing (DDSP) library by Engel et al. (2020). The architecture is designed to process polyphonic musical audio in real-time, making use of classical DSP methods for greater interpretability. Utilising recent advancements in lightweight polyphonic pitch detection models, multiple input audio streams can be processed simultaneously, and with a novel stochastic latent dimension, the model can generate novel audio timbres outside of the training dataset. Due to its lightweight nature, the proposed architecture is designed to be used for live audio transformations with minimal input latency. The paper also discusses the limitations of the existing state-of-the-art model, which is deterministic and restricted to monophonic processing. Throughout, the paper explores potential applications of the proposed model. These include not only versatile timbre transfer between distinct instruments but interpolation between timbres, resulting in the creation of new sounds that can expand the aural pallet of musicians, sound designers, and experimental composers using live electronics. Furthermore, the model extends the library's toolkit, such as natural pitch shifting and room acoustic reverb modelling to previously unusable polyphonic inputs.

Keywords: Digital Signal Processing, Machine Learning, Real-time, Polyphony, Timbre Transfer

1 Introduction

Digital signal processing (DSP) refers to the utilisation of algorithms and methodologies to process and analyse signals, including but not limited to audio and video. The use of DSP is a cornerstone in creative expression for the digital artist, using technology to explore sounds otherwise not possible acoustically. These techniques are often developed based on a solid foundation of knowledge and theory, enabling the creation of processes that can effectively extract desirable features or reduce unwanted noise,

[1] Code and audio examples at `https://github.com/TeeJayBaker/PolyDDSP`

among other applications. This theoretical basis enables the creation of efficient models that can generalise well over the relevant domain. This approach is also reflected in the use of structural priors within neural networks, such as convolution and recurrence, which are designed to take advantage of underlying patterns and relationships in the data being processed.

For Musical audio, we can exploit many structures straight from music theory and spectral analysis. Given its harmonic nature, we can approximate music as a combination of two components: harmonic - a series of sinusoids with integer multiples of a fundamental frequency, and noise - any remaining elements not so clearly defined in the frequency domain. We can leverage this decomposition to construct a decoder using classical synthesis techniques, resulting in lightweight, robust, and expressive audio generation. While the DDSP [1] library's current state-of-the-art (SOTA) work offers an excellent implementation of these techniques, it has a significant limitation: it cannot handle polyphonic audio.

In music, polyphony denotes the act of playing or singing multiple distinct notes at the same time. While some instruments, such as most woodwind instruments, are monophonic and can generally only play one note at a time[2], many others are polyphonic and rely on playing multiple notes at once for creative expression. The CREPE [2] pitch encoder is a critical component for gathering pitch information within the DDSP library, and while CREPE is both lightweight and state-of-the-art for pitch accuracy, it is limited by its monophonic nature. Consequently, the DDSP library is only capable of reproducing monophonic audio signals, which restricts its applicability to polyphonic musical audio.

In this paper, we introduce the PolyDDSP model, which combines the modular and classical techniques from the DDSP architecture with state-of-the-art polyphonic pitch detection models. This new architecture is designed to handle polyphonic audio while maintaining lightweight performance and modular interpretability through the incorporation of multiple audio channels throughout the model. In addition, incorporating a stochastic latent dimension that closely resembles that of a traditional VAE will enable a more organic variation in the generated sounds, including the ability to interpolate between various timbres that have been learned. This allows the creation of new novel hybrid instrument sounds, broadening the possibilities of musical expression in the digital studio. The lightweight design of the proposed model also creates the opportunity for the development of a real-time audio plugin, similar to DDSP-VST, which can be used for live audio transformations within a digital audio workstation powered by machine learning.

The main feature of the DDSP toolkit is its timbre transfer capability, creating a unique tool for the digital studio to surpass the limitations of acoustic instruments by facilitating novel routes for real-time timbral hybridisation. However, its capabilities extend far beyond this. Through the complete reconstruction of audio from fundamental elements, the toolkit can accomplish tasks such as transposing audio while maintaining accurate instrument timbre, modifying performance dynamics, and even manipulating reverb characteristics, including complete dereverberation. The generalisation work

[2] They are capable of polyphonic expression using contempory extended techniques such as multiphonics.

presented in this paper extends the applicability of this toolkit to polyphonic audio, significantly expanding the potential usage by accommodating audio inputs with multiple simultaneous notes.

2 Related Work

Transcription: Automatic Music Transcription (AMT) has been a long-standing problem in music, with Klapuri et al.'s probabilistic model for polyphonic pitch estimation setting the baseline in 2006 [4]. Recently, the CREPE model [2] achieved state-of-the-art accuracy for monophonic pitch estimation and was utilised in the original DDSP paper [1]. However, there has been no model that has matched CREPE's monophonic accuracy in the polyphonic domain. Bittner et al. proposed a deep learning-based approach for polyphonic pitch estimation in their paper on deep salience representations [5], laying the groundwork for further development. In their latest work, Basic-Pitch [6], Bittner et al. split the pitch detection pipeline into three tasks and created a lightweight, instrument-agnostic model that accurately detects pitch deviations and relates them to score-level note continuity. The accuracy of frame-level pitch detection is only slightly less than that of more computationally intensive, instrument-specific models.

Style/Timbre Transfer In the relatively young field of timbre transfer, earlier approaches such as the Universal Music Translation Network [7] relied on multiple separately trained decoders for domain transfer. This led to the timbre reconstruction falling entirely on the decoder, causing costly training. In contrast, Engel et al. [1] split the encoding between a fundamental pitch encoder, a residual encoder for timbre, and a raw extracted loudness envelope. Their model has a strong pre-baked music theory foundation, which allows it to require less training time and data to specialise to a specific domain and generate high-quality audio. However, the model's restrictive monophonic pitch detector and lack of latent interpolation leave significant room for improvement.

Audio Generation Audio generative modelling encompasses various disciplines, such as music, speech, and sound design. Various methods have been developed to address the complexity and controllability of generating high-quality audio. WaveNet [8] is a pioneering autoregressive generative model that produces realistic audio. However, it comes at a high computational cost, particularly for long output sequences. Recently, models based on techniques such as Diffusion [9] and Language Modelling [10] have been developed that produce excellent audio quality from minimal input. However, these models are also computationally costly and lack fine user control.

In contrast, Spectral Modelling Synthesis (SMS) [11] is a lightweight and modular approach that splits audio into harmonic and noise components [12]. These components can be generated separately using simpler techniques like additive and subtractive synthesis, driven by simple parameters. SMS provides greater control over the synthesis process, avoiding issues such as phasing alignment and spectral leakage and offering fully parametric flexibility. Thus, it is capable of producing quality audio from minimal training.

3 Methodology

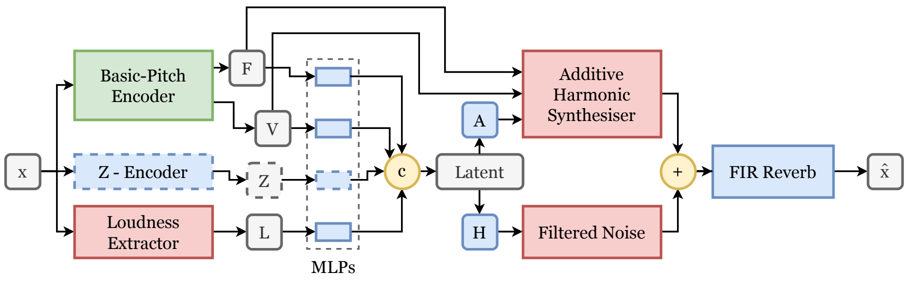

Fig. 1: Pathway through the model, with deterministic elements in red, trainable elements in blue, and the pre-trained elements in green. Tensor operations are labelled in yellow (with c being concatenation and + being addition) and dashed lines indicate optional components.

In this section we will discuss the methodology currently implemented within the proposed audio generative approach as well as the DDSP components that have been modified for multi-channel operation. Within this paper, each audio channel within the model will be referred to as a voice, in line with more traditional synthesis vocabulary.

3.1 Encoders

Pitch encoder: The model utilises a pre-trained basic pitch encoder developed by Bittner et al. [6]. Unlike most AMT models, this fully-convolutional model generates three distinct posteriorgrams (Y_o, Y_p, Y_n), each representing a different aspect of musical transcription. Y_o captures note onsets, Y_p tracks fine pitch, and Y_n records note events. This approach enables the model to achieve precise frequency quantised note-tracking while retaining detailed pitch information necessary for expressive performance, such as bends and vibrato.

To ensure continuous reproduction for each note during the synthesis step, we use the note-tracking Y_n to allocate each full note instance to a single voice in the pitch encoding F. As new notes appear, we assign them to the next inactive voice to ensure multiple non-overlapping voices. Finally, we apply fine pitch changes from Y_p to more closely match the input pitch and create a matching array within V with relevant note velocity values for each note in F.

Z-Encoder In certain musical instruments like the violin and piano, each performed note is typically played with a consistent timbre[3]. However, with instruments like the

[3] This is in the context of the general performer. Virtuoso performers will often use many techniques to explore the timbral aspects of their instrument for expressive performance.

guitar, a single pitch can be and is often produced with a diverse range of timbres, influenced by factors like the neck position and use of extended techniques. To accurately reproduce the audio in these cases, it is important to extract additional timbre information from the input and therefore we employ the optional Z-Encoder.

The Z-Encoder extracts this extra timbre information from the input audio in the form of Mel Frequency Cepstrum Coefficients (MFCCs). These coefficients are extracted by analysing the spectral envelope of the audio through a log mel-scaled spectrogram and they represent the distribution of spectral energy across the frequency scale. To utilise these coefficients, they are passed through a scalable normalisation layer, followed by a 512 unit Gated Recurrent Unit (GRU), then finally each time-step is fed through a linear layer to obtain Z, a frame-wise timbre representation for the input audio.

To improve the accuracy of timbre construction and reduce model complexity in reconstruction steps further into the model, we are developing a novel convolution based, pitch-informed source separation step. Utilising the temporally aligned transcription of the audio input provided by our pitch encoder, we can more easily separate individual voices by fundamental frequency from input spectrograms using simple lightweight convolution steps. This will allow for individual voices to have unique MFCCs and loudness envelopes to more closely reconstruct each voices timbre at later stages in the model, more closely to the much simpler monophonic case.

Loudness Extraction: To extract a loudness envelope, we also utilise the same steps as Hantrakul et al. [13] based on a simple psychometric model of perceived loudness. An A-Weighting of the power spectrum of input audio is log-scaled and centred according to the mean and standard deviation of the whole dataset. This specific weighting places higher value on higher frequencies to more closely match human perception.

3.2 Decoders

Latent Spaces and Envelopes: The majority of the control parameters driving the output synthesisers are contained in filter envelopes, A and H as shown in Figure 1. The tensor A contains a concatenation of both the global output amplitude envelope A_G, functionally controlling the ADSR envelope of each note generated, and each voice's harmonic spectra amplitude envelope $A_{v,i}$, responsible for creating the correct harmonic balance for each instrument. The envelope H controls the individual frequency bands in our filtered noise.

The multi-voice operation of the model requires the use of more complex latent space structure. Some components of the model perform better with limited input information, while other aspects require global information to function. Extracting each envelope from the latent space involves a two-step process: the specific features are fed through a GRU layer, followed by a dense linear layer. However, there is a distinction in the choice of features. In the case of $A_{v,i}$, the GRU layer for each voice only receives input related to the pitch, velocity, loudness, and timbre encoding of that specific voice. On the other hand, for the global amplitude envelope A_G and noise filter envelopes H, their respective GRU layers are fed with inputs consisting of encoding for the pitch and velocity of every voice, as well as the global amplitude and timbre.

Additive Harmonic Synthesiser: An Additive Harmonic Synthesiser is a type of synthesizer that generates complex waveforms by adding multiple simpler waveforms, typically sinusoidal waves. All acoustic instruments produce sound by using a resonating body, which is often a string or an air chamber. Due to the physics of standing-wave oscillations in resonant bodies, the timbre's generated by these instruments are characterised by a spectrum of harmonics. These harmonics start with a fundamental frequency denoted by f_0 and are followed by an infinite series of integer multiples of that frequency $i * f_0$. The key to recreating an instruments timbre lies in accurately recreating the correct balance of these harmonics via our harmonic amplitude envelope $A_{v,i}$.

In this model, the sinusoidal oscillator is constructed as a bank of $V * H$ oscillators, where V is number of voices and H is our set harmonic cutoff, that outputs signal $x(n)$ of discrete time steps n:

$$x(n) = A_G(n) \sum_{v=1}^{V} \sum_{i=1}^{H} A_{v,i}(n) \sin(\phi_{v,i}(n))$$

Where $A_G(n)$ is our global amplitude envelope, $A_{v,i}(n)$ is our specific harmonic amplitude envelope, $\phi_{v,i}(n)$ is the instantaneous phase at timestep n, obtained from the frequency embedding F_v as follows: $\phi_{v,i}(n) = 2\pi \sum_{m=0}^{n} i * F_v(m)$

In summary, the whole harmonic oscillator is parameterised by the three time dependent parameter sets: $F_v(n)$ the fundamental frequencies, $A_G(n)$ the global amplitude envelope and $A_{v,i}(n)$ the harmonic distribution for each voice.

Filtered Noise: Subtractive synthesis works in the opposite way to additive synthesis. Rather than compounding simple waveforms to create more complex sounds, it starts with a colourful audio signal such as white noise and filters it until it reaches the desired sound. In this work, we implement a filtered noise technique similar to that of Engel et al. [1] by applying a Linear Time-Variant Finite Impulse Response (LTV-FIR) filter to a stream of uniform noise. To process this efficiently, we use frame-wise convolution through multiplication in the Fourier domain. Our extracted envelope tensor, denoted as H, represents our filter convolution function for each frequency band. We then apply this filter to the Inverse Discrete Fourier Transform (IDFT) of uniform noise, N, to obtain Y. We convert back to the audio domain by taking the IDFT of Y, resulting in the framed audio output, y, from which we construct the full audiorate signal using overlap-add.

Reverb: In most neural synthesis models, the room reverb is baked into generative process, as it is an essential component of producing realistic sounding audio. In contrast, this model applies room reverb after synthesis using a convolution step. This approach offers several benefits: it allows for greater transparency by enabling the extraction of dry audio from the model, and it offers more control over the room acoustics in the generated audio. However, standard convolution via matrix multiplication is computationally intensive and can hinder training and performance. To address this, we utilise the same techniques as those used in the filtered noise model - explicit convolution via multiplication in the frequency domain, which has been found to produce sufficiently accurate reproduction.

3.3 Other Methodology

Upsampling: The information contained within audio data is very dense, and at pure audiorate, it has a resolution that is too high to work with in real-time, even at the reduced 16kbps used in this model. To solve this problem, the model employs audio frames with a 64-bit length, and the encoder only extracts information at this frame rate level before upsampling it back up to audiorate much later in the model for resynthesis. Each frame lasts for 4ms, which is fine enough to fully track changes in important attributes such as F_0 and loudness envelopes while reducing the temporal dimension of a second of audio from 16,000 to 250.

Bilinear interpolation is sufficient to upsample discrete variables, such as F_0, for parameterising the additive synthesizer. However, when it comes to smoothing the upsampling of various continuous envelopes and preventing artifacting, we use overlapping Hamming windows centered at each frame.

Spectral reconstruction loss: For our training objective we utilise spectral reconstruction loss. This will allow for comparisons without considering audio phase differences between input and output, as these will not affect how the reconstruction sounds and therefore are not important to consider during the training process.

For input spectrogram $\mathcal{S}$ and reconstruction $\hat{\mathcal{S}}$, the L1 loss for a given spectrogram is as given:

$$\mathcal{L} = ||\mathcal{S} - \hat{\mathcal{S}}||_1 - \alpha|| \log \mathcal{S} - \log \hat{\mathcal{S}}||_1,$$

where α is log weighting term. This is summed over multiple FFT window sizes i to get a multi-scale loss $\mathcal{L}_{multi-scale} = \sum_i \mathcal{L}_i$. Calculating the sum of different windows sizes produces a better match over multiple resolutions, some fine detail matching without loss of the overall picture.

4 Experiments

The proposed polyphonic model's effectiveness hinges on two key contributions: extending the DDSP model for polyphonic use ensuring no loss in performance compared to the monophonic case, and the effectiveness of a stochastic latent space for learning various timbres within one model. To evaluate, tests mirroring the original DDSP paper will be conducted, assessing timbre and loudness accuracy through MFCC and Loudness L_1 deviations on GuitarSet and the Solo Violin dataset. GuitarSet allows for complex timbre test on a polyphonic dataset, while Solo Violin allows a direct comparison to the older model. The models performance on both sets combined will showcase the abilities of the stochastic latent space.

4.1 Datasets

GuitarSet [14] is a dataset of 360 audio recordings of guitars, each lasting approximately 30 seconds, featuring six different players playing 30 musical leadsheets across various genres and tempos in both comping and soloing styles. The recordings were played on the same guitar with consistent room acoustics to ensure uniform timbre.

Solo Violin comprises 13 minutes of monophonic solo violin performances by John Garner from the MusOpen royalty-free music library. All performed on a single violin in a consistent room environment.

4.2 Evaluation Metrics

MFCC L_1 Distance: An important measure of the models reconstruction accuracy is it's ability to match input and output timbre. Mel Frequency Cepstrum Coefficients are used as part of the timbral encoder (z-encoder) step as they are an accurate representation of timbral quality and so two indentical timbres at the same pitch should produce identical MFCCs. The L_1 distance between input and output MFCC vectors should provide a representative measure of the models ability to match timbre.

Loudness L_1 Distance: Similarly to MFCCs, the reconstructed track should produce an identical loudness envelope if it is reproduced accurately. Again, this is computed by computing the L_1 distance between the ground truth audio and the synthesised audio's loudness encoding L. Please note that neither of these metrics are used by the model to evaluate during training, so there should be no inherent training bias.

References

1. J. Engel, L. Hantrakul, C. Gu and A. Roberts. DDSP: Differentiable Digital Signal Processing. ICLR (2020)
2. J. W. Kim, J. Salamon, P. Li and J. P. Bello. CREPE: A Convolutional Representation for Pitch Estimation. arXiv (2018)
3. DDSP-VST. `https://magenta.tensorflow.org/ddsp-vst` (2022)
4. A. Klapuri. Multiple Fundamental Frequency Estimation by Summing Harmonic Amplitudes. ISMIR (2006)
5. R. M. Bittner, B. McFee, J. Salamon, P. Li, and J. P. Bello, Deep Salience Representations for f0 Estimation in Polyphonic Music. ISMIR (2017)
6. R. M. Bittner, J. J. Bosch, D. Rubinstein, G. Meseguer-Brocal and S. Ewert. A Lightweight Instrument-Agnostic Model for Polyphonic Note Transcription and Multipitch Estimation. arXiv (2022)
7. N. Mor, L. Wolf, A. Polyak and Y. Taigman. A Universal Music Translation Network. ICLR (2018)
8. A. v. d. Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior and K. Kavukcuoglu. WaveNet: A Generative Model for Raw Audio. arXiv (2016)
9. S. Forsgren and H. Martiros. Riffusion - Stable diffusion for real-time music generation. `https://riffusion.com/about` (2022)
10. Z. Borsos, R. Marinier, D. Vincent, E. Kharitonov, O. Pietquin, M. Sharifi, O. Teboul, D. Grangier, M. Tagliasacchi and N. Zeghidour. AudioLM: a Language Modeling Approach to Audio Generation. arXiv (2022)
11. X. Serra and J. Smith. Spectral modeling synthesis: A sound analysis/synthesis system based on a deterministic plus stochastic decomposition. Computer Music Journal (1990)
12. J. W. Beauchamp. Analysis, synthesis, and perception of musical sounds. Springer (2007)
13. L. Hantrakul, J. Engel, A. Roberts and C. Gu Fast and Flexible Audio Synthesis ISMIR (2019)
14. Q. Xi, R. Bittner, J. Pauwels, X. Ye, and J. P. Bello. Guitarset: A Dataset for Guitar Transcription. ISMIR (2018)