ICGAN: AN IMPLICIT CONDITIONING METHOD FOR INTERPRETABLE FEATURE CONTROL OF NEURAL AUDIO SYNTHESIS

Yunyi Liu

Computing and Audio Research Lab
University of Sydney
Sydney, Australia
`yunyi.liu@sydney.edu.au`

Craig Jin

Computing and Audio Research Lab
University of Sydney
Sydney, Australia
`craig.jin@sydney.edu.au`

ABSTRACT

Neural audio synthesis methods can achieve high-fidelity and realistic sound generation by utilizing deep generative models. Such models typically rely on external labels which are often discrete as conditioning information to achieve guided sound generation. However, it remains difficult to control the subtle changes in sounds without appropriate and descriptive labels, especially given a limited dataset. This paper proposes an implicit conditioning method for neural audio synthesis using generative adversarial networks that allows for interpretable control of the acoustic features of synthesized sounds. Our technique creates a continuous conditioning space that enables timbre manipulation without relying on explicit labels. We further introduce an evaluation metric to explore controllability and demonstrate that our approach is effective in enabling a degree of controlled variation of different synthesized sound effects for in-domain and cross-domain sounds.

1. INTRODUCTION

In recent years, neural audio synthesis has achieved excellent performance in speech modelling [1, 2] and music generation [3, 4]. However, it remains difficult to model sound effects with pure generative models in that neural networks typically offer a low degree of interpretability and controllability. A common approach to achieving sound generation control is via conditioning on external information. Conditional generation is a common technique to constrain generative models to generate data based on specific input information, which can be categorical labels, texts, images, compressed embeddings, etc. Considering the limited availability of datasets for general sound effects, conditioning on discrete labels can be effective, as most datasets provide at least categorical information. For example, Barahona-Ríos and Collins [5] proposed to synthesize knocking sounds conditioned on different emotions to guide the GAN generator for audio generation. Similarly, Communita et al. [6] also studied GAN sound generation for modelling footstep sounds conditioned on discrete contact surfaces while Liu et al [7] focused on conditioning on different types of audio. To be more effective in utilizing available audio datasets, DarkGAN [8] uses knowledge distillation (KD) to extract categorical information from large pre-trained classification models [9]. The classification model is able to output the associated probabilities (termed as soft labels) for each class, which could be utilized as conditioning information to guide the

sound synthesis. Although such approaches achieved higher audio qualities without requiring large amounts of labelled datasets, the discrete nature of the conditioning method still poses drawbacks, including limited expressiveness [10, 11], lack of continuity [12], and failure to capture the hierarchical semantic relationships between classes [13]. Conditioning on text embeddings, on the other hand, provides a simple yet effective control with human languages. AudioLDM [14] for example, is capable of synthesizing intricate audio waveforms agnostic to sound types given input texts by training in a self-supervised fashion which requires less label data. Nevertheless, text-to-audio synthesis models still require large amounts of audio datasets with descriptive labels in order to achieve reasonable performance. Additionally, texts themselves are also limited by the expressiveness in describing the subtle differences among different sounds.

Apart from conditioning on discrete labels, it is possible to use continuous vectors [15] as conditioning information, but may require specific training setup. Regression labels (continuous, such as ages and angles in image) provide a smooth and expressive representation that can help capture hierarchical [16] and nuanced variations across different classes [17]. CcGAN [16] achieves this by assigning weights using a Gaussian kernel based on the distance from the target label, allowing the model to handle sparse data more effectively. The proposed hard vicinal and soft vicinal discriminator loss allows the generative model to smooth the transition across continuous labels. It has been demonstrated that by introducing uncertainty, label smoothing [18] helps generative models improve their training stability and robustness. However, in the domain of audio, it is generally very difficult to obtain regression labels, and the number of high-quality datasets for sound effects pales in comparison with images. This makes applying a similar conditioning method to sound effects modelling difficult.

In this research, we study sound effects modelling under a GAN architecture by conditioning our generator on probabilistic soft labels. Different from CcGAN [16], we wish to blur the boundaries between discrete classes without requiring regression labels. To achieve this, we introduce an implicit conditioning method by manipulating discrete labels into continuous probabilistic vectors as conditioning. In Section 2, we elaborate on the proposed method by integrating it into a WGAN [19] model for Mel-spectrogram generation. In Section3 and Section4, we discuss the training details and the experiments as well as the evaluation metrics we used. We show the control effectiveness of the soft-labelling approach and propose two metrics to understand and evaluate the controllability of our conditioning approach using a pre-trained audio classifier. The results are shown in Section 5.

< **73** >

2. METHODOLOGY

2.1. Proposed conditioning method

We start by formulating our research goal. For a generator model, G, whose task is to output a two-dimensional Mel-spectrogram, M, we wish to condition the model on a continuous vector, C, which encodes the acoustic attributes of input sounds across different categories. To this end, we replace the one-hot vectors with a random variable sampled from a Gaussian distribution. Given a conditional vector C_s, sampled from a Gaussian distribution parameterized by mean and variance, our modelling objective is to estimate the conditional probability distribution $p(x|C_s)$ of the target data x. This can be formulated as:

$$p(x|C_s) = \int p(x|z, C_s)\, p(z|C_s)\, dz \qquad (1)$$

where:

- x represents the target data we aim to model,

- C_s is the conditioning vector, providing context or attributes that the target data x should adhere to, and
  $C_s \sim \mathcal{N}(\mu, \sigma^2)$, where μ and σ are learned from a classifier.

- z is a latent variable capturing aspects of x not specified by C_s,

- $p(z|C_s)$ represents the distribution of latent variables conditioned on C_s,

- $p(x|z, C_s)$ denotes the likelihood of x given both z and C_s.

Specifically, we first employ a CNN-based encoder classifier as shown in Figure 1 to extract acoustical information from M. The encoder classifier learns the spectral-temporal information and is expected to output class probabilities of the inputs. Instead of using the probabilities directly as conditioning labels, we wish to obtain a control space that allows us to interpolate the sound characteristics smoothly. To this end, we transform the output logits from the encoder classifier into mean and variance variables, μ and σ^2, respectively, which are used for re-parameterization to enable gradient flow. Similar to a VAE[20], we sample a vector from standard Gaussian distribution as $\epsilon \sim \mathcal{N}(0, \mathbf{I})$ to obtain the sampled vector $C_s = \mu + \epsilon \cdot \sigma^2$. By introducing uncertainty and noise through the sampling process, we wish to construct a more continuous space of categorical information among different classes of sounds. Instead of relying on a simple concatenation process, we integrate the conditioning information via a feature-wise linear modulation (FiLM) [21] operation that is applied to both C_s and z. FiLM applies an affine transformation to the intermediate features of a neural network using scaling and shifting operations, mathematically represented as $FiLM(z, C_s) = f_\gamma(C_s) \cdot z + f_\beta(C_s)$, where f_γ and f_β are neural networks (typically MLPs) that map the conditioning input C_s to the latent space dimension. This technique allows for precise control over how external information influences network activation values, leading to accurate and context-aware outputs.

2.2. Model architecture

To demonstrate how our conditioning method performs, we integrate our conditioning method into a GAN network for Mel-spectrogram generation. Mel-spectrogram correlates well with human perception and has received much attention in neural audio

synthesis [22, 23]. It serves as an interpretable intermediate representation that could be effectively learned with small amounts of data. With the advancement of neural vocoders [2, 24], we can usually get decent audio reconstructions from Mel-spectrograms. We denote our model as ICGAN (implicit conditioning GAN) with the entire model architecture as shown in Figure 1. In addition to conditioning the generator on the class labels, we also condition it on the extracted amplitude envelope from the Mel-spectrograms. Amplitude envelopes serve as a regulator that guides the generator to synthesize spectrograms with similar envelopes. This enables us to synthesize audio in a frame-level manner by inputting a guiding envelope during the inference. Additionally, with the help of amplitude information, the generator model may achieve higher synthesis quality, which is discussed in Section 5. To obtain the amplitude envelope from the Mel-spectrograms, we sum and average the power amplitude across all frequency bins to get the frame-level amplitude envelope A:

$$A(t) = \frac{1}{F} \sum_{f=1}^{F} P(f, t)\,, \qquad (2)$$

where $P(f, t)$ is the power at frequency bin f and time frame t for all frequency bins $f \in [1, F]$. In this way, the generator is guided by the amplitude information as well as the class-conditioning vectors. To synthesize the 2D spectrograms, we use the same RNN-based generator to model Mel-spectrograms as [25]. The generator is a three-layer GRU unit with an internal size of 512 followed by a linear layer. It sequentially transforms the input information to Mel-spectrograms with the same sequence length. As for the encoder classifier and the discriminator, we adapt a simple LeNet-5 [26] as shown in Figure 1. Both the encoder classifier and discriminator models employ 2D CNN layers to extract spatial information from the input spectrograms. Once the model is trained, we transform the generated Mel spectrogram to audio waveforms using a pre-trained neural vocoder [24]. The HifiGAN vocoder was trained on the Audioset[27] dataset, which contains millions of labelled sound events. It is expected to be able to convert Mel-spectrograms back to audio waveforms of agnostic to the audio source. Our complete model implementation can be accessed at the companion website[1].

2.3. Loss functions

Instead of providing explicit labels as input into the generator, we employ continuous categorical labels as conditioning vectors learned from the classifier. To facilitate the learning of the conditioning vectors, we force the re-parameterized Gaussian distribution to be similar to a Gaussian distribution formulated based on discrete class labels with a ground-truth mean and variance pair (refer to Fig. 1). We use the Kullback-Leibler divergence (KLD) to measure the similarity of the two distributions. We compute a regularization loss L_{reg} that incorporates the KLD using the one-hot vector label L, as well as the mean μ_n and variance σ_n^2 vectors obtained from the encoder classifier:

$$\mathcal{L}_{reg} = -\frac{1}{2} \sum_{n=1}^{N} \left(1 + \log(\sigma_n^2) - (\mu_n - L_n)^2 - \sigma_n^2\right), \qquad (3)$$

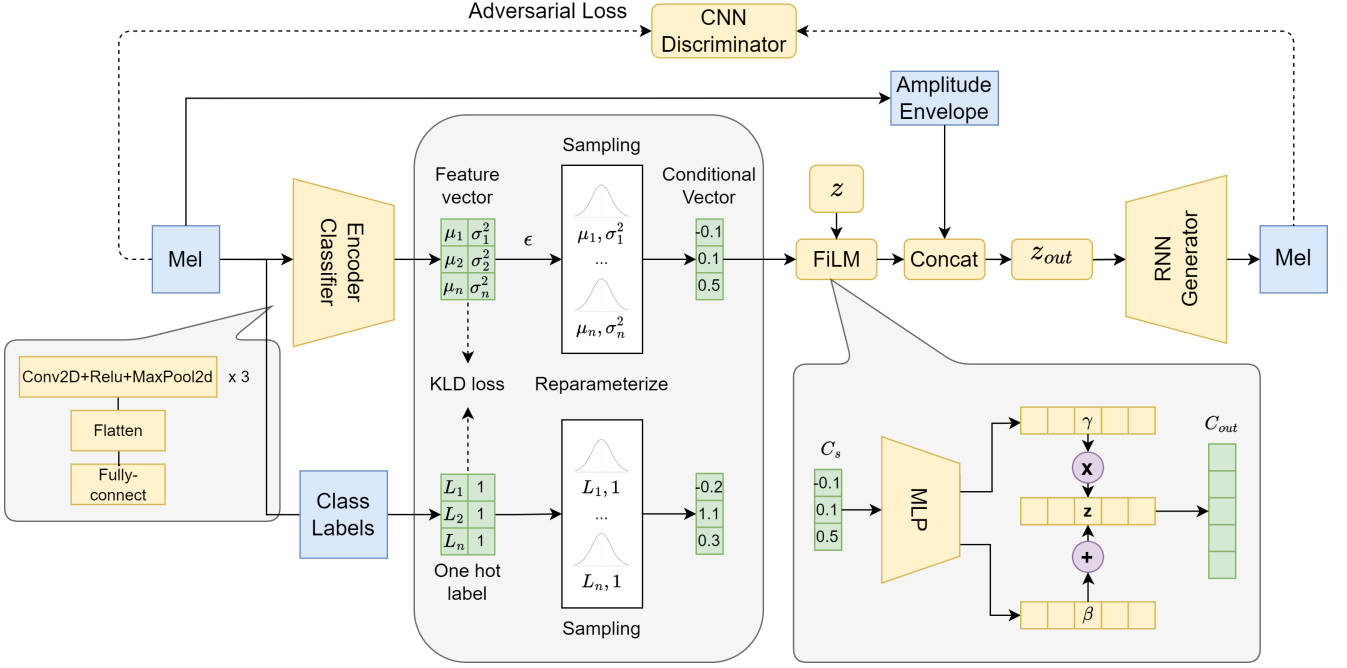
[1] https://github.com/Reinliu/ICGAN

Figure 1: Proposed model architecture. The model is composed of a CNN encoder classifier, an RNN generator, and a CNN discriminator. Yellow boxes are part of the neural networks. Blue boxes indicate training inputs and outputs, while greens represent explicit variables.

where N is the dimension of the label space, which corresponds to the total number of classes, and one hot values L_n are used to indicate the target distribution mean. Apart from the regularization loss, we adopt the Wasserstein GAN (WGAN) [19] loss framework for the discriminator, mirroring its proven approach to evaluate the discrepancy between real and generated data distributions. The generator loss is shown below:

$$L_G = -\mathbb{E}_{\tilde{x}\sim\mathbb{P}_g}[D(\tilde{x})] + \mathcal{L}_{reg} \qquad (4)$$

where $\mathbb{E}_{\tilde{x}\sim\mathbb{P}_g}[\cdot]$ denotes the expectation over samples $\tilde{x}$ generated by the generator's distribution $\mathbb{P}_g$ and $D(\tilde{x})$ represents the discriminator's (critic's) score for the generated samples. Notice that our generator loss also includes the regularization loss as illustrated above. In this way, the generator learns to accommodate changes associated with the classifier updating its output. This allows the generator and encoder to update their weights simultaneously within each batch, helping to stabilize the training and make the results more reliable.

With regard to the discriminator, we enhance training stability and enforce the Lipschitz constraint essential for optimal discriminator behavior by integrating the gradient penalty mechanism that is a hallmark of the WGAN-GP model [28] as shown below:

$$L_D = \mathbb{E}_{\tilde{x}\sim\mathbb{P}_g}[D(\tilde{x})] - \mathbb{E}_{x\sim\mathbb{P}_r}[D(x)]$$
$$+ \lambda \cdot \mathbb{E}_{\hat{x}\sim\mathbb{P}_{\hat{x}}}\left[(\|\nabla_{\hat{x}}D(\hat{x})\|_2 - 1)^2\right] \qquad (5)$$

- $\mathbb{E}_{x\sim\mathbb{P}_r}[D(x)]$ is the average score that the discriminator gives to real samples.

- $\lambda \cdot \mathbb{E}_{\hat{x}\sim\mathbb{P}_{\hat{x}}}\left[(\|\nabla_{\hat{x}}D(\hat{x})\|_2 - 1)^2\right]$ represents the gradient penalty. This term enforces a Lipschitz constraint and is

crucial for the model's stability. The penalty is applied to the gradients of the discriminator's scores with respect to interpolated samples between real and generated samples, pushing these gradients to have a norm of 1. We set $\lambda = 10$ in our experiments.

3. EXPERIMENTS

3.1. Dataset

As we are mainly interested in modelling sound effects, we meticulously construct an impact-sound-based audio dataset with sounds selected from the Boom Library[2] and BBC Sound Library [29]. Because of copyright issues, these datasets are not open to the public. Our dataset consists of three kinds of impact sounds: footsteps, gunshots, and hits. Each of these sound types are also comprised of several classes dependent on the labelling of such sounds. For example, footsteps contain 'Concrete', 'Gravel', 'Leaves', etc., while hits contain 'Punch face', 'Hit bag', 'Hit bone', etc. There are 6 subcategories of footstep sounds, 10 subcategories of gunshot sounds, and 9 categories of hit sounds. Each of the subcategories contains 100-200 sounds. In total, there are 4817 sounds and we split our dataset into 4335 (90%) for training and 482 (10%) for testing. Each of the sounds is sampled at 16 kHz and has a length of 4 seconds. We pre-process each sound to extract its Mel-spectrogram with a hop length of 64, and FFT size of 1024. All of the Mel-spectrograms have a length of 400 frames and 64 frequency bins. This setting corresponds to a pre-trained HiFiGAN vocoder [14] trained on the Audioset [27] dataset, allowing us to easily transform the Mel-spectrograms back to audio waveforms.

---

[2]https://www.boomlibrary.com/

<   **75**   >

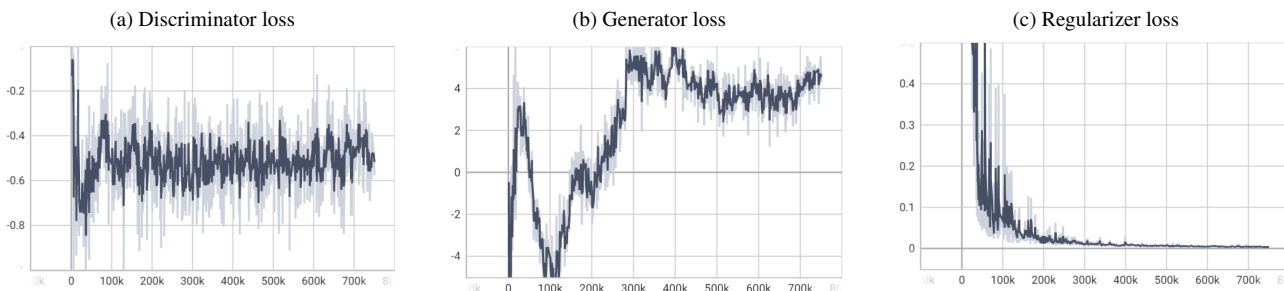(a) Discriminator loss   (b) Generator loss   (c) Regularizer loss

Figure 2: Visualization of the loss trend in our ICGAN model. We generally see a convergence after 300k iterations when all three losses were shown stable. The regularizer loss quickly converges while discriminator and generator loss oscillates around -0.5 and 4 respectively.

3.2. Training

We trained our model with the aforementioned datasets of three categories of sounds (footstep, gunshot, hits) for 10,000 epochs on an RTX 3080. Each model is conditioned with the variations within each category (eg. concrete, gravel, snow for the footstep category). We use an Adam optimizer [30] with a learning rate of $l = 0.0001$, $\beta_1 = 0.5$, $\beta_2 = 0.999$. During training, the discriminator loss stabilizes between -0.4 and -0.7 after about 100 k iterations indicating convergence. The regularization loss converges more slowly and requires approximately 300 k iterations as shown in Figure 2. As the regularization loss is added to the generator loss, the generator loss oscillates until the regularization loss converges after the 300 k iterations. Eventually, after the regularization loss stabilizes, the generator loss converges to a value of around four.

In addition to the in-class sounds (sounds within the same category), we briefly test the inference capability of our model on cross-class sounds. Therefore, we further trained our model on a sound effect dataset DCASE2023 [31] and condition it on the 7 categories such as motor engine, dog barks, and rain. We show the performance of the conditioning method in Section 5.

4. EVALUATION METHODS

4.1. Interpolation between two classes

Once the model is fully trained, we can abandon both the encoder and the discriminator. Instead of extracting information from a sound's Mel spectrogram, we can define an arbitrary conditioning vector C_s. Entries in the C_s vector close to unity indicate strong correspondence to a particular class within the sound type and values close to zero indicate no correspondence. In order to explore the influence of the conditioning vector, we interpolate 100 points between two target classes within a sound type. In other words, the values for one class increase from zero to one, while the values for the other class decrease from one to zero. In order to evaluate the "class" of the sound we use a separate pre-trained classifier that is unrelated to the sound generation (described in the next section). In Figure 3, we show four example pairs for each of the three sound categories. Each sound category was trained using a separate model with its own dataset. We randomly picked two classes within each category (eg, Footstep: concrete and gravel). Considering Figure 3, we generally see smooth transitions from one class

to another. Nonetheless, the sharpness of the transition seems to vary as does the completeness of the transition.

4.2. Proposed metric for control effectiveness

To measure the effectiveness of our conditioning method, we incorporate a pre-trained classifier that indicates the probability for each class of sound. This essentially links the conditioning information with the generated output. To this end, we use PANNs [9], a large-scale audio classification model trained on vast amounts of sounds [27] with 14 CNN layers. It was reported to achieve great acoustic event detection and audio tagging performance agnostic to sound types. Based on the pre-trained PANNs model, we fine-tuned and trained it on our dataset until it converged and reached an evaluation score of 79.5%.

We propose two evaluation metrics to evaluate our conditioning method: Maximum separation distance (MSD) and range of effective control (REC). The MSD is designed to quantify the breadth of the model's conditioning space. It does so by measuring the maximal extent to which the model's output can be varied by adjusting the conditioning vector to the limits indicating only one of the two target classes. In other words, ideally the conditioning vector can shift the generated sound completely from one class to the other. Practically, it can be incomplete. We measure the MSD score as the average of the total range of the softmax probability score for each of the two target classes:

$$\text{MSD} = (P(j)_{max} - P(j)_{min} + P(k)_{max} - P(k)_{min})/2, \quad (6)$$

where j, k are two interpolation variables with a range varying from zero to one.

The range of effective control (REC) is a metric that focuses on the practical operational range of the conditioning vector to change the model's output. It is calculated by determining the interval within which changes to the conditioning vector lead to significant changes in the output. More specifically, in Figure 3, we see the curves can saturate as the interpolation values are varied towards the ends of the interpolation range. This is to say, the slope of the curves can become flat towards the ends. Thus, we can define a threshold value for the absolute value or magnitude of the slope of the curve. If the magnitude of the slope is too small, less than some threshold, θ, we say the response has saturated. We
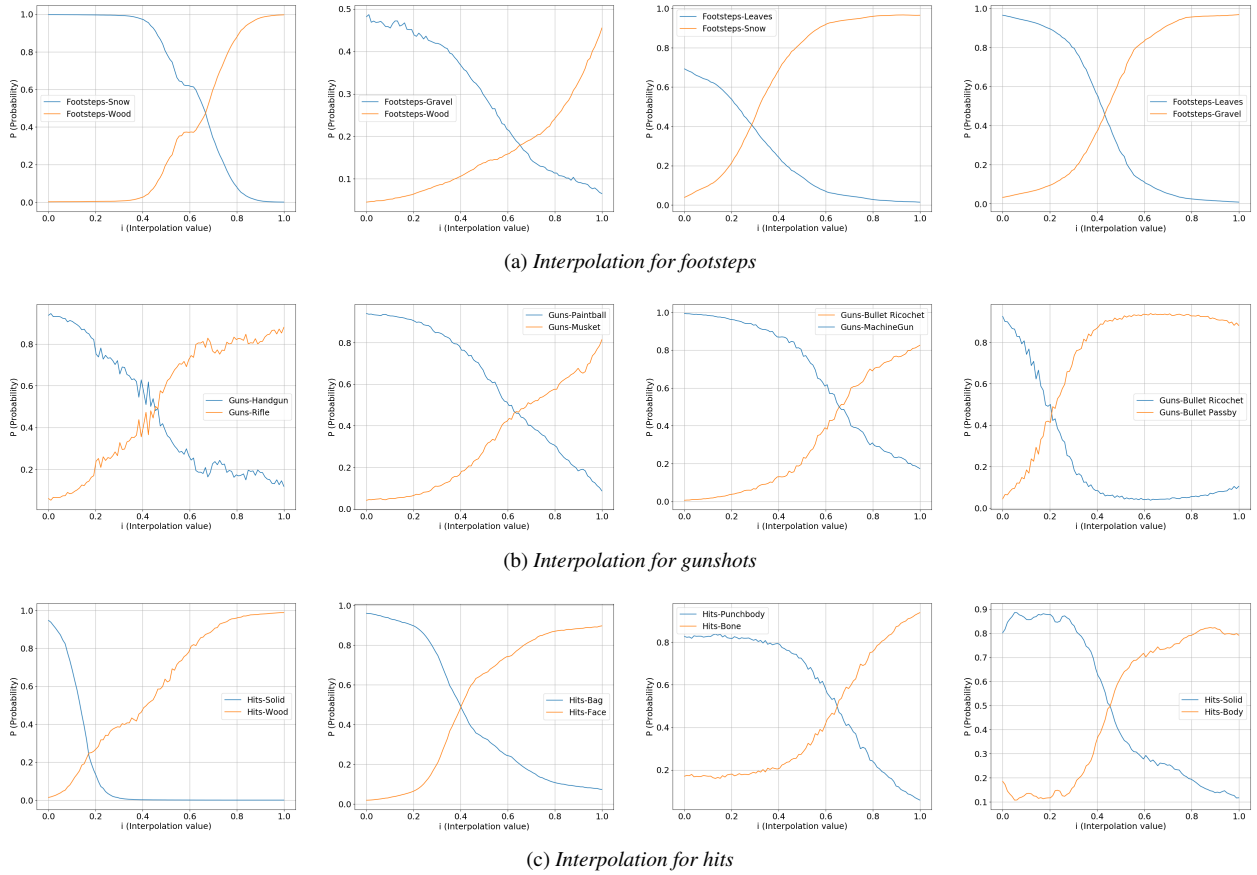
<   **76**   >

(a) *Interpolation for footsteps*

(b) *Interpolation for gunshots*

(c) *Interpolation for hits*

Figure 3: *Effect of interpolating points in the conditioning space. We randomly selected four combinations of interpolation between two classes in each sound category. The horizontal axis represents the interpolation points between two targets, with one increasing from 0 to 1 and the other from 1 to 0. The vertical axis represents the probability for that class returned by a trained classifier.*

define the REC score as shown in equation 7, $i \in [0, 1]$.

$$\text{REC} = i_{\text{end}} - i_{\text{start}} \qquad (7)$$

where:

- i_{start} is the first interpolation value where $\left| \frac{\Delta P(i)}{\Delta i} \right| > \theta$,

- i_{end} is the last interpolation value within the specified range where $\left| \frac{\Delta P(i)}{\Delta i} \right| > \theta$.

The REC score measures the segment of the output response curve for which the rate of change in the class probability is greater than the threshold value, θ. We compute the REC score for all pairs of classes under consideration, to explore how effectively the interpolation between classes controls the timbre attributes of the sounds.

4.3. Evaluation setup

In order to evaluate the synthesis performance of our model, we need to compare sounds generated by the model with real sounds. Therefore, we use our test dataset (refer to Section 3.1) to provide reference sounds. We extract the amplitude envelopes of the reference sounds and use a conditioning vector obtained by sampling a Gaussian model with a one-hot ground truth class label vector corresponding to the class of the reference sound as the mean for the

Gaussian model. We provide the amplitude envelope and conditioning vector as inputs to our ICGAN model and thus obtain one synthesized sound for each reference sound.

We compare our ICGAN model with LTS (Latent timbre synthesis) [32], which enables smooth transition from one sound to another by interpolating the VAE latent space. We denote ICGAN as the test in which we interpolated only conditioning vectors. Since LTS can only interpolate the latent space from one target sound to another and it does not take in any amplitude information as conditioning, for a fair comparison, we interpolated the amplitudes using ICGAN between two target sounds in the same way as LTS and named it ICGAN-w (with amplitude interpolation). For ablation study, we have also retrained our model by completely removing the amplitude envelope from the conditioning and named this ICGAN-n (no amplitude information). Finally, We added a comparison with the classical conditioning method as [6], namely, by inputting one-hot representations of the discrete labels into the generator and discriminator networks. We denote this model as conditional GAN (CGAN). Please note that we also interpolated values in the conditioning space of CGAN even though during training it was only accepting one-hot vectors. The reason for this is to test whether the model is able to extrapolate smooth transitions between discrete values. All results are shown in Table 1.

<   **77**   >

We evaluate the quality of the synthesis by comparing the similarity between the synthesized sounds and the reference sounds. We select a range of evaluation metrics including Frechet Audio Distance (FAD) [33], Frechet Inception Distance (FID) [34], and Log-Spectral Distance (LSD). Both FAD and FID are common metrics for evaluating the quality of generated data (audio and images) in generative models. They rely on pre-trained classifiers to extract features from both real and generated samples. The distance scores are then reported by computing the mean and covariance of these feature sets. This statistical approach captures not just the sound quality, but also the diversity within the datasets. For FAD, we use the VGGish [35] classifier with a sample rate of 16kHz, and for FID, we employ the InceptionV3 [36] classifier. Additionally, we compute the pairwise log spectral distance between the generated Mel-spectrogram and the reference Mel-spectrogram. We report the final LSD as the average score from each category.

5. RESULTS

Dataset	Model	Control		Audio Quality		
		MSD↑	REC↑	FAD↓	FID↓	LSD↓
Footsteps	ICGAN	0.683	0.392	**2.85**	**64.61**	0.119
	ICGAN-w	0.807	**0.696**	/	/	/
	ICGAN-n	0.744	0.610	16.66	163.4	0.118
	CGAN	**0.851**	0.022	4.90	102.5	**0.086**
	LTS	0.759	0.667	8.86	145.6	0.125
Guns	ICGAN	0.637	0.579	**2.64**	**60.44**	0.235
	ICGAN-w	0.733	**0.721**	/	/	/
	ICGAN-n	0.722	0.629	7.09	177.4	0.197
	CGAN	**0.841**	0.018	2.88	76.43	**0.187**
	LTS	0.637	0.711	4.97	126.9	0.289
Hits	ICGAN	0.647	0.863	**1.27**	60.58	0.136
	ICGAN-w	0.795	0.903	/	/	/
	ICGAN-n	0.773	0.807	5.19	110.6	0.146
	CGAN	**0.911**	0.011	2.43	95.57	**0.124**
	LTS	0.677	**0.936**	2.93	97.87	0.176
DCASE	ICGAN	0.509	0.210	**6.956**	**79.60**	0.272
	ICGAN-w	0.727	**0.569**	/	/	/
	ICGAN-n	0.641	0.513	16.49	185.5	0.292
	CGAN	**0.818**	0.026	10.21	108.6	**0.246**
	LTS	0.693	0.512	7.67	131.4	0.321

Table 1: *Comparison of model synthesis performance and control affordances. ICGAN is the proposed model as illustrated in 1. ICGAN refers to our proposed method by interpolating only conditioning space. ICGAN-w is the same ICGAN model but we also interpolated the extracted amplitudes from two target sounds. Therefore the audio quality is the same as ICGAN. ICGAN-n is a re-trained model which completely removes amplitude information. CGAN denotes the conditional GAN using concatenation. LTS refers to 'Latent Timbre Synthesis' [32] which does latent space interpolation between two target sounds. Up-arrows indicate the higher the score the better, and vice versa.*

5.1. Conditioning performance

Since we have several classes of sounds for each sound type or category, there are several possible pairs of classes with which to interpolate between. We refer to these pairs of classes as interpolation pairs. For the purposes of showing results, we average the results across a sample of interpolation pairs. Average results obtained for the MSD and REC are shown in Table 1. When computing the REC, we use a threshold value given by $\theta = 1e^{-2}$. For the MSD score, we found that CGAN achieves the highest score. This makes sense because it was hard conditioned on discrete values between two ends, namely 0 and 1. The classification could successfully tell the class on the two ends. On the contrary, REC reflects that CGAN gets much lower scores because the transition between two ends is rather discrete and drastic. Our ICGAN-w, which we interpolated the amplitudes together with the conditioning space, achieves the highest REC overall, and performs second best in MSD. This shows the potential of our method, as it no longer requires interpolating in the uninterpretable latent space to shift from one sound to another. Although by interpolating the conditioning space only did not yield the best performance, we nonetheless found it useful as it provides a straightforward and interpretable control over the sound categories, which could not be easily achieved using other models relying on extensive labels.

Additionally, the results for the REC score indicate that the effective range of conditioning varies across different sound types. We found the Hits category obtains the highest REC overall, whereas DCASE dataset attains the lowest. We hypothesize that this can be attributed to the degree of variance within different sound classes. To test this, we extract and average the MFCC features for each sub-category, then compute and normalize the pairwise Euclidean distances between categories, and finally get the averaged similarity scores for each dataset. We found that our Hits dataset did have higher variances among different sub-categories, with the normalized similarity score 0.568, which is higher than Guns 0.451, Footsteps 0.316 and DCASE 0.342. The high degree of variations among different sub-categories could enable the encoder classifier to efficiently learn the intricate differences and thereby output meaningful and smooth values, which likely results in the smooth transitions different classes. However, how exactly dataset variations could contribute to different control affordances needs further investigation.

5.2. Synthesis performance

In Table 1, we found that our model excels in statistical similarity metrics reflected in FAD and FID scores, indicating a high degree of performance in the quality and diversity of the generated data. However, from the log spectral distances which compare the pairwise signals directly, we realized the explicit conditioning method achieves higher accuracy in the spectrogram reconstruction. This is because our implicit conditioning method introduces uncertainty through sampling, resulting in lower accuracy for reconstructing signals with definitive labels. On the contrary, this also brings higher sample diversities reflected in the statistical measures. We also noticed an increase in dissimilarity between the target and generated data trained by the DCASE Foley dataset. It becomes more difficult for both models to reconstruct the signals when the conditioned data vary more. We have also observed that after removing the extracted amplitudes, the audio quality was reduced significantly. We believe the amplitude information provides an important guidance for the generator for synthesis.

In Figure 4, we show the synthesized Mel spectrograms corresponding to different classes within the Foodsteps sound category: concrete, leaves, and gravel. Note that all three sounds were generated using the same amplitude envelope, and the spectrograms

< **78** >

(a) Footstep-concrete
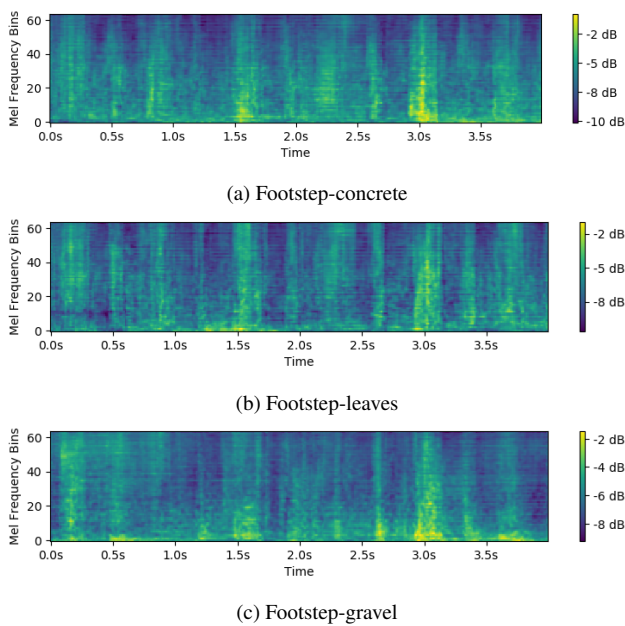

(b) Footstep-leaves

(c) Footstep-gravel

Figure 4: *Generated in-class sounds. The models are trained in the same footsteps category with different variations denoted by its property.*

exhibited similar temporal changes. The amplitude conditioning helps the model maintain the overall contours of the spectrogram while offering controllability varying timbral characteristics. In addition to conditioning on in-class sounds, our model could also be trained to synthesize out-of-domain sounds. We show the results of cross-domain sounds trained with DCASE dataset in our accompaniment website [3].

6. DISCUSSION AND CONCLUSION

In this research, we proposed a novel conditioning method that translates discrete labels into a continuous probabilistic labelling space. We have applied this implicit conditioning method to the domain of neural audio synthesis and explored model performance in terms of the effectiveness of the conditioning vector in controlling the sound class and the audio quality of the generated sounds. We primarily focused on in-domain variations in sound effects, but briefly consider cross-domain sound effects. Our approach shows promise when compared with conventional conditioning methods, but requires more development. We show that the conditioning vector can effectively interpolate between sound classes and enable a smooth transition in the timbre space without relying on excessive labels. We also observed that the control effectiveness seems to relate to the variations of a dataset, but we will leave this question to future investigation. We hope our proposed conditioning approach sparks more creativity in the sound design and synthesis field and allows for interesting future explorations in the continuous conditioning space, such as interpolation involving more than two dimensions and also mixing the features of various target sounds. Future work could include incorporating a training mechanism that encourages a more consistent and controllable

conditioning space.

7. REFERENCES

[1] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. W. Senior, and K. Kavukcuoglu, "Wavenet: A generative model for raw audio," in *The 9th ISCA Speech Synthesis Workshop*, Sunnyvale, CA, USA, September 2016, p. 125, ISCA.

[2] Kundan Kumar, Rithesh Kumar, Thibault de Boissière, Lucas Gestin, Wei Zhen Teoh, Jose M. R. Sotelo, Alexandre de Brébisson, Yoshua Bengio, and Aaron C. Courville, "Melgan: Generative adversarial networks for conditional waveform synthesis," in *Neural Information Processing Systems*, 2019.

[3] Antoine Caillon and Philippe Esling, "Rave: A variational autoencoder for fast and high-quality neural audio synthesis," 2021.

[4] Jesse Engel, Kumar Krishna Agrawal, Shuo Chen, Ishaan Gulrajani, Chris Donahue, and Adam Roberts, "Gansynth: Adversarial neural audio synthesis," in *International Conference on Learning Representations*, 2019, vol. abs/1902.08710.

[5] Adrián Barahona-Ríos and Sandra Pauletto, "Synthesising knocking sound effects using conditional wavegan," in *17th Sound and Music Computing Conference, Online*, 2020.

[6] Marco Comunità, Huy Phan, and Joshua D. Reiss, "Neural synthesis of footsteps sound effects with generative adversarial networks," in *Audio Engineering Society Convention 152*, Online and In-Person, May 2022, 2022, Audio Engineering Society, Convention Paper 10583.

[7] Yunyi Liu and Craig Jin, "Conditional sound effects generation with regularized wgan," in *20th Sound and Music Computing Conference, Stockholm*, 2023.

[8] Javier Nistal, Stefan Lattner, and Gaël Richard, "Darkgan: Exploiting knowledge distillation for comprehensible audio synthesis with gans," in *International Society for Music Information Retrieval Conference*, 2021.

[9] Qiuqiang Kong, Yin Cao, Turab Iqbal, Yuxuan Wang, Wenwu Wang, and Mark D. Plumbley, "Panns: Large-scale pretrained audio neural networks for audio pattern recognition," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 28, pp. 2880–2894, 2019.

[10] Aäron van den Oord, Nal Kalchbrenner, Oriol Vinyals, Lasse Espeholt, Alex Graves, and Koray Kavukcuoglu, "Conditional image generation with pixelcnn decoders," in *Proceedings of the 30th International Conference on Neural Information Processing Systems*, Red Hook, NY, USA, 2016, NIPS'16, p. 4797–4805, Curran Associates Inc.

[11] Zhiting Hu, Zichao Yang, Ruslan Salakhutdinov, Xiaodan Liang, Lianhui Qin, Haoye Dong, and Eric P. Xing, "Deep generative models with learnable knowledge constraints," in *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, Red Hook, NY, USA, 2018, NIPS'18, p. 10522–10533, Curran Associates Inc.

[12] Eszter Vértes and Maneesh Sahani, "Flexible and accurate inference and learning for deep generative models," in *Neural Information Processing Systems*, 2018.

[3] https://reinliu.github.io/ICGAN-Paper/

< **79** >

[13] Yunchuan Guan, Yu Liu, Ke Zhou, and Junyuan Huang, "Hierarchical meta-learning with hyper-tasks for few-shot learning," in *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, New York, NY, USA, 2023, CIKM '23, p. 587–596, Association for Computing Machinery.

[14] Haohe Liu, Zehua Chen, Yiitan Yuan, Xinhao Mei, Xubo Liu, Danilo P. Mandic, Wenwu Wang, and MarkD . Plumbley, "Audioldm: Text-to-audio generation with latent diffusion models," in *International Conference on Machine Learning*, 2023.

[15] Heewoo Jun, Rewon Child, Mark Chen, John Schulman, Aditya Ramesh, Alec Radford, and Ilya Sutskever, "Distribution augmentation for generative modeling," in *Proceedings of the 37th International Conference on Machine Learning*. 2020, ICML'20, JMLR.org.

[16] Xin Ding, Yongwei Wang, Zuheng Xu, William J. Welch, and Z. Jane Wang, "Ccgan: Continuous conditional generative adversarial networks for image generation," *CoRR*, vol. abs/2011.07466, 2020.

[17] Amin Heyrani Nobari, Wei Chen, and Faez Ahmed, "Pcdgan: A continuous conditional diverse generative adversarial network for inverse design," in *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, New York, NY, USA, 2021, KDD '21, p. 606–616, Association for Computing Machinery.

[18] Chang-Bin Zhang, Peng-Tao Jiang, Qibin Hou, Yunchao Wei, Qi Han, Zhen Li, and Ming-Ming Cheng, "Delving deep into label smoothing," vol. 30, pp. 5984–5996, jan 2021.

[19] Martín Arjovsky, Soumith Chintala, and Léon Bottou, "Wasserstein generative adversarial networks," in *International Conference on Machine Learning*, 2017.

[20] Diederik P. Kingma and Max Welling, "Auto-encoding variational bayes," *International Conference on Learning Representations*, vol. abs/1312.6114, 2013.

[21] Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron Courville, "Film: visual reasoning with a general conditioning layer," in *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence*. 2018, AAAI'18/IAAI'18/EAAI'18, AAAI Press.

[22] Xubo Liu, Turab Iqbal, Jinzheng Zhao, Qiushi Huang, Mark D. Plumbley, and Wenwu Wang, "Conditional sound generation using neural discrete time-frequency representation learning," *2021 IEEE 31st International Workshop on Machine Learning for Signal Processing (MLSP)*, pp. 1–6, 2021.

[23] Adrian Barahona Rios and Tom Collins, "Specsingan: Sound effect variation synthesis using single-image gans," in *Proceedings of the Sound and Music Computing Conference*, 2022.

[24] Jungil Kong, Jaehyeon Kim, and Jaekyoung Bae, "Hifi-gan: generative adversarial networks for efficient and high fidelity speech synthesis," in *Proceedings of the 34th International Conference on Neural Information Processing Systems*, Red Hook, NY, USA, 2020, NIPS'20, Curran Associates Inc.

[25] Yunyi Liu and Craig Jin, "Impact on quality and diversity from integrating a reconstruction loss into neural audio synthesis," *Acoustical Society of America Journal*, vol. 154, pp. A99–A99, Oct. 2023.

[26] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[27] Jort F. Gemmeke, Daniel P. W. Ellis, Dylan Freedman, Aren Jansen, Wade Lawrence, R. Channing Moore, Manoj Plakal, and Marvin Ritter, "Audio set: An ontology and human-labeled dataset for audio events," in *Proc. IEEE ICASSP 2017*, New Orleans, LA, 2017.

[28] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville, "Improved training of wasserstein gans," *Advances in neural information processing systems*, vol. 30, 2017.

[29] Films for the Humanities & Sciences, *BBC Sound Effects Library*, Princeton, N.J., 1991, Sound effects library.

[30] Diederik P. Kingma and Jimmy Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, Yoshua Bengio and Yann LeCun, Eds., 2015.

[31] Keunwoo Choi, Jaekwon Im, Laurie Heller, Brian McFee, Keisuke Imoto, Yuki Okamoto, Mathieu Lagrange, and Shinosuke Takamichi, "Foley sound synthesis at the dcase 2023 challenge," *In arXiv e-prints: 2304.12521*, 2023.

[32] Kıvanç Tatar, Daniel Bisig, and Philippe Pasquier, "Latent timbre synthesis," *Neural Computing and Applications*, vol. 33, pp. 1–18, 01 2021.

[33] Kevin Kilgour, Mauricio Zuluaga, Dominik Roblek, and Matthew Sharifi, "Fréchet audio distance: A reference-free metric for evaluating music enhancement algorithms," in *Interspeech*, 2019.

[34] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter, "Gans trained by a two time-scale update rule converge to a local nash equilibrium," in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, Red Hook, NY, USA, 2017, NIPS'17, p. 6629–6640, Curran Associates Inc.

[35] Shawn Hershey, Sourish Chaudhuri, Daniel P. W. Ellis, Jort F. Gemmeke, Aren Jansen, R. Channing Moore, Manoj Plakal, Devin Platt, Rif A. Saurous, Bryan Seybold, Malcolm Slaney, Ron J. Weiss, and Kevin W. Wilson, "Cnn architectures for large-scale audio classification," *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 131–135, 2016.

[36] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna, "Rethinking the inception architecture for computer vision," *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2818–2826, 2015.

[37] Robin Rombach, A. Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer, "High-resolution image synthesis with latent diffusion models," *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10674–10685, 2021.

< **80** >