



Audio Engineering Society

Conference Paper 1

Presented at the AES International Conference on Artificial Intelligence and
Machine Learning for Audio
2025 September 8-10, London, UK

This paper was peer-reviewed as a complete manuscript for presentation at this conference. This paper is available in the AES E-Library (<http://www.aes.org/e-lib>) all rights reserved. Reproduction of this paper, or any portion thereof, is not permitted without direct permission from the Journal of the Audio Engineering Society.

Motor2Synth: Leveraging Differentiable Digital Signal Processing for Generating Combustion Engine Sounds Compatible with Active Sound Design Frameworks

Thiago Henrique Gomes Lobato¹, Stefan Hank¹, Zhang Hanyi², Luo Haofu²

¹HEAD acoustics GmbH

²Xiaomi

Correspondence should be addressed to Thiago Lobato (Thiago.lobato@head-acoustics.com)

ABSTRACT

In Active Sound Design (ASD) applications for electric vehicles, many manufacturers want to have at least some sounds that emulate aspects of combustion engine noise, while being different enough to show their own identity. Consequently, combustion engine sounds frequently serve as the foundation for creative sound design. Traditional artificial sound generation methods, however, struggle to replicate the distinct acoustic properties and dynamics of combustion engines, leading to unrealistic results. Although real vehicle recordings are available, their chaotic nature and significant RPM fluctuations make them challenging to process for sample-based synthesis. This paper introduces a novel approach to model arbitrary combustion engine sounds using artificial intelligence, specifically through Differentiable Digital Signal Processing (DDSP). DDSP models a sound or process with differentiable implementations of typical signal processing modules such as harmonic generator, noise filter and reverberation. The model can then generate sounds based on an estimation of a fundamental frequency and signal loudness, effectively as a synthesizer, being the reason, we call our tool Motor2Synth. For motor sounds, however, the fundamental frequency should be the rotational speed (RPM), and pre-trained models for music such as CREPE fail to provide good results. To address this, we developed a self-supervised pitch estimator based on the PESTO model to estimate rotational speed, integrating it into a DDSP framework to generate sounds. We show that our model can correctly estimate the rpm of combustion engine noise while also producing realistic outputs that can be used as base for ASD applications. Finally, we demonstrate a designed sound using a DDSP version of a high-performance engine as a component to create a sound variant for the car Xiaomi SU7 and Xiaomi SU7 Ultra.

1 Introduction

In recent years, the percentage of electric vehicles being produced and purchased has sharply increased [1]. These vehicles have a different, less prominent sound signature than combustion engines, which allows engineers to shape the vehicles' identity with their own sound compositions through Active Sound Design (ASD). The types of sounds can vary from very futuristic to classical sporty tones, as long as

they fulfil the user's expectations. These expectations often rely on familiar sounds; thus, classical combustion engine sounds serve as a basis for many designs. To generate ASD sounds, a combination of synthesizers and digital signal processing (DSP) effects can be used. This approach works especially well when developing futuristic sounds but is limited when emulating combustion engine sounds due to their unique characteristics and variations as a

function of the car's rotational speed. These characteristics are particularly difficult to emulate and demand a high amount of manual fine-tuning for each individual case, which is very expensive or impractical when modelling many different motors. An alternative would be to use recorded motor samples, but commercially available options mostly contain recordings with constant acceleration, which are not suitable for the often-used sample-based synthesis which is applied in electric cars to process and playback the sound interactively.

Recently, the use of Differentiable Digital Signal Processing (DDSP) has proven effective in tasks such as instrument modification [2], piano synthesis [3], automatic effect transfer [4], wavetable synthesis [5], sound matching of a synthesizer [6], modelling of animal sounds [7] and many others [8]. DDSP works by modelling classical DSP approaches as differentiable elements in an automatic differentiation framework such as PyTorch [9]. For instrument modelling, a combination of harmonic generation and filtered noise is commonly used [2], possibly with additional support effects like reverb. This allows one to learn any instrument as a function of its loudness and pitch using only a small number of recordings. Loudness is usually calculated as the A-weighted sound pressure level, and pitch is estimated with neural models such as CREPE [10].

Unlike conventional engine-sound pipelines, DDSP can learn the motor's acoustic behaviour end-to-end. This is achieved since its harmonic oscillators and noise generators are implemented as trainable DSP block so that a DDSP model 1) automatically captures narrow-band resonances of the block, exhaust and chassis without manual tuning, 2) separates and parametrises broadband noise so that intake hiss or valve clatter can be shaped independently, and 3) extrapolates to unseen RPM trajectories simply by driving the latent pitch and loudness controls beyond the training set [2].

By contrast, procedural / physical-model synths must represent each acoustic cavity with explicit filter coefficients, and thus matching the coupled resonances of real hardware quickly becomes labour-intensive. Additionally, the hand-tuned model can break when the target engine geometry or load profile changes [11]. Sample-based and granular replay systems avoid that design effort but tie fidelity to the coverage of the original recordings: wide-range pitch-shifting smears formants, and the fixed loops make it hard to decouple tonal orders from stochastic components, limiting both realism and design space [12].

This work explores the use of a DDSP framework to model motor sounds as "instruments", so they can be used as synthesizers in classical digital audio workstations (DAWs). We demonstrate that with a DDSP approach, it is possible to fully transform a motor into a synthesizer with satisfactory accuracy for creative applications. We start by presenting the dataset and methodology used, then we show our results and the application of our approach to create a component for a real car ASD, the Xiaomi SU7/SU7 Ultra. A full representation of our method can be seen in Figure 1.

2 Data

To train and evaluate the model, we used sample packs of combustion engine noise with the total of 9 vehicles, namely:

- 1966 Chevelle SS Big Block
- 1966 Chevelle SS Small Block
- 1966 Mustang
- 1968 Galaxie
- 1969 Satellite
- 1976 Regal
- 1997 Express
- 2007 Durango
- 2007 GT500

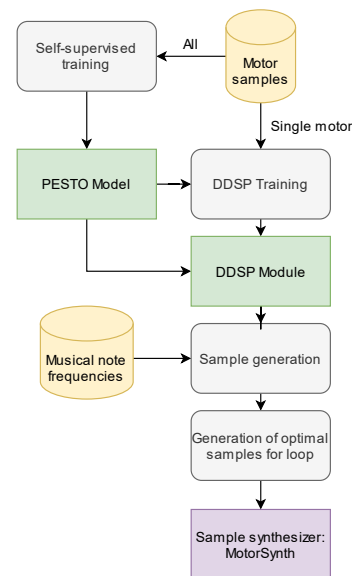


Figure 1. Schematic of all steps necessary to get from motor samples to a musical synthesizer in ASD applications using our Motor2Synth approach.

This list encompasses motors considered “vintage” as well as relative modern combustion engine ones. The

samples were bought from the boom Library [13] and contained diverse measurements for different conditions. We pre-processed the samples removing the silence, so that only the parts in which the motor was running were considered for the training. The total length of “useful” audio for each motor was around 15 minutes, which is a relatively small amount and was enough to generate good results. The spectrogram of some of the generated signals can be seen in Figure 2.

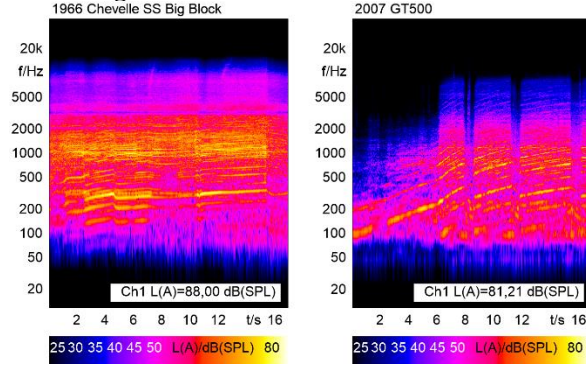


Figure 2. Motor training samples given to the DDSP model.

We see that each motor has a very different harmonic distribution and different resonances as function of the RPM, which explains the challenge of properly modelling them.

3 DDSP model

For the DDSP model we use the Pytorch DDSP framework [14] to train a harmonic plus noise model with a differentiable reverberation module. The standard noise implementation of the library was a concatenation of filtered noise, which we modified to perform an overlap-and-add synthesis. The actual sound model of a single block s_b is simply defined:

$$s_b = A \cdot \sum_{i=1}^{N_h} a_i \sin(2\pi f_0 \cdot i \cdot t) + h(t) * n(t) \quad (1)$$

For the harmonic part, N_h is the number of harmonics (in our case 100, which was necessary to model modern motors), A the global harmonic amplitude, a_i the strength of the partials between 0 and 1 and f_0 is the fundamental frequency (in our case the rotational speed). For the noise part, $h(t)$ is an impulse response defined with a FIR filter with 75 samples and $n(t)$ is sampled from a Gaussian noise with mean zero and variance 1. Blocks are then combined in an overlap-and-add fashion with 75% overlap and using a

Hanning window. Afterwards, the resulting signal is processed with a learnable reverberation module defined as a noise with exponential decay. The data sampling rate was set to 16000 Hz (since motor harmonics rarely go above 8000 Hz) with 160 samples block size.

The standard pitch estimation of the used Pytorch DDSP framework [14] is CREPE [10], which was trained mainly for musical instruments and voice. In the results section we will show that the version of CREPE trained with musical instruments is not suitable to identify RPM of motor signals, and thus it is not possible to create a reasonable motor model with it. To train a new CREPE, labelled RPM data would be needed, which is expensive to get, especially for sound-sample data. As an alternative, we trained an RPM estimator using the PESTO [15] model. The main idea of the model is that it is easier to learn relative pitch differences than absolute ones. Thus, if we can apply a known pitch shift to a signal and have a model that has equivariant outputs to this pitch difference, we just need a single calibration afterwards to have a good estimator. Additionally, we can add noise to our data for which the shift is invariant (this can be either white noise or any disturbance that should not affect the shift) and train the model to identify those. This can then be learned in a fully unsupervised fashion; thus, we do not need labels. The RPM estimator was trained using the data of all vehicles. To consider that each vehicle may have resonances at different harmonics/orders, thus possibly confusing the RPM estimators, a post-processing of the estimated rotational speed with a scalar factor to maximize the number of orders with high energy can be done. This factor can also be used to consider the high Subharmonic-to-Harmonic Ratio (SHR) of motor sounds by selecting an “artificial” RPM equal to half the actual RPM. In this way, the DDSP model is also able to handle half-orders, which are very prominent in motor sounds.

At last, to ensure the pitch is mostly continuous between time steps, we modelled the time-pitch transition as a hidden-Markov model that favours continuous estimations, so that sharp transitions on the RPM were penalized (this sampler was also used for CREPE estimator but without success). Additionally, we used a moving average smoothing to compensate the discrete resolution of PESTO.

To evaluate the estimation, we estimated the ground-truth rpms of 100 samples from different cars with the RPM generator tool from the software ArtemiS. As a fully signal-processing baseline, we also added the YIN [16] pitch estimator to the evaluation. To compensate possible boundary effects

due to the PESTO pitch smoothing, we removed the boundaries not fully in the average window from all comparisons. Additionally, since we used “artificial” rpms to handle sub-harmonics, we evaluate the estimation up to a factor that is calculated for each case individually.

4 Training scheme

To learn the actual DDSP model, we trained each vehicle individually. The training was done with the AdamW optimizer with 1e-3 learning rate with cosine schedule to 1e-4. 80 % of the data was used for the training and 20% for the validation. The loss was a multi-scale STFT loss defined as follows:

$$L = \frac{1}{M} \cdot \sum_{i=1}^N \|S_{i,x} - S_{i,y}\|^1 + \|\log(S_{i,x}) - \log(S_{i,y})\|^1 \quad (2)$$

in which $S_{i,x}$ is a spectrogram with a block size defined by i and M is the number of samples. We used scales with length {128,256,512,1024,2048,4096} containing 75% overlap in time. This loss allows one to evaluate the representation in multiple time-frequency scales.

We noticed that for all vehicles the loss could reach validation values of about 2.X. However, by training the model that much we ended up with DDSP models that 1) extrapolate badly above the frequency/loudness range of the sample data (which was not so large) and 2) used too much tonal components to compensate the noise, generating very “metallic” sounds. By searching other stop criteria, we found out that interrupting training at the epoch in which the validation loss crossed a threshold below 5.0 was enough to produce realistic generations while retaining reasonable extrapolations above and below the training range. This value coincides with the recommendation found in the Magenta DDSP training notebook example found in their code repository [2].

5 Results: RPM estimation

In Figure 3 we show results of the absolute frequency deviation of the different frequency estimators for 100 manually labelled samples. We see that CREPE has an average error of 17.3 % and some very strong outliers. This result is very similar to the YIN method, which shows that CREPE has an accuracy equivalent to a typical signal-processing approach under the domain-shift from musical instruments to motor

sounds. The PESTO method, on the other hand, shows an error of only 0.8%, which is an order of magnitude lower and, in musical terms, is less than half a semi-tone (or around 18 cents). This indicates that for RPM estimations, both YIN and CREPE trained mainly for musical instruments are not as appropriated as a tailored PESTO model, which is not particularly surprising, since they both represent very different type of sounds and the motor sounds have a considerably higher noise level.

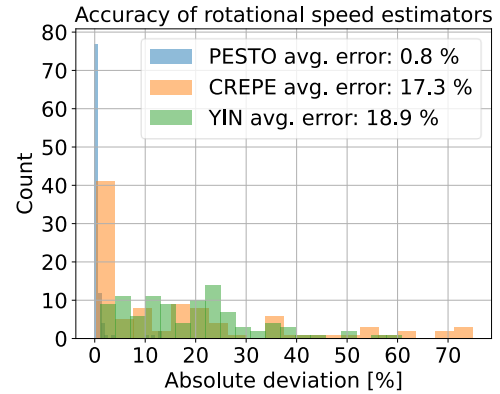


Figure 3. Accuracy of frequency estimation in the 100 manually labelled examples for different estimators.

To grasp the meaning of this difference in accuracy for the pitch estimation, we show in Figure 4 and 5 below some estimations in real signals done both with PESTO and CREPE, in which it becomes clear that the pre-trained CREPE model is not able to consistently estimate a valid rotational speed for motor sounds, as it shows sharp discontinuities in signals with continuous changes. It is however very likely that by training CREPE with motor data, the result will improve. However, getting the training data is a very expensive endeavour that we found to be unnecessary due to the high accuracy of PESTO.

6 Results: sound reproduction

Spectrograms of some results can be seen in the examples that follow in the Figures 6 and 7 for different vehicles, all with SFTF-multi scale errors of around 5.0.

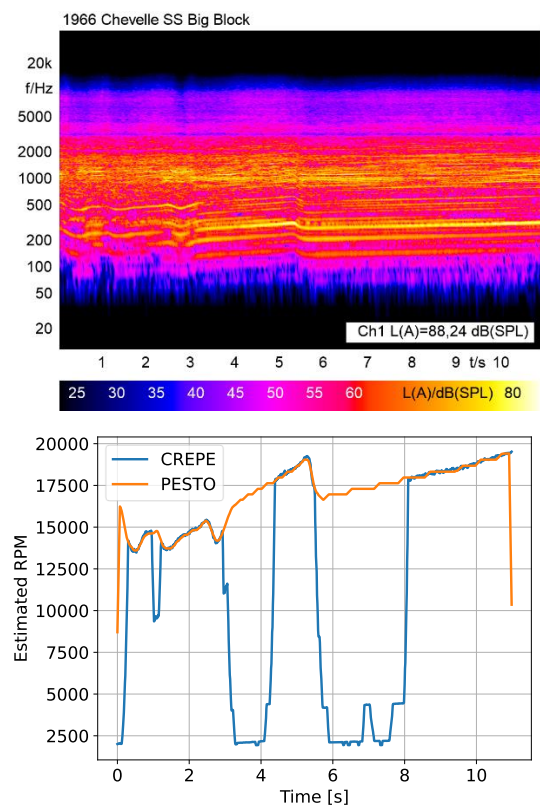


Figure 4. Estimated rotational speed with CREPE and PESTO of a 1966 Chevelle sample.

All sounds are still easily distinguishable in an informal A/B comparison. However, to our main sound designer, the fidelity was judged enough for creative applications, especially since the idea is to use those sounds as basis or layers in a sound composition. Thus, we have defined a process in which we can approximately represent a set of combustion engine motor recordings in a full parametric fashion, with respect to a loudness and pitch distribution, that is good enough for a sound designer to use as basis for new sounds.

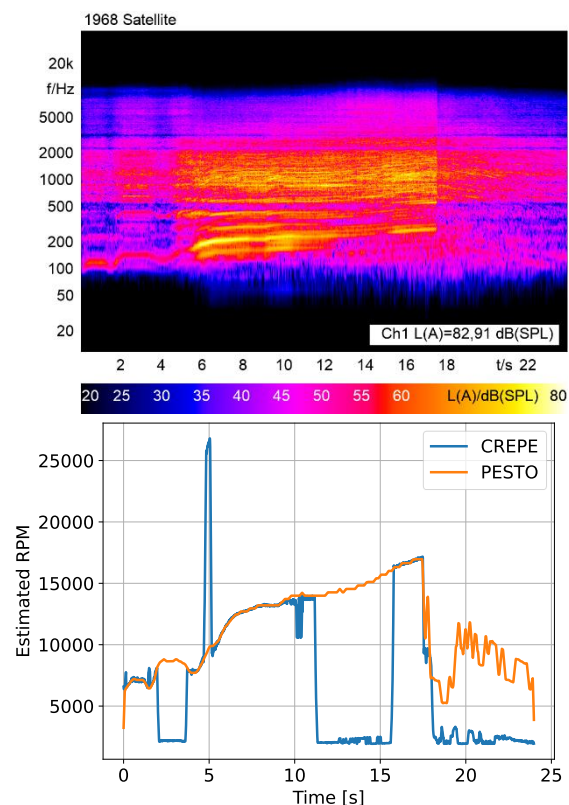


Figure 5. Estimated rotational speed with CREPE and trained PESTO of a 1968 Satellite sample.

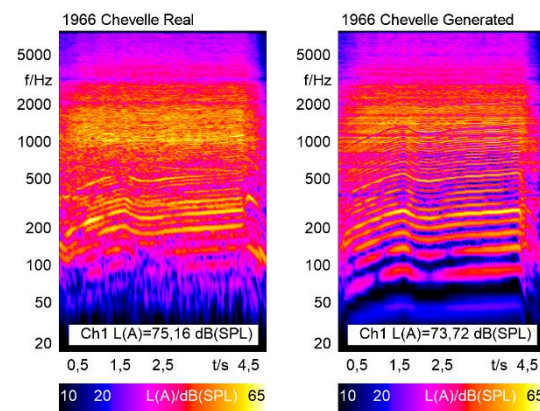


Figure 6. Example of the DDSP reconstruction of a 1966 Chevelle Big Block.

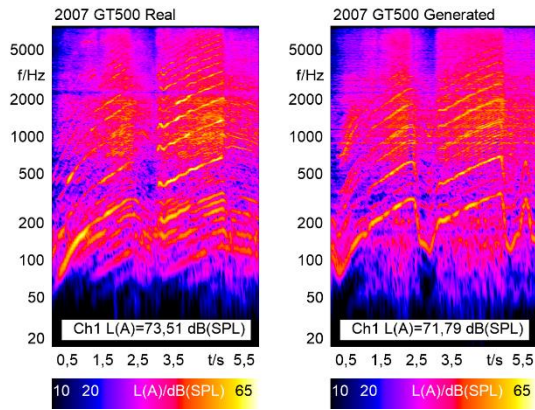


Figure 7. Example of the DDSP reconstruction of 2007 GT500.

A good example of this parametric capability can be seen in Figure 8 below in which we perform a “motor transfer”, namely transforming an input measurement of a vintage 1966 Chevelle to a more modern 2007 GT500. This is done by estimating the loudness and pitch from the Chevelle and use those as inputs to the GT500 model.

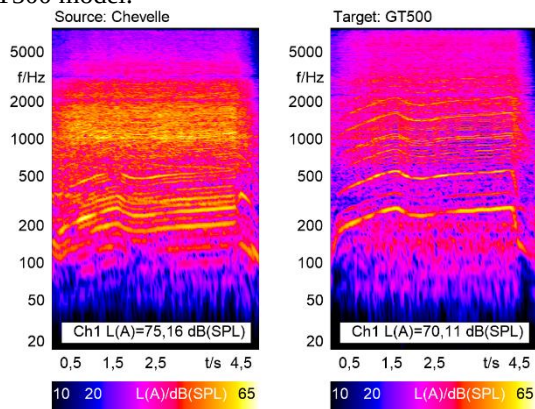


Figure 8. Example of a “motor transfer”, in which left we have the input 1966 Chevelle motor and right the target 2007 GT500.

Besides the motor transfer approach, one possible creative application is motor interpolation. This could be done either by 1) assigning a latent variable to each motor and then interpolating between both or by 2) interpolating the filter and harmonic parameters given by two different models. Although we don’t investigate such an approach in this paper, we see it as a very interesting creative application for future research.

7 Data pre-processing for ASD

In ASD applications for electric vehicles, sounds need to follow a given interface and be connected to the main inputs of the car such as the user pedal input. Algorithms also need to be efficient and low energy. We achieve this by using the software QNX® Sound from Blackberry, which allow us to obtain drivers’ inputs and use those to generate our sounds, which we do based on samples. Those samples are generated by sampling the different motors at given musical pitches and different levels of loudness. This can be seen, in fact, as a downgrade from the actual DDSP model, however, it is necessary step to a practical implementation. Some advantages of the DDSP model, though, such as range extrapolation and separation of tonal and noise components can still be leveraged.

For the samples to be usable, we need to place them on a loop, so that they comply with the application of possibly indefinitely generation during the drive. Thus, it is important that the transition between end and start of the loop cannot be perceived by the driver, which could disturb their immersion and be perceived as annoying.

For motor samples, a simple cross-fade is not enough to produce good loops, since they possess a strong modulation that is audible if the sound loops at the wrong position. Additionally, due to the noise, the perfect cut point does not exactly match the signal period. To solve this problem, we searched the position that best matched the initial window of the signal. This approach makes sure that there’s no strong discontinuity in the signal, both in respect to individual samples as well as to longer modulations.

An example of this process can be seen in Figure 9, in which a window is defined, matched and the signal is looped. One can see that both the scale of the window (long modulation) as well as in the scale of individual samples the resulting signal looks healthy and, indeed, does not produce any hearable artefact.

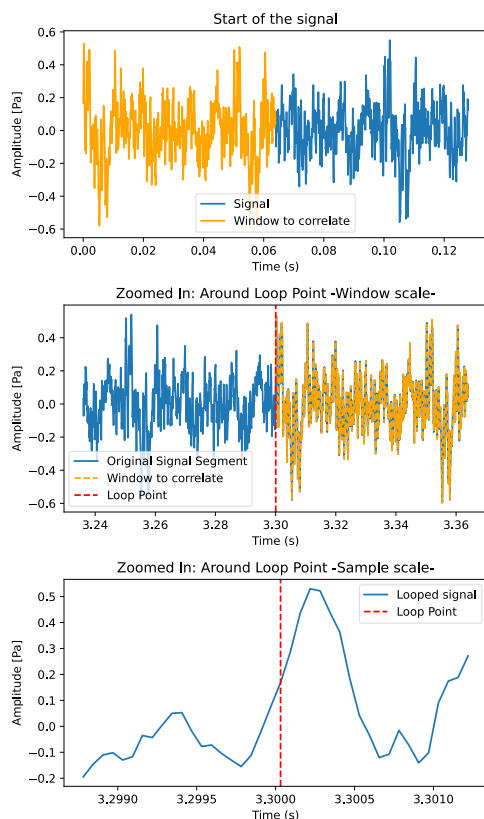


Figure 9. Process to estimate ideal cut position for the signal. An initial window is defined and then the best correlation in the signal is sought. A cut point is then selected as the one that produces the smoothest loop transition.

8 Creation of a real ASD sound

The Xiaomi SU7 is an electric full-size sedan with a unique and sporty visual design. The challenge of the active sound design process was to create a sound that matches all different power versions of the Xiaomi SU7 and creates a unique and distinctive sound, that fits to the visual design and supports the driving experience.

In total, the vehicle has three different active sound designs, which are all created with very different techniques. The purpose of the sound design utilizing the Motor2Synth Engine was to reinterpret the sound and emotion of a V8 engine.

For the sound design, a car with a V8 motor engine was selected as basis and a DDSP model was created in the exact same matter as the other cars so far. An example of a real and reconstructed sample is shown in Figure 10 (with a multi-scale STFT loss of ca. 4.9).

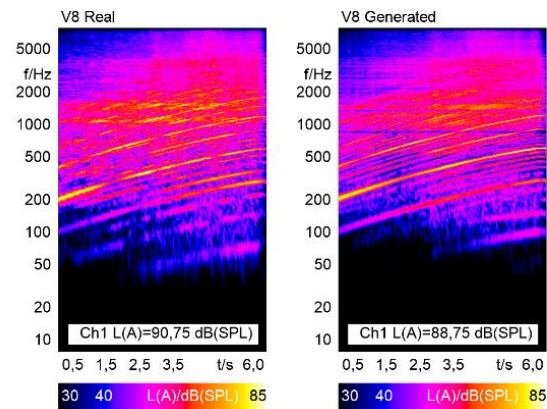


Figure 10. Example of the real and generate V8 motor used as basis for the Xiaomi SU7.

For the dynamic playback of the active sound design, a sample-based and an additive synthesis approach were used. This allows a good tradeoff between creative flexibility and efficient processing of the sound during driving. Since the car does not shift gears and the sound also does not have a virtual gear shifting, one challenge was to avoid a too high pitch shift of the used sound samples. By using a Shepard tone-like structure, the samples could cover the entire speed range by only shifting two octaves in pitch. The samples for the ASD sound were derived from the Motor2Synth engine. By utilizing the ability of the Motor2Synth engine to render sounds with different pitches, single layers for the low, mid and high frequency range could be rendered and filtered to reconstruct a unique V8 like sound.

With additional processing, using frequency filter, saturation, compression and modulation-based effects a futuristic character could be added to the V8 sound. After applying the effects, the sound samples had to be rendered and recut to result in an audio sample that can be looped perfectly. However, since the Motor2Synth engine already provides quasi periodic samples, this process was easier. In Figure 11 a spectrogram of the final sound can be seen.

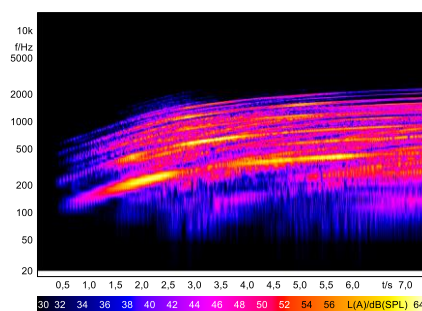


Figure 11. The created V8 sound design without wind-, road and powertrain noise.

9 Conclusion

We've shown that it is possible to use a DDSP approach with a differentiable harmonic generator plus noise module to learn "motor synthesizers" based only on a couple of minutes of training data. Those models can then be used to generate samples for a sample-based synthesizer and, even though they are still perceptually different from the original one, are deemed realistic enough for creative applications by a sound designer.

In this work, we also demonstrated that the main fundamental frequency estimators used for instrument modelling are not appropriate for motor sounds, likely due to the sound characteristics of combustion engines and musical instruments being considerably different. This was mitigated by training a model specific for motor sounds using the self-supervised model PESTO. This model was validated both by the quality of the generated sounds as well as by utilizing artificial generation, in which the ground truth of the rotational speed was known. This test showed that the CREPE model and YIN algorithm had a similar average estimation error of, respectively, $\sim 17\%$ and $\sim 18\%$, while PESTO had an error of only $\sim 1\%$, one order of magnitude lower, showing that classical pitch estimators used in music are not optimal for estimating the RPM. Figures 4 and 5 of this paper show what this magnitude difference means in the results.

At last, we show that our approach is ready and relevant for real applications, since it is already in use on a real electric vehicle, the Xiaomi SU7. A future work could be done to model different motors with the same network backbone but using different embeddings for each vehicle, so that designers could sample the latent space and easily interpolate between very different motors. An interesting creative application of this could be to allow drivers themselves to manipulate such a latent space based on pre-made designs to obtain the "perfect" individual sound.

References

- [1] International Energy Agency, Global EV Outlook 2023. Paris, France: IEA, 2023. [Online]. Available: <https://www.iea.org/reports/global-ev-outlook-2023> (accessed 30 Jun 2025)
- [2] J. Engel, L. Hantrakul, C. Gu, and A. Roberts, "DDSP: Differentiable digital signal processing," in Proc. Int. Conf. Learning Representations (ICLR), Virtual/Addis Ababa, Ethiopia, 2020.
- [3] L. Renault, R. Mignot, and A. Roebel, "Differentiable piano model for MIDI-to-audio performance synthesis," in Proc. 25th Int. Conf. Digital Audio Effects (DAFx20in22), Vienna, Austria, 6–10 Sep 2022.
- [4] C. J. Steinmetz, N. J. Bryan, and J. D. Reiss, "Style transfer of audio effects with differentiable signal processing," arXiv preprint arXiv:2207.08759, 2022.
- [5] S. Shan, L. Hantrakul, J. Chen, M. Avent, and D. Trevelyan, "Differentiable wavetable synthesis," in Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing (ICASSP), Singapore, 22–27 May 2022, pp. 4598–4602.
- [6] N. Masuda and D. Saito, "Synthesizer sound matching with differentiable DSP," in Proc. 22nd Int. Soc. Music Information Retrieval Conf. (ISMIR), Online, 7–12 Nov 2021, pp. 428–434.
- [7] M. Hagiwara, M. Cusimano, and J.-Y. Liu, "Modeling animal vocalizations through synthesizers," arXiv preprint arXiv:2210.10857, 2022.
- [8] B. Hayes, J. Shier, G. Fazekas, A. McPherson, and C. Saitis, "A review of differentiable digital signal processing for music and speech synthesis," *Frontiers in Signal Processing*, vol. 3, 2024, Art. 1284100, doi: 10.3389/frsip.2023.1284100.
- [9] A. Paszke et al., "PyTorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32 (NeurIPS 2019)*, Vancouver, Canada, 2019, pp. 8024–8035.
- [10] J. W. Kim, J. Salamon, P. Li, and J. P. Bello, "CREPE: A convolutional representation for pitch estimation," in Proc. IEEE Int. Conf. Acoustics, Speech and Signal Processing (ICASSP), Calgary, Canada, 15–20 Apr 2018, pp. 161–165.
- [11] S. Baldan, H. Lachambre, S. Delle Monache, and P. Boussard, "Physically informed car engine sound synthesis for virtual and augmented environments," in Proc. IEEE 2nd VR Workshop on Sonic Interactions for

- Virtual Environments (SIVE), Arles, France, 24 Mar 2015, pp. 1–6, doi: 10.1109/SIVE.2015.7361287.
- [12] J. Jagla, J. Maillard, and N. Martin, “Sample-based engine noise synthesis using an enhanced pitch-synchronous overlap-and-add method,” *J. Acoust. Soc. Am.*, vol. 132, no. 5, pp. 3098–3108, Nov. 2012, doi: 10.1121/1.4754663
- [13] Boom Library, Cars – V8 (sound-effects collection). [Online]. Available: <https://www.boomlibrary.com/sound-effects/>.
- [14] L. Renault, `ddsp_pytorch` (Version 1.0) [Source code]. GitHub, 2022. Available: https://github.com/acids-ircam/ddsp_pytorch (accessed 30 Jun 2025)
- [15] A. Riou, S. Lattner, G. Hadjeres, and G. Peeters, “PESTO: Pitch estimation with self-supervised transposition-equivariant objective,” in *Proc. 24th Int. Soc. Music Information Retrieval Conf. (ISMIR)*, Milan, Italy, 5–9 Nov 2023, pp. 535–544.
- [16] A. de Cheveigné and H. Kawahara, “YIN, a fundamental frequency estimator for speech and music,” *J. Acoust. Soc. Am.*, vol. 111, no. 4, pp. 1917–1930, Apr. 2002, doi: 10.1121/1.1458024.