Preference-Driven Texture Modeling Through Interactive Generation and Search

Shihan Lu, Mianlun Zheng, Matthew C. Fontaine, Stefanos Nikolaidis, and Heather Culbertson

***Abstract*—Data-driven texture modeling and rendering has pushed the limit of realism in haptics. However, the lack of haptic texture databases, difficulties of model interpolation and expansion, and the complexity of real textures prevent data-driven methods from capturing a large variety of textures and from customizing models to suit specific output hardware or user needs. This work proposes an interactive texture generation and search framework driven by user input. We design a GAN-based texture model generator, which can create a wide range of texture models using Auto-Regressive processes. Our interactive texture search method, which we call "*preference-driven*", follows an evolutionary strategy given guidance from user's preferred feedback within a set of generated texture models. We implemented this framework on a 3D haptic device and conducted a two-phase user study to evaluate the efficiency and accuracy of our method for previously unmodeled textures. The results showed that by comparing the feel of real and generated virtual textures, users can follow an evolutionary process to efficiently find a virtual texture model that matched or exceeded the realism of a data-driven model. Furthermore, for 4 out of 5 real textures, $\geq$80% of the preference-driven models from participants were rated comparable to the data-driven models.**

***Index Terms*—Texture modeling, high-frequency vibrations, machine learning, haptic display**

I. Introduction

Generating realistic textures for virtual tool-surface interactions provides touch sensations in the increasingly popular pen-based digital design. One common approach to texture modeling and rendering uses data recorded from a physical interaction, the most common being vibrations produced when dragging a tool across the surface, to create models that can accurately reproduce the feeling of the surface [1], [2], [3], [4]. Previous work has shown that this data-driven modeling and rendering of tool-surface interactions can capture some key dimensions of the haptic perception [5]. However, fully bridging the gap from real to virtual remains a challenge. Additionally, there are several limitations that hinder the improvement towards large-scale texture generation and customization for output hardware and user needs.

There are countless textures in the real world, but it is time-consuming and laborious to manually record and model data from a large number of textures due to current data-driven procedures. Furthermore, the data recording can only be done by specific devices that are not accessible to most users. Textures for which it is challenging to record data through a tool are either excluded from the dataset or produce inaccurate models. For instance, during recording the tool can easily get stuck in soft surfaces like foam and carpet, or high friction can cause a jerky and noncontinuous motion path on the surface. Creating texture models for textured surfaces without recording technology would greatly release the constraints in the data collection step of the data-driven method.

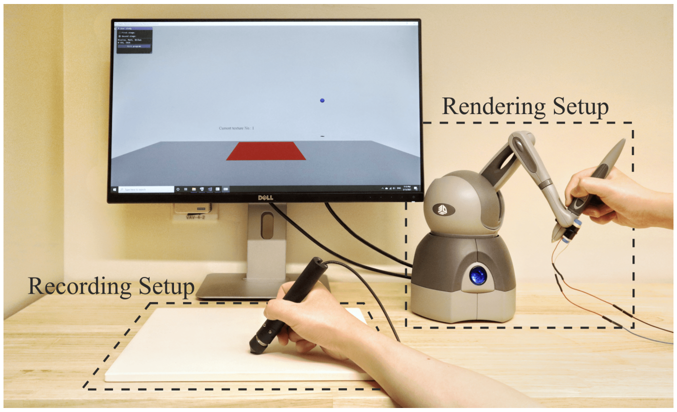

Fig. 1: Perception mismatch between texture recording to rendering due to actuation methods in tool-surface interactions.

The data-driven texture models from the widely-used Penn Haptic Texture Toolkit [6] are created from acceleration signals modeled as Auto-Regressive (AR) processes. Due to the high nonlinearity in the AR coefficient computation and the models' variation with both force and speed, it is not possible to find a simple relationship between AR coefficients and texture attributes. This intractable problem prevents us from creating new perceptually meaningful textures through direct interpolation or extrapolation of the existing texture models.

In data-driven methods, there exists an intrinsic gap in the transition from texture recording to texture rendering in terms of different actuation methods [7]. Data-driven methods have focused on minimizing the error between recorded and modelled signals with respect to certain components, such as the frequency spectrum [8]. However, there is no guarantee that simply minimizing signal errors will result in a more realistic haptic experience considering the fundamental differences between the way the vibrations are felt in natural interactions through a tool and after being output through an actuator attached to a tool (Fig. 1). This gap caused by the different actuation methods in recording and rendering has not been bridged and will be exaggerated by model errors.

To overcome the limits of data-driven modeling, we propose an interactive texture generation and search system, which is a two-stage process. The texture generation contains a generative adversarial network (GAN) for mapping the latent space into texture models. The texture search evolves the texture models generated by the GAN using an evolutionary algorithm given input about user's preference. We use the

The authors are with the Department of Computer Science, University of Southern California, Los Angeles, CA 90089, USA (email: {shihanlu, mianlunz, mfontain, nikolaid, hculbert}@usc.edu)

generative modeling to produce a bounded space of textures and the evolutionary algorithm to realize a controllable and interactive texture tuning process under the guidance of user's preference. We create a haptic user interface to evaluate our system's efficiency and accuracy of generating and searching new virtual textures that feel similar to the given real textures.

II. Background

With the increasing demand of realistic virtual textures in areas such as fashion design, online shopping, and surgical training, open-source datasets that collect haptic texture signals in tool-surface interactions have surfaced [6], [9], [10], [11]. These haptic texture databases greatly facilitate the burst of ideas in related areas of haptic research and further the communication between researchers from different areas.

Haptic texture modeling has shifted focus from physics-based simulation to data-driven methods, resulting in a significant increase in realism. Then Penn Haptic Texture Toolkit (HaTT) [6] was created to accelerate data-driven texture modeling and rendering research. Compared to physics-based simulation, data-driven methods treat the mechanism for transitioning between a physical interaction to haptic feedback as a black box, and directly map the inputs (interactive motion) to outputs (haptic feedback) under certain assumptions and constraints. This mapping process relies on pre-recorded haptic signals, which are often vibration. Culbertson et al. [5] and Shin et al. [3] used high-frequency vibration recordings to create haptic texture models using Auto-Regressive processes and frequency-decomposed neural networks, respectively.

The growth in popularity of personal and consumer haptic devices creates the need for haptic feedback that can be customized for both users and devices. However, existing vibrotactile-based texture datasets (HaTT [6]: 100 textures, LMT [10]: 108 textures) cannot fully satisfy this need for customization given the range of textures in the wild and the possibilities of output devices. Compared to vision, recording texture information usually requires more effort and specialized equipment, making it challenging to expand a texture dataset. To address this limitation, recent work has been looking into generative models of textures. Generative Adversarial Networks (GANs) have shown the potential of regenerating and fine-tuning realistic texture samples in vision [12]. Gan et al. [13] further combined generative adversarial training and perceptual feature regression for visual texture generation. For haptic textures, Ujitoko et al. [14] first proposed a method of generating acceleration signals from texture images or attributes via a conditional generative adversarial network (CGAN). However, this type of end-to-end CGAN model for generating the frequency spectrum of the acceleration signal from texture images cannot be used for real-time rendering of signals that match the user's motion due to computational constraints when transferring the frequency spectrum of acceleration to temporal acceleration signals. This interactivity between a user's motion and rendered signals has been shown to be critical to the realism of the virtual interaction [15].

Ujitoko et al. [16] further fine-tuned the intermediate acceleration signals by linearly sampling the latent vector of GAN model and showed the possibility of controllably directing the user's perception by the interpolation of GAN. However, this sampling was done in a monotonic direction between two pre-defined textures, excluding possibilities of tuning in other directions. Hassan et al. [17] introduced "Haptic Authoring" concept to synthesize new virtual textures by interpolating existing texture models based on the correlation with their affective descriptions. As the appearance of personalization for the vibrotactile patterns design [18], [19], researchers started to shift the focus from expert decisions to personal preferences. They used Interactive Evolutionary Computation (IEC) to optimize the vibrotactile properties when users interacted with the system. Compared to handcrafted universal vibrotactile patterns based on prior knowledge, interactive generation of vibration can effectively fit specific user and application.

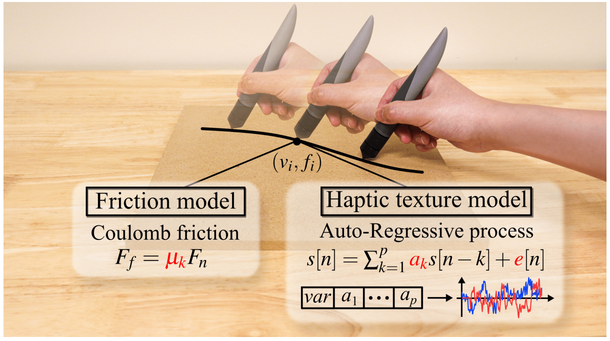

Fig. 2: Coulomb friction model and Auto-Regressive haptic texture model for surfaces in HaTT. Key parameters optimized by recorded data in the previous data-driven method and tuned by user's preference in this work are shown in red.

In this work, we combine the interpolation of GAN and the self-adaptation of Covariance Matrix Adaptation Evolution Strategy (CMA-ES) to realize the free but bounded texture search by user's preference during the interaction with a set of textures. CMA-ES is a *comparison-based* strategy, which enables us to ask users to order textures, instead of specifying absolute values. We use the ordering to direct the search towards textures that are perceived close to the real texture.

III. GAN-Based Texture Model Generator

In this section, we introduce our GAN-based texture model generator, including details on input processing and structuring for the HaTT dataset, and deep convolutional generative adversarial networks of haptic texture models. This GAN-based texture model generator provides the base for our interactive texture search, which is introduced in Section IV.

A. Input Processing and Structuring

We use the Penn HaTT database created by Culbertson et al. [6] as the base of our generative model design due to its wide range of textures and included texture models. HaTT collected and haptically modeled 100 textures in ten categories ranging from metal to carpet. Each texture contains a friction model and a haptic texture model including a set of AR models

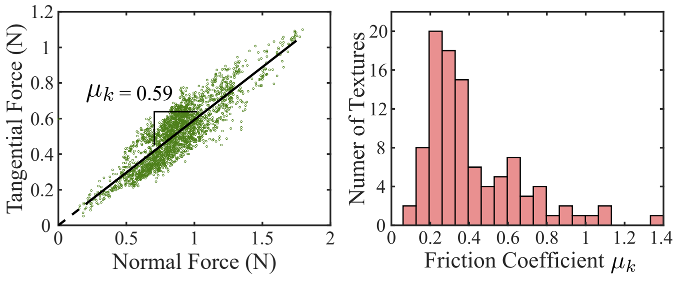

Fig. 3: (left) The Coulomb friction coefficient μ_k is computed by fitting a line for F_f vs F_n through the origin in the data-driven method; (right) Distribution of μ_k across HaTT

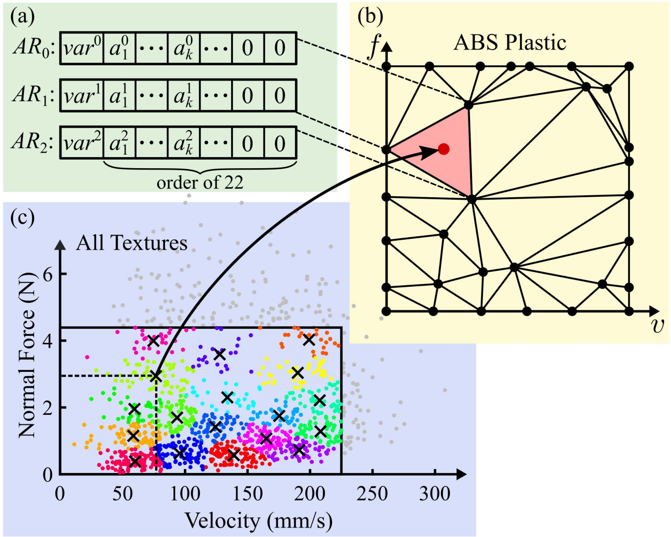
(a)
AR_0: var^0 a_1^0 ⋯ a_k^0 ⋯ 0 0
AR_1: var^1 a_1^1 ⋯ a_k^1 ⋯ 0 0
AR_2: var^2 a_1^2 ⋯ a_k^2 ⋯ 0 0
order of 22
(b)
f
ABS Plastic
v
(c)
All Textures
Normal Force (N)
0
2
4
6
0
50
100
150
200
250
300
Velocity (mm/s)

Fig. 4: Haptic texture model structuring: (a) Creating a uniform AR order for models of each force-speed entry by zero padding; (b) Creating a uniform number of AR models for each texture by interpolation via Delaunay triangulation. For ABS plastic, black dots are AR models in HaTT and the red dot is one of interpolated AR models; (c) K-means clustering for selecting a set of force-speed values for interpolation.

that cover the range of user's force and speed during the interaction, as shown in Fig. 2.

1) Friction Model: Friction is one of the key components affecting a human's perception when interacting with a texture [20]. Prior work has shown that friction has a direct impact on a human's judgement of slipperiness through a tool [21].

HaTT models the friction for the tooltip dragging across the surface as Coulomb friction, where $\mu_k = F_f/F_n$. The friction coefficient is modeled by finding the slope of the line fitting the recorded tangential F_f and normal F_n forces (Fig. 3). F_n and F_f are extracted from the recording based on the surface plane ($\hat{\alpha}_n$) and motion direction ($\hat{\alpha}_f$) calculated from tool position data. The friction model adds viscous damping around zero speed to increase stability during the rendering; the slope of the viscous damping is set as a constant for all textures.

2) Haptic Texture Model: The tool-interaction vibrations have been shown to be directly correlated with the texture's roughness [22]. For the haptic texture model, the vibration signal recorded at 10 kHz is first segmented into multiple stationary segments and then each segment is modeled as a piecewise Auto-Regressive (AR) process [5]:

$$s[n] = \sum_{k=1}^{p} a_k s[n-k] + e[n] \tag{1}$$

where p is the AR order, a_k is the kth AR coefficient, and e is the prediction error (Fig. 2).

An AR model is defined by a set of *AR coefficients* $(a_{1,2,\ldots,p})$ and the *variance of prediction errors* (var). Implicitly, the variance of the prediction errors encodes vibration amplitude and the AR coefficients encode vibration frequency. For a surface, each AR model from the segment is stored as a vertex in a Delaunay triangulation, labeled with the median speed v_i and force f_i of the segment. This triangulation is used for the model search and interpolation during the rendering for the surface, as shown in Fig. 4(b). All AR models in the triangulation constitute the *haptic texture model* of the surface.

a) Creating a Uniform AR Order: The recording is segmented into stationary segments before the modeling, and this segmentation determines the AR order (i.e., the number of AR coefficients) for each recording, resulting in inconsistent AR order (17∼22) across textures due to unconstrained interactions. To create a uniform data length that can be fed into the GAN for training, we pad zeros on the end of the AR coefficients up to the order of 22, as shown in Fig. 4(a). The zeros padded in the higher order of the AR process do not affect its performance according to the definition in Eq. (1).

b) Creating a Uniform Number of AR Models: After segmentation, each segment is labeled with the median speed and force during this period, and modeled as an AR process which is stored in the Delaunay triangulation (black dots in Fig. 4(b)). Due to the unconstrained nature of the recording, the number of stored AR models varies for each texture. However, our GAN-based model generator requires that all input texture models are made up of the same number of AR models. So, we interpolate the AR models to a smaller set of force-speed values using the Delaunay triangulation [2] to create a uniform number of AR models across textures. We select this set of force-speed values using K-means clustering on the force-speed entries of all texture models after removing outliers and boundary entries; the force-speed values are chosen as the centroid of each cluster (Fig. 4(c)). This clustering allows us to select force-speed values that both span the motion space well and accurately represent the amount of the user's time spent at that part of the motion space (i.e., the points are denser in the regions of the motion space most frequently used). The number of clusters is set at 18, which is the average number of AR models over all textures in HaTT. All interpolated AR models from a surface (red dot in Fig. 4(b)) form a haptic texture model used for the training later.

c) Input Augmentation: In this work, we use the Penn HaTT database, which consists of 100 textures with one model set for each. However, this amount of data is insufficient to train the network described in Section III-B, requiring us to augment the texture models. Above, we apply K-means clustering on all force-speed entries, generating 18 clusters, and use the clusters' centroids as force-speed values for

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TOH.2022.3173935, IEEE Transactions on Haptics

interpolation. We assume that AR models behave similarly but not exactly the same within a cluster of force-speed entries. We randomly select an entry from each cluster to constitute a new set of force-speed values for interpolation, labeling them with the centroids of those clusters. It ensures the independence between texture models due to the random selection in each cluster. We repeat this process 20 times for each texture, obtaining 2000 texture models in format of (*number of AR models*)×(*length of AR model*). It is valid to label the randomly selected entry from each cluster with its centroid due to our assumption on the similarity of haptic feedback from similar motions and the isotropic nature of all surfaces in HaTT.

d) Reflection Coefficients (RC): AR coefficients are notoriously sensitive to quantizing errors, since subtle disturbances can lead to the system instability [23]. We use RC as an alternative representation of AR coefficients because they can be easily converted to and from AR coefficients without information loss and are robust to quantizing errors [24]. They also have symmetric scales and fewer constraints on parameters for stability - when the absolute values of RC are less than 1, it implies a stable system. So, AR coefficients a_k with the order of 22 are converted to RC with 21 coefficients.

B. Generative Adversarial Networks of Texture Models

We propose to generate new haptic texture models from latent vectors by training a deep convolutional generative adversarial network (DCGAN) [25] considering its ability of learning intermediate representations of unlabelled dataset. We designed the network with minimal modifications on the standard DCGAN to show the generality of the framework and found that its simple structure was sufficient for our purposes; other generator designs may also qualify for the role in this preference-driven framework. The network design is shown in Fig. 5. Our DCGAN model consists of a group of multi layer perceptrons (MLP) as a generator G to map the latent space to texture models, and another group of MLPs as a discriminator D to judge whether a texture model is perceptually realistic by training on both real and generated texture models.

The training goal for D is to maximize the probability of correctly judging a haptic texture model from both real and generated texture models. In the meantime, we train G to minimize the probability that D judges the generated texture models as fake. So, G and D follow a minmax game to improve the perceptual realism of the generated texture models, with the loss function $L(G,D)$:

$$\begin{aligned}\min_G \max_D L(G,D) =& \mathbb{E}_{x\sim p_y(x)}[\log D(x)] \\ &+ \mathbb{E}_{z\sim p_z(z)}[\log(1-D(G(z)))]\end{aligned} \tag{2}$$

where z is the latent vector, $D(\mathbf{x})$ is the probability that $\mathbf{x}$ is a real texture model instead of a generated one, and $G(z)$ is the generated texture model mapped from the latent vector z. p_y is the distribution on a set of real texture models y and p_z is the prior distribution on latent vectors.

After training, we expect that the generator G can generate perceptually realistic texture models by mapping from the latent vectors z and that the outputs $G(z)$ can cover a distribution similar to the real texture models. Both properties are evaluated in Section III-D. Although the outputs of the generator have a similar distribution as real texture models, the generator can produce many different models that do not exist in the original dataset, considering the large amount of outputs compared to the limited number of textures in the dataset.

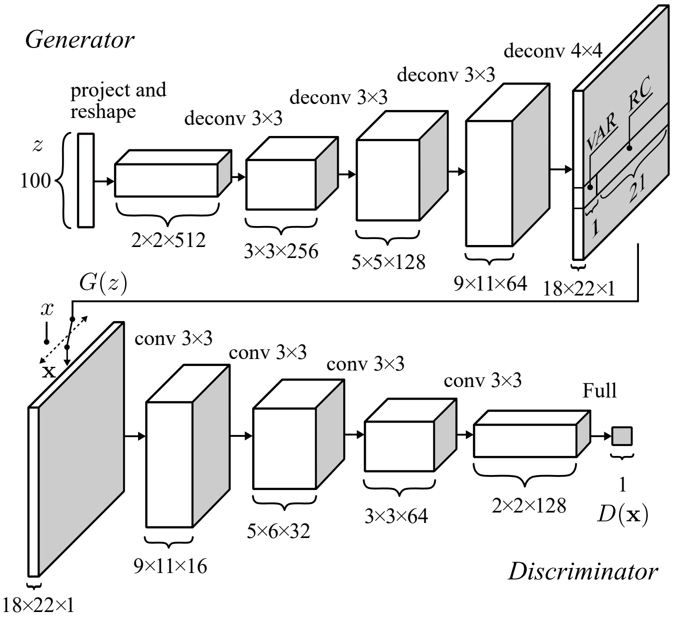

Fig. 5: Network architecture and parameters of DCGAN for generating haptic texture models. The tensor shapes are specified as: *AR feature 1 (model number)* × *AR feature 2 (variance (VAR) + reflection coefficients (RC))* × *input channels*.

C. Network Architecture & Hyper-Parameters

The architecture of our DCGAN model with a four-layer 2D convolutional generator and a four-layer 2D convolutional discriminator is shown in Fig. 5. The generator is deliberately set deeper than the discriminator since we consider the reconstruction of texture models as a harder task than the distinction of models. The hyper-parameters in this work are tuned by random search. We set the latent space as dimension 100. We use spectral normalization [26] for all layers except the last in both generator and discriminator. We also use leaky Rectified Linear Unit with $\alpha = 0.2$ as the activation function for the middle layers of both generator and discriminator to speed up training. Before outputting the texture model by the generator, we apply the softplus on the variance and tanh on the reflection coefficients separately, and then concatenate them to form the final texture model. For the discriminator, we use the sigmoid for the last layer to output a single scalar in $[0,1]$, representing the probability that a texture model is assigned as real.

The model was trained with the Adam optimizer [27] with mean squared error (MSE) serving as the objective function. To balance the convergence speeds, we set the learning rate to be 0.0001 for generator and 0.0004 for discriminator. We trained the models with a minibatch size of 64. The optimal model was chosen at 1600 training epoch for generating haptic texture models. We implemented the model in PyTorch and

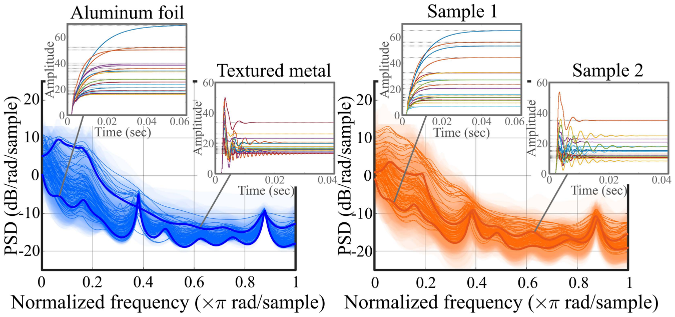

Fig. 6: Comparison of system responses between 100 raw texture models from the HaTT (left) and 100 generated texture models from the generator (right). Each line shows the average PSD of a texture model over force-speed entries and the fading area shows the range of PSD from all force-speed entries. The callout plots show the step responses of two sample texture models of all force-speed entries for better comparison.

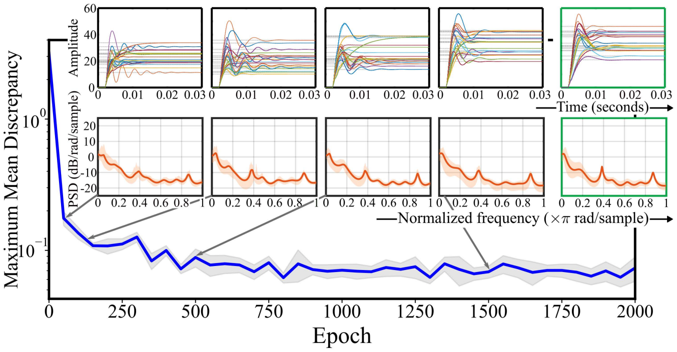

Fig. 7: MMD between ground truth texture models and generated texture models. The top row shows the step response of the texture models of all force-speed entries. The second row illustrates the average and range of the PSD over all force-speed entries. Plots outlined in black show the response of the texture models generated by the GAN generator trained until the pointed epoch. Plots outlined in green show the response of a ground truth texture model from HaTT.

conducted the training on a machine equipped with 64 GB RAM, an Intel Xeon E5-2623v3 CPU, and a GeForce GTX Titan X with 12 GB. The training lasted about 2 hours.

D. Model Evaluation

We conducted qualitative and quantitative analyses on the GAN generator to evaluate the performance of the network.

1) Qualitative Analysis: To qualitatively measure the accuracy and diversity of the generated texture models, we sampled 100 new texture models and visualized the system response at each force-speed entry of all the generated models as Power Spectral Density (PSD). Compared to the PSD of the response from 100 raw texture models in HaTT as shown in Fig. 6, we can qualitatively judge that the trained generator is capable of covering a sufficient variety of textures with high quality. Furthermore, the generated texture models were resampled to the haptic rate of 1 kHz and displayed on a 3D haptic device to evaluate the authenticity of the haptic feedback by the authors.

2) Quantitative Analysis: To evaluate the stability of DCGAN training, we conducted a two-sample test based on kernel Maximum Mean Discrepancy (MMD) [28], a statistical test to decide if two sets of samples are drawn from different distributions [29]. We define the distribution of raw data as p_{data} and the distribution of generated data as p_g, and the corresponding sampling from the distributions as X_{data} and X_g. The smaller the MMD between X_{data} and X_g, the closer p_{data} and p_g are, and the better the performance will be. Fig. 7 is a visualization of MMD between 100 randomly sampled real texture models X_{data} from HaTT (p_{data}) and 100 generated texture models X_g from generator (p_g) along with the training epoch. MMD was calculated and plotted every 50 epochs. The kernel designed for the MMD computation is $\sigma_q = 1, 2, 4, 8, 16$ and $\lambda = 1$. We visualized the examples of step response and PSD of generated texture models evolved along the MMD curve. The plot showed the fast training convergence and the correlation between lower MMD and better texture quality compared to a real texture model from HaTT.

IV. Interactive Texture Search By Human Preference

Using the trained generator in the DCGAN which maps the latent space to haptic texture models, this section introduces our preference-driven texture modeling framework which combines human preferences and an evolutionary texture search.

A. Evolutionary Texture Search

To find a desired virtual texture using the DCGAN generator, we need a method to evolve the latent vector of the generator to the desired region in the latent space. Many optimization methods, either gradient-based or gradient-free algorithms, are able to search a space with the assumptions that the underlying space is a continuous and smooth manifold and the function to optimize is well-behaved. In this case, however, the evaluation of the latent vector in the DCGAN is discontinuous and rugged since it is hard to find an explicit

Algorithm 1 Interactive Texture Search by CMA-ES

```
Input: The mean vector $\boldsymbol{\mu} \in \mathbb{R}^n$ represents the favorite solution, the step-size $\sigma \in \mathbb{R}_+$ controls the step length, and number of candidates $\lambda$.
Initialize: The covariance matrix $\mathbf{C} = \mathbf{I} \in \mathbb{R}^{n \times n}$, which controls the shape of the distribution.
repeat
    $z_i \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{C}),\ i = 1, ..., \lambda$        ▷ sampling
    $z_i^p \leftarrow$ project($z_i$)        ▷ project to surface of hypersphere
    textures $\leftarrow G_\theta(z_i^p)$        ▷ generate texture candidates
    interface.render(textures)
    wait until interface.selectButtonPressed(idx) True
    $\boldsymbol{\mu} \leftarrow$ update(idx)        ▷ idx is index of selected texture
    Adapt $\mathbf{C}$ and $\sigma$ by rank-one update
until user stops
```

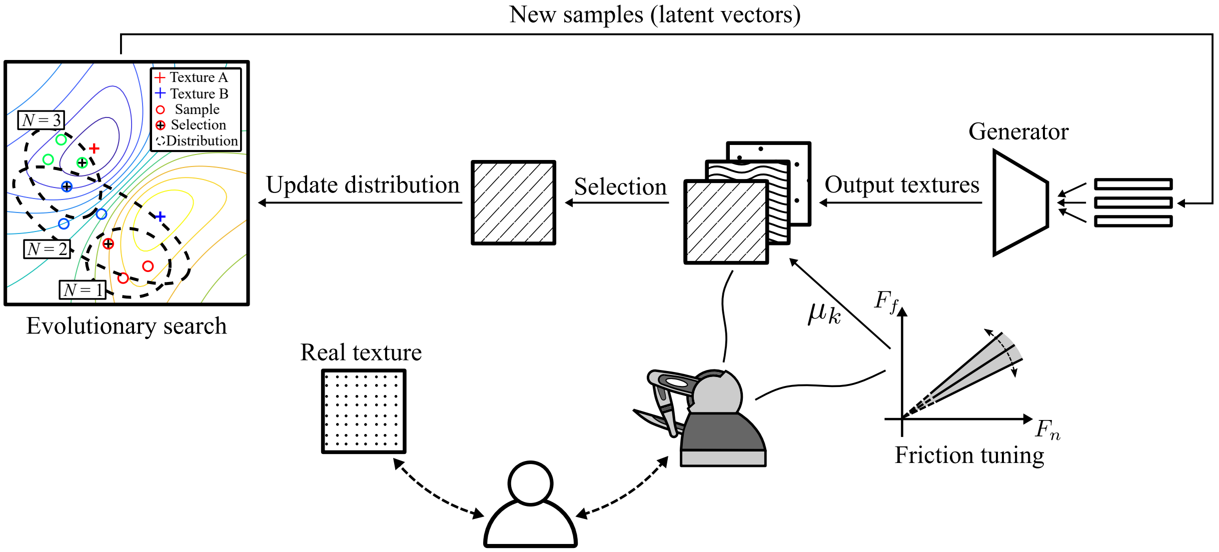

Fig. 8: Preference-driven texture modeling framework. Users interact with a real texture and a set of virtual texture candidates, which are generated from the latent vectors by the GAN generator, using a customized tool and a haptic device, respectively. Based on their preference, users select the virtual texture haptically closest to the real texture. The evolutionary algorithm will update the distribution based on the selection. The new latent vectors sampled from the updated distribution are passed back to the generator to output new texture candidates. Meanwhile, users can adjust the surface friction to match the actual resistance. The process continues until users think the final selected texture feels haptically close enough to the real texture.

fitness function to perceptually evaluate a texture. So, we consider using an evolution strategy to search the latent space.

Standard evolution strategies still require an objective fitness function to optimize for. However, our goal is to generate virtual textures that haptically match the real texture without collecting data from the real texture, which is not quantifiable. Our key insight is to use Covariance Matrix Adaptation Evolution Strategy (CMA-ES) [30], an efficient *comparison-based* algorithm. We can use the ordering specified by the user within a set of generated virtual textures, rather than the absolute evaluation values, to evolve the latent vector in the DCGAN. It allows users to interactively manipulate the search. Moreover, CMA-ES has been successfully used in latent vector evolution in the image domain [31], [32], [33] and performs well in difficult (ill-conditioned, noisy, rugged) manifolds.

To find the target in a latent space, CMA-ES generates a set of latent vectors by sampling a normal distribution and orders them by the user preferences. Based on the order, CMA-ES moves the mean and updates the covariance matrix of the distribution from which we sample new latent vectors. This process repeats until the generated sample reaches the goal.

Algorithm 1 shows the details of the interactive texture search by CMA-ES. The initial step size σ is set as 0.15 and the number of candidates λ at each iteration is 3. We set a small λ to prevent user fatigue and we use mirrored sampling [34] to account for the small λ. From a normal distribution with the same dimension as the latent space, we sample λ latent vectors (line 2). To retain realism in the GAN-generated outputs, we project the latent vectors to the surface of a hypersphere with radius $\sqrt{n}$ [35]. We pass the projected latent vectors through the generator to produce texture models (line 4) and render the textures via a haptic interface (line 5). After touching all textures with the haptic interface, users select the texture perceptually closest to the given real texture (line 6). We convert the user selection to an ordering of the latent vectors, so that the selected vector is ranked highest and the remaining two vectors are randomly ordered. Following the CMA-ES update rules [36], we use the ordering to update the mean of the distribution so it maximizes the probability of the selected texture (line 7), and adapt the covariance matrix of the distribution and the step size to increase the probability of previous search steps (line 8). We then sample a new set of latent vectors based on the updated distribution at the next iteration. This procedure continues until users feel that the selected texture is close enough to the real texture.

B. Preference-Driven Texture Modeling Framework

Our preference-driven texture modeling framework combines the generator in the DCGAN trained on the HaTT dataset and the evolutionary algorithm CMA-ES for evolving the latent vector of the generator (Fig. 8). The major difference between preference-driven and data-driven methods is the basis for modeling a target texture. In data-driven methods, the basis for creating the texture model is the recorded signals from the interaction with the real surface. In preference-driven methods there is no recording step, and the model is created on the basis of a user's subjective preference.

The integration of generative adversarial networks and interactive evolution strategy allows the generator to bound the search space and the evolution strategy to guide the search by a user's preference. The GAN-based generator is trained over a group of real texture models, ensuring realism and diversity of generated texture models and providing a constrained search space for the evolutionary algorithm. The generator starts by transforming a set of latent vectors sampled from a normal distribution into the texture models that have a

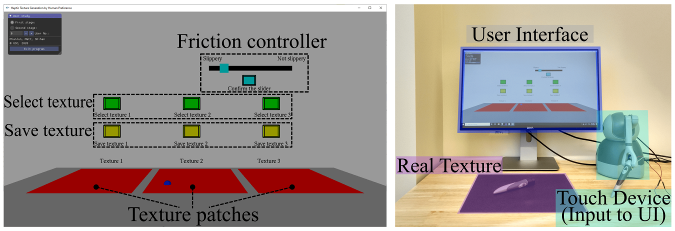

Fig. 9: GUI (left) and experiment setup (right) in Phase 1

decent interactivity of tool states. This interactivity allows the user to freely interact with the generated virtual textures and select the texture closest to the unmodeled real texture. The selection only relies on the user's preference from interactions with a set of textures. Based on the selection, the evolutionary algorithm updates the distribution where we resample the latent vectors, to increase the likelihood of the selected texture.

During the interaction with the generated textures, considering that the Coulomb friction model is controlled by a single scalar of the friction coefficient μ_k, we allow the user to tune the coefficient by a slider based on their perceived resistance from the real texture (Fig. 8). By user's preference during the interaction, the texture search and friction tuning jointly lead to the virtual texture that feels close to the real texture.

V. System Evaluation

We conducted a human-subject study to determine the effectiveness and efficiency of our preference-driven texture modeling system. We wished to further evaluate the quality of the generated textures compared to real textures and data-driven virtual textures using both objective and subjective metrics. We divided the study into two phases – texture generation and search, and texture evaluation – with two separate groups of participants. The study was approved by the USC IRB under protocol UP-20-00806, and participants were compensated with a $15 gift card for their participation.

A. *User Interface and Set-up*

We created a simulation platform in C++ using OpenHaptics and OpenGL for haptic rendering, binding with Python blocks for texture generation and search. In Phase 1, the GUI consists of three "select" buttons and "save" buttons that correspond to three texture patches, and a slider for controlling the friction of all textures (Fig. 9 (left)). Users interacted with the GUI by moving the stylus of a 3D Systems Touch device. In real-time, the user's force and speed were input to texture models to generate virtual texture vibrations that were displayed to the user through an MM3C Haptuator (TactileLabs) rigidly attached to the tip of the stylus. The surface friction was displayed as a force using the Touch device's motors. In Phase 2, the GUI contains only one texture patch at a time, and users touched the texture patch using the Touch device. In both phases, the real texture was placed next to the Touch device. Users touched the real texture using a tool with the same grip as the stylus and a custom 3D-printed tip with a diameter of 3.2 mm; they were instructed to use motions similar to how they interacted with the virtual textures (Fig. 9 (right)).

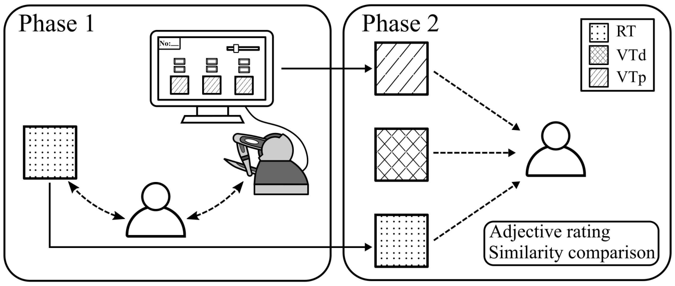

Fig. 10: Experiment procedures of Phase 1 and Phase 2. RT is real texture, VTd is virtual texture by data-driven method, and VTp is virtual texture by preference-driven method.

B. *Experiment Procedure*

The participant was seated at a table and interacted with the simulated texture patches and other GUI components using the Touch device. They were allowed to touch and move along the texture patches in any way they wanted, but tapping and pressing were not allowed. The participant interacted with the real texture using the tool described in Section V-A given the same motion constraints as for virtual textures. They wore headphones playing white noise to block auditory cues. We separated the experiment into two phases and recruited two separate groups of participants for each phase. Five participants (Age: M = 23.8, SD = 1.48; 4 right-handed, 1 ambidextrous) participated in Phase 1, and 14 participants (Age: M = 24.8, SD = 3.56; all right-handed) participated in Phase 2. Fig. 10 shows the two-phase experiment procedure.

1) Phase 1 - Texture Generation and Search: Participants selected the virtual texture in a set of three that felt closest to a given real texture. During the comparison, participants used a slider to freely adjust the friction of the virtual textures. After participants selected their preferred texture, all three texture candidates were updated. Participants could save the texture felt close to the real texture and then keep selecting. This process continued until participants thought the final saved texture felt similar enough to the real texture. The participant completed this modeling process for the five real textures shown in Fig. 11 in random order. For each real texture, the final saved virtual textures were used in the next evaluation phase, creating a total of 5 (number of textures) × 5 (number of participants) = 25 preference-driven virtual textures.

2) Phase 2 - Texture Evaluation: In Phase 2, we provided another set of participants with the following textures:

1) 25 preference-driven virtual textures created by the five participants in Phase 1 (VTp$_i$, $i = 1, \ldots, 5$ where i means ith participant)
2) 5 real textures (RT)
3) 5 data-driven virtual textures (VTd) created by the data recorded by the first author using the method in Section III-A

Participants first interacted with all 35 texture samples in random order and rated the perceptual dimensions of: roughness (smooth–rough), hardness (soft–hard), and slipperiness (not-slippery–slippery). These perceptual dimensions are

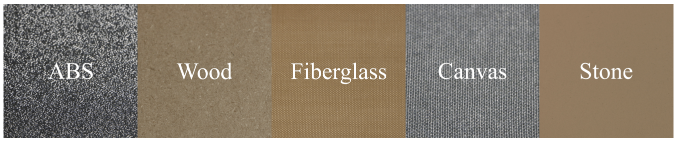

Fig. 11: Five textured surfaces, which were not in HaTT, were used as targets for preference-driven texture modeling.

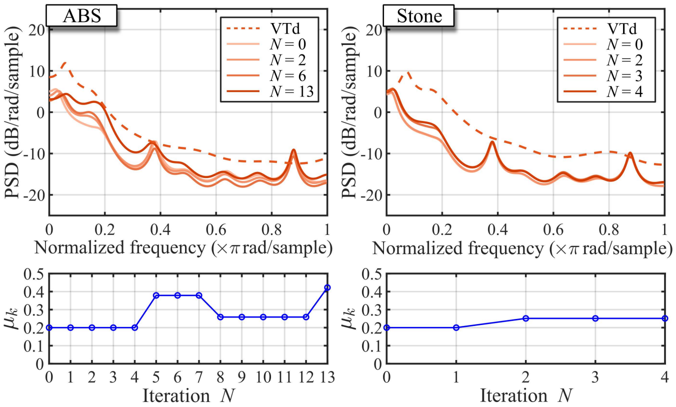

Fig. 12: Examples of texture PSD and friction coefficient μ_k progression based on the participant's selection. The solid lines in the PSD plots represent VTp at Nth iteration and the dash lines represent VTd. For the clarity, we only show the average PSD over all force-speed entries.

widely used to describe a texture felt through a tool [20]. Participants were then given pairs of a real texture and a corresponding virtual textures (either VTd or VTp$_i$) in random order and were asked to rate their similarity on the scale "completely different"–"completely the same". We conducted this similarity comparison with 30 pairs of conditions (VTd-RT (5 pairs) and VTp$_i$-RT (25 pairs)) to test the hypotheses:

1) the preference-driven virtual texture VTp$_i$ feels close to the corresponding real texture;
2) the preference driven virtual texture VTp$_i$ represents the corresponding real texture equivalently or better than the data-driven virtual texture.

We randomize the order of the adjective ratings and similarity comparison steps to eliminate order effects.

VI. Results

A. Phase 1

Our analysis of Phase 1 focuses on the robustness and efficiency of our preference-driven texture modeling framework. To show how the selected textures change toward the user's preference, we visualized the PSD of the texture models and the friction coefficients for a single preference-driven modeling cycle for ABS and stone in Fig. 12. Due to the distinct accelerometers used for the HaTT dataset (two dual-axis ADXL321, up to 2.5 kHz for X and Y axes) and our recording for VTd (one 3-axis ADXL326, up to 1.6 kHz for X and Y axes and up to 550 Hz for Z axis), it is difficult to directly compare the PSD between VTp and VTd at certain frequencies in Fig. 12. However, we can clearly see the change of the PSD of VTp along the iteration N. Moreover, the different PSD shape between the final saved VTp and VTd indicates that there may exist multiple or even better ways to model a texture besides using the recorded data directly.

TABLE I: Evaluation Metrics of Phase 1: Texture Search

Texture	# of iterations	# of friction tuning	Time (mins)
ABS	5.2 ± 2.77	2.2 ± 0.84	4.44 ± 1.28
Wood	5 ± 4.53	3.2 ± 1.64	4.35 ± 2.25
Fiberglass	5.4 ± 2.70	2.4 ± 1.14	4.38 ± 0.79
Canvas	4.4 ± 1.52	2.4 ± 1.34	5.01 ± 2.2
Stone	3.8 ± 1.10	1.8 ± 1.64	3.56 ± 0.79

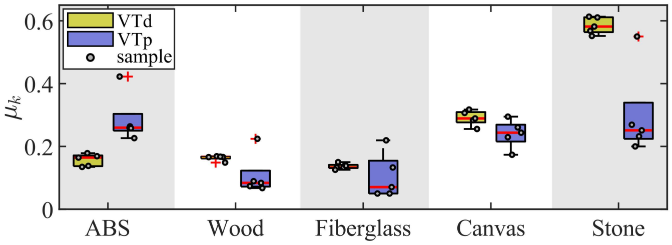

Fig. 13: Comparison of Coulomb friction coefficients μ_k between VTd and VTp. Five samples for VTd were computed from five 10-second data recorded by the first author. Five samples for VTp were tuned by five participants in Phase 1.

To evaluate the efficiency of the texture search, we summarized some key metrics in Table I. The number of selection iterations lies at around five for most textures, and the number of friction tuning iterations is about two. The average time for searching a texture is steady at about 5 minutes across all textures and users. All the metrics have reasonable variances across the textures, which indicates that users are able to find textures with different characters using this system.

B. Phase 2

Our analysis of Phase 2 emphasizes the texture evaluation and comparison of preference-driven virtual textures (VTp), data-driven virtual textures (VTd) and real textures (RT), both objectively and subjectively. We introduce the spectral difference analysis [2] as a metric for objective vibration comparison, and use adjective ratings and similarity comparison for subjective evaluation.

1) Friction Comparison: The friction coefficient μ_k in the Coulomb friction model is tuned by users in the preference-driven method, while it is directly computed by recorded data in the data-driven method. Fig. 13 shows the comparison of friction coefficients μ_k between these two methods. The variance of the coefficients from the preference-driven method is larger than that from the data-driven method (average of std dev of μ_k across textures: 0.017 for VTd and 0.081 for VTp). For the rough textures such as ABS and stone, there is a median offset of the coefficients between these two methods (-0.10 for ABS and 0.33 for stone). It shows the subjectivity of capturing the surface friction property by pure perceived resistance, and the discrepancy between real friction measurements and virtual friction perception.

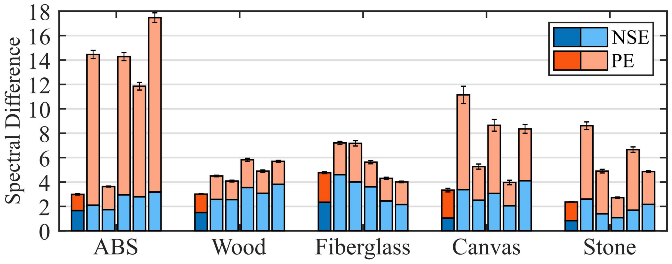

Fig. 14: Spectral difference of VTd-RT and VTp$_i$-RT across textures. For each texture, dark color bars show VTd-RT and other five bars show VTp$_i$-RT. Error bars show standard error.

The surface friction has been shown to affect perceived roughness and slipperiness [37], [38], and it is a key variable for judging the realism of a virtual texture [21]. Therefore, the friction discrepancy between the two methods leads us to further investigate the subjective ratings of roughness, slipperiness, and similarity later in this section.

2) Spectral Difference Analysis: The goal of our spectral difference analysis is to quantitatively compare vibration signals from VTd, VTp$_i$ with signals from RT using a frequency-spectra metric as in [2]. Since the vibration signals depend on the actual force and speed of the motion, we used the same pre-defined set of ten force-speed trajectories with a duration of ten seconds each for all three types of textures.

The vibration data was recorded from real textures using a 3-axis ADXL326 accelerometer and simulated by texture models for the two types of virtual textures. We computed the spectra using a window of 2.56 ms with 50% overlap. The spectral difference (SD) evenly allocates weights on the Normalized Spectral Error (NSE) and the Power Error (PE):

$$\mathrm{SD} = \frac{1}{N_w}\sum_j \left(\frac{\mathrm{NSE}_j}{\mathrm{std(NSE)}} + \frac{\mathrm{PE}_j}{\mathrm{std(PE)}}\right) \tag{3}$$

where N_w is the number of the windows, and NSE_j and PE_j are corresponding errors at jth window, which describe the errors of spectral shape and spectral power, respectively (see [2]).

The bar plot of the spectral difference (SD) between data-driven virtual texture and real texture (VTd-RT) and between preference-driven virtual texture and real texture (VTp$_i$-RT) is shown in Fig. 14. Not surprisingly, the SD of VTp$_i$-RT is generally larger than that of VTd-RT, because VTd is modelled directly using the data recorded from the real texture while VTp$_i$ does not rely on this data. For rough textures such as ABS, canvas, and stone modelled by VTp$_i$, PE dominates the SD. PE also mainly contributes to the difference of SD between VTp$_i$-RT and VTd-RT for those rough textures. For relatively smooth textures such as wood and fiberglass, there is an even distribution between NSE and PE, in both VTd-RT and VTp$_i$-RT conditions. Their PE is smaller than most rough textures while their NSE remains on the same level. In addition, their distinction of SD between VTd-RT and VTp$_i$-RT is much smaller than that of the rough textures.

3) Subjective Analysis: To further evaluate the textures modeled by our method, we analyzed participants' responses from adjective ratings and similarity comparison.

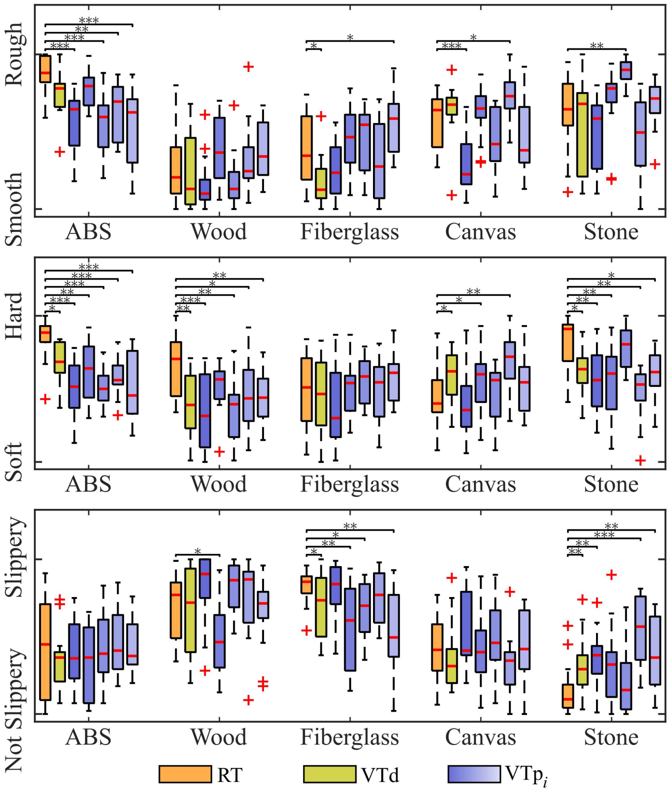

Fig. 15: Adjective ratings for real texture (RT), data-driven virtual texture (VTd) and preference-driven virtual texture (VTp$_i$). Statistical significance in the adjective ratings between RT and virtual textures (VTd and VTp$_i$) is marked ($*** \equiv p < 0.001$, $** \equiv p < 0.01$, $* \equiv p < 0.05$).

a) Adjective Ratings: We first conducted a two-way ANOVA separately for each texture and each adjective scale with type (RT, VTd, and VTp$_i$, where $i = 1,2,..,5$) and participant as factors. Significance of type was found in all textures for all adjective scales except fiberglass's hardness ($p = 0.23$), and ABS's and canvas's slipperiness ($p = 0.56$; $p = 0.15$). Participant was a significant factor in all textures for all scales except fiberglass's roughness ($p = 0.29$) and stone's hardness and slipperiness ($p = 0.07$; $p = 0.16$). By the post-hoc multiple comparison test, the virtual textures with ratings significantly different from the real textures are labeled with significance bars in Fig. 15. For the roughness scale, most significant differences occurred in the ABS. The virtual textures, especially the preference-driven textures, could not capture the roughness of the real ABS. Both data-driven and preference-driven virtual textures failed to capture the hardness of the real texture for all but fiberglass because the hardness was not directly rendered and participants were only allowed to drag the stylus horizontally across the texture. For slipperiness, significant variation from the real texture occurred mainly in the two extreme textures – most slippery (fiberglass) and least slippery (stone). For these two textures, there was no obvious difference between VTd and VTp$_i$ in the rated slipperiness.

As discussed above, there were huge gaps in the friction coefficients between VTd and VTp for ABS and stone. However, those gaps were not proportionally reflected in the ratings

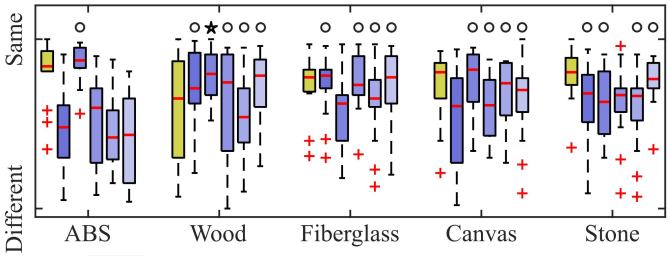

Fig. 16: Similarity ratings of VTd-RT and VTp$_i$-RT conditions. "○" means that the similarity rating of VTp$_i$-RT for a texture is not significantly different from that of VTd-RT ($p > 0.05$), and "⋆" means that the similarity rating of VTp$_i$-RT is significantly higher than that of VTd-RT.

of roughness and slipperiness. The ANOVA and post-hoc test within texture indicated that for ABS, VTd had significantly higher roughness than VTp$_1$ ($p = 0.018$), VTp$_3$ ($p = 0.010$), and VTp$_5$ ($p = 0.034$); for stone, VTd had significantly lower roughness than VTp$_3$ ($p < 0.001$). Moreover, there was no significance found between VTd and VTp$_i$ in the slipperiness ratings for both ABS and stone. These ratings contradicted our observed differences in the friction coefficients.

b) Similarity Comparison: Fig. 16 shows the similarity ratings for different comparison conditions (VTd-RT and VTp$_i$-RT). We first conducted two-way ANOVA within texture using participant and comparison condition as factors and then conducted a post-hoc test for the textures that show significance in terms of comparison condition. For wood, fiberglass, canvas, and stone, at least four out of five preference-driven virtual textures, which are marked with ○ in Fig. 16, achieved a similarity rating that was not significantly different from the corresponding data-driven virtual texture. The rating of VTp$_2$ of wood was found to be significantly higher than that of wood's VTd ($p = 0.017$) and was marked with a ⋆. Yet, for ABS the VTd outperformed four VTp$_i$ based on the comparison with the real ABS. Only VTp$_2$ of ABS reached a similarity rating not significantly different from the VTd.

We take a more detailed look at ABS, the texture with the largest difference in the similarity rating between VTd-RT and VTp$_i$-RT. The friction coefficient comparison for ABS does show an offset between VTd and VTp, but this offset does not affect participants' slipperiness rating. Instead, the spectral difference analysis reveals the large difference in the power error of the vibration between VTd and four VTp$_i$ for ABS (all except VTp$_2$), which coincides with the difference in roughness ratings. These four VTp$_{i=1,3,4,5}$ have significantly lower rated roughness than that of RT, and their similarity ratings are significantly lower than that of VTd. These differences mean that for some rough textures such as ABS, users' perception of the vibration power may be biased by other factors such as interactive motion, individual difference, and lack of other feedback (e.g., audio), which consequently causes a bias in the perceived roughness. This perception bias during the texture search can negatively affect a preference-driven texture's realism during the texture evaluation.

VII. Discussion

A. Secondary Result Analysis

1) Preference-Driven Friction Tuning: The results showed that the variance of friction coefficients from the preference-driven method was larger than that from the data-driven method. Furthermore, there existed an offset of the coefficients between these two methods, especially for rough textures (ABS and stone). However, there was no significant difference in the slipperiness ratings between VTd and VTp in Phase 2.

Previous work in haptic perception has shown that the slipperiness/stickiness perception is moderated by deep cutaneous receptors, the Pacinian Corpuscles, and consequently depends on multiple factors including age, contact force, and skin properties [38]. This variation in perception partially explains the relatively large variance of friction tuning across users and was also mentioned by one user in the post-experiment survey. Bensmaïa and Hollins [37] further showed that the perception of slipperiness was mediated by the subjective intensity of vibrations. Since the rough textures delivered stronger vibration through the tool, it was more likely that user's perception of slipperiness was affected when interacting with those textures, resulting in an offset between the actual and tuned friction coefficients. Similarly, the above factors can be applied to the texture evaluation phase, in which the slipperiness of rough textures was rated similarly for VTd and VTp although their friction coefficients were very different.

From the post-experiment survey in Phase 1, one participant indicated that the perceived friction from real textures was related to the contact angle. The participant tried to keep this angle constant but found it difficult to confidently judge the friction of real textures. We also observed noticeable variances on contact angle and normal force across participants during the exploration. Two participants reported that the layout of the three texture candidates in the simulation caused different haptic perception during the search, especially the perceived friction, due to the mechanical limit of the haptic device. They also mentioned that in the simulation, the perceived friction from the stylus was perceptibly different in the different movement directions, which was another issue caused by the device's mechanical limit. Prior knowledge of a texture was also specified to affect user's perception. For instance, one participant assumed that bumpy textures (e.g. ABS) usually had a high friction, although this may not match their perceived friction from the real texture. All points above can explain the relatively large variance and evident offset of the preference-driven friction tuning compared to the data-driven method.

2) Preference-Driven Haptic Texture Modeling: The power error (PE) was shown to be the main contributor to the spectral difference (SD) between the real textures and VTp for rough surfaces (ABS, canvas, stone). This PE was also shown to be the main source of difference between VTd and VTp for these surfaces. In contrast, the normalized spectral error (NSE) was fairly constant across all surfaces and model types.

Differences in roughness rating between real and virtual textures were seen mainly in ABS. The significance in rated roughness between VTd and VTp for ABS aligned with our findings from the spectral difference analysis, which showed a

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TOH.2022.3173935, IEEE Transactions on Haptics

correlation between PE and perceived roughness. The hardness ratings for both VTd and VTp varied from the real texture due to a lack of separate hardness rendering, such as the display of tapping transients [21]. Compared to real textures, variation in the slipperiness ratings for virtual textures was observed mostly in the two extreme textures (fiberglass and stone). There was no evident difference between the slipperiness ratings of VTd and VTp for any texture even though their friction coefficients were distinct. This lack of difference in the perception of slipperiness shows its complexity and the interplay of friction and vibration in its perception.

The similarity comparison showed that 72% preference-driven textures performed comparably to the data-driven textures. However, for certain rough textures such as ABS, the perceptual bias of vibration power during the texture search may degrade the texture evaluation by other users. Besides the perceptual bias on vibration power, another possibility for the poor performance of the texture search on ABS is that most users in Phase 1 have no previous experience operating the haptic device and no experience with texture simulation. So, it is difficult for them to determine how a real texture should feel in the virtual environment since we do not offer any virtual texture samples before the study, and such uncertainty is amplified for rough textures. This issue may be alleviated if the operators are experienced in haptics or if they have a high haptic sensitivity due to their occupation, such as a surgeon.

From the post-experiment survey in Phase 1, one participant mentioned that they felt distracted by visual feedback from the real texture during the virtual texture search and that this visual feedback may unintentionally affect their haptic perception. Another participant mentioned that the visual misalignment between real and virtual textures created difficulty in the texture judgement since we did not map the image of the real texture to the virtual texture. When the simulation presented haptic feedback on a visually non-textured surface, it confused the user and negatively affected their perception. The conflict between the vision and haptics in the texture rendering can be eliminated by adding the texture image to the virtual texture or by blocking the visual feedback from the real texture. These comments also emphasized the importance of multimodal congruence in the virtual texture rendering.

Several participants reported difficulty in rating the hardness and mentioned the virtual texture felt softer than the real texture in Phase 2, which was in line with our expectation due to our imposed limitations on their motions and the absence of direct hardness rendering. One participant also noticed that the white noise played through the headphones for blocking the sounds from the interaction had an impact on their haptic perception. This indicated the importance of sound modeling and rendering in the virtual tool-surface interactions [39].

B. Method Comparison and Limitations

Previous work on texture generative models focuses on the interactivity of a texture's attributes, such as generating the frequency spectrum of the acceleration, so that it is closer to the generation of a "static texture" equivalent to an image in vision [14]. Conversely, this work focuses on the interactivity of a tool's state and creates a generator of "*texture model*" that can generate acceleration data given a user's motions in real time. The texture model in this work is similar to a "dynamic texture" equivalent to a video. This shift in focus gives users freedom to interact with the texture as they desire, rather than constraining them to follow a certain path, speed, and force. This freedom is significant in our system since our interactive search method for a target texture fully relies on a user's preference. Giving users freedom to interact with the texture can ensure the accuracy of the real-time haptic perception and maximize the effectiveness of the texture search.

Since HaTT provides the raw acceleration data and a set of Auto-Regressive (AR) texture models for each surface, it gives us a concrete base for constructing generators directly on texture models rather than on raw data, and allows us the flexibility to apply modifications. Learning with structures from AR models can also greatly reduce the dependence on a large database, compared with end-to-end learning on raw acceleration data [14]. Although we may not be able to surpass the capacity of the AR model itself in this approach, many previous works have proven the effectiveness of modeling a variety of textures using AR processes [40], [5]. How to use a model-free method to efficiently represent texture models would also be an interesting future direction. For the method's robustness to the hardware variance such as tooltip change, it does not affect the modeling procedure although the final preference-driven texture will be varied accordingly since we model how the texture feels when touched with a specific tool.

VIII. Conclusion

This paper created an interactive texture generation and search system driven by human preferences, called "*preference-driven*" texture modeling. Our system can quickly model and render new textures using a GAN-based texture model generator based on the existing haptic texture database and an interactive evolutionary algorithm combined with human selections for texture search. In contrast to data-driven methods, our preference-driven method does not require additional data collection for unmodeled textures, which is a process that is not accessible for most consumers.

We implemented this preference-driven texture modeling method on a haptic device with an integrated GUI. We conducted a two-phase human-subject study to evaluate the efficiency and accuracy of the texture generation and search for unmodeled textures. The results indicated that the texture search can be completed within a reasonable and consistent time range across textures and users. Most preference-driven textures created by separate users were rated comparably to the data-driven textures and achieved high similarity ratings compared to the real textures. However, subjective vibration power error in the search may reduce the realism of preference-driven textures for certain rough texture such as ABS.

To simplify the interactive texture generation and search framework, we use the self-adapted search step in the evolutionary algorithm for texture search. For the more precise control in fine-tuning of the texture, a hierarchical search with user-defined search step can be applied, which can be of interest to professional designers. Additionally, with the growth of haptic texture data created under this framework, a collaborative interactive evolution for texture design can be made based on textures previously designed by others. Users can resume the texture search of others from intermediate or final results to not only accelerate the process, but also to satisfy their own design requirements.

This framework for texture modeling can be generalized to other surface haptic texture rendering methods (i.e., ultrasonic and electrovibration) and other interaction types (i.e., bare finger interaction). To use the framework on other texture display systems, the architecture of the generator must be adapted, but the main algorithm remains consistent. Furthermore, using the interplay between vision and touch, it is possible for users to infer the texture properties of an object under the guidance of vision alone, rather than through touch. Inferring and searching texture properties of an object without haptic data recording can greatly enrich existing haptic texture databases and help us remove the constraints of hardware, time, and expertise in the data recording and modeling processes.

ACKNOWLEDGEMENT

The authors thank Giovanni Sutanto and Chaoyang He for useful conversations about texture generative models. The first author thanks Alex Atcheson for suggestions on study design.

REFERENCES

[1] A. M. Okamura, J. T. Dennerlein, and R. D. Howe, “Vibration feedback models for virtual environments,” in *Proc. IEEE Int. Conf. on Robotics and Automation*, vol. 1, 1998, pp. 674–679.

[2] H. Culbertson, J. Unwin, B. E. Goodman, and K. J. Kuchenbecker, “Generating haptic texture models from unconstrained tool-surface interactions,” in *Proc. IEEE World Haptics Conf.*, 2013, pp. 295–300.

[3] S. Shin, R. H. Osgouei, K.-D. Kim, and S. Choi, “Data-driven modeling of isotropic haptic textures using frequency-decomposed neural networks,” in *Proc. IEEE World Haptics Conf.*, 2015, pp. 131–138.

[4] A. Abdulali and S. Jeon, “Data-driven modeling of anisotropic haptic textures: Data segmentation and interpolation,” in *Int. Conf. Human Haptic Sens. Touch Enabled Comput. Appl.*, 2016, pp. 228–239.

[5] H. Culbertson, J. Unwin, and K. J. Kuchenbecker, “Modeling and rendering realistic textures from unconstrained tool-surface interactions,” *IEEE Transactions on Haptics*, vol. 7, no. 3, pp. 381–393, 2014.

[6] H. Culbertson, J. J. L. Delgado, and K. J. Kuchenbecker, “One hundred data-driven haptic texture models and open-source methods for rendering on 3d objects,” in *Proc. IEEE Haptics Symposium*, 2014, pp. 319–325.

[7] Y. Ujitoko, S. Sakurai, and K. Hirota, “Vibrator transparency: Re-using vibrotactile signal assets for different black box vibrators without re-designing,” in *Proc. IEEE Haptics Symp.*, 2020, pp. 882–889.

[8] N. Heravi, W. Yuan, A. M. Okamura, and J. Bohg, “Learning an action-conditional model for haptic texture generation,” in *Proc. IEEE Int. Conf. on Robotics and Automation*, 2020, pp. 11 088–11 095.

[9] M. Strese *et al.*, “A haptic texture database for tool-mediated texture recognition and classification,” in *Proc. IEEE Int. Symp. on Haptic, Audio and Visual Environments and Games*, 2014, pp. 118–123.

[10] M. Strese, C. Schuwerk, A. Iepure, and E. Steinbach, “Multimodal feature-based surface material classification,” *IEEE Transactions on Haptics*, vol. 10, no. 2, pp. 226–239, 2016.

[11] H. Seifi, K. Zhang, and K. E. MacLean, “Vibviz: Organizing, visualizing and navigating vibration libraries,” in *Proc. IEEE World Haptics Conf.*, 2015, pp. 254–259.

[12] W. Xian *et al.*, “Texturegan: Controlling deep image synthesis with texture patches,” in *Proc. IEEE Conf. Comput. Vis. Pat. Recog.*, 2018, pp. 8456–8465.

[13] Y. Gan *et al.*, “Perception driven texture generation,” in *Proc. IEEE Int. Conf. on Multimedia and Expo*, 2017, pp. 889–894.

[14] Y. Ujitoko and Y. Ban, “Vibrotactile signal generation from texture images or attributes using generative adversarial network,” in *Int. Conf. Human Haptic Sensing Touch Enabled Comput. App.*, 2018, pp. 25–36.

[15] H. Culbertson and K. J. Kuchenbecker, “Should haptic texture vibrations respond to user force and speed?” in *Proc. IEEE World Haptics Conf.*, 2015, pp. 106–112.

[16] Y. Ujitoko, Y. Ban, and K. Hirota, “Gan-based fine-tuning of vibrotactile signals to render material surfaces,” *IEEE Access*, vol. 8, pp. 16 656–16 661, 2020.

[17] W. Hassan, A. Abdulali, and S. Jeon, “Authoring new haptic textures based on interpolation of real textures in affective space,” *IEEE Trans. Ind. Electron.*, vol. 67, no. 1, pp. 667–676, 2019.

[18] E. Pescara, M. Hefenbrock, T. Rzepka, and M. Beigl, “Vibration personalization with evolutionary algorithms,” in *Proc. ACM Int. Symp. Pers. Ubiquitous Comput. and Wearable Comput.*, 2018, pp. 436–439.

[19] E. Pescara, F. Dreschner, K. Marky, K. Kunze, and M. Beigl, “Genvibe: Exploration of interactive generation of personal vibrotactile patterns,” in *Proc. Int. Conf. Augmented Humans*, 2020, pp. 1–9.

[20] S. Okamoto, H. Nagano, and Y. Yamada, “Psychophysical dimensions of tactile perception of textures,” *IEEE Transactions on Haptics*, vol. 6, no. 1, pp. 81–93, 2012.

[21] H. Culbertson and K. J. Kuchenbecker, “Importance of matching physical friction, hardness, and texture in creating realistic haptic virtual surfaces,” *IEEE Trans. on Haptics*, vol. 10, no. 1, pp. 63–74, 2016.

[22] R. L. Klatzky and S. J. Lederman, “Tactile roughness perception with a rigid link interposed between skin and surface,” *Perception & Psychophysics*, vol. 61, no. 4, pp. 591–607, 1999.

[23] J. S. Erkelens, *Autoregressive modelling for speech coding: estimation, interpolation and quantisation*. Citeseer, 1996.

[24] R. Viswanathan and J. Makhoul, “Quantization properties of transmission parameters in linear predictive systems,” *IEEE Trans. on Acoustics, Speech, and Signal Processing*, vol. 23, no. 3, pp. 309–321, 1975.

[25] A. Radford, L. Metz, and S. Chintala, “Unsupervised representation learning with deep convolutional generative adversarial networks,” in *Proc. Int. Conf. for Learning Representations*, 2016.

[26] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, “Spectral normalization for generative adversarial networks,” *Proc. Int. Conf. for Learning Representations*, 2018.

[27] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” in *Proc. Int. Conf. for Learning Representations*, 2015.

[28] C.-L. Li, W.-C. Chang, Y. Cheng, Y. Yang, and B. Póczos, “Mmd gan: Towards deeper understanding of moment matching network,” in *Adv Neural Inf Process Syst*, 2017, pp. 2203–2213.

[29] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. Smola, “A kernel two-sample test,” *The Journal of Machine Learning Research*, vol. 13, no. 1, pp. 723–773, 2012.

[30] N. Hansen and A. Ostermeier, “Completely derandomized self-adaptation in evolution strategies,” *Evolutionary Computation*, vol. 9, no. 2, pp. 159–195, 2001.

[31] P. Bontrager, A. Roy, J. Togelius, N. Memon, and A. Ross, “Deepmasterprints: Generating masterprints for dictionary attacks via latent variable evolution,” in *Proc. IEEE Int. Conf. on Biometrics Theory, Applications and Systems*, 2018, pp. 1–9.

[32] P. Bontrager, W. Lin, J. Togelius, and S. Risi, “Deep interactive evolution,” in *Int. Conf. Comput. Intel. in Music, Sound, Art and Design*, 2018, pp. 267–282.

[33] J. Secretan *et al.*, “Picbreeder: evolving pictures collaboratively online,” in *Proc. CHI Conf. Human Factors Comput. Sys.*, 2008, pp. 1759–1768.

[34] D. Brockhoff, A. Auger, and N. Hansen, “On the effect of mirroring in the ipop active cma-es on the noiseless bbob testbed,” in *Proc. Conf. companion on Genetic and Evol. Comput.*, 2012, pp. 277–284.

[35] S. Menon *et al.*, “Pulse: Self-supervised photo upsampling via latent space exploration of generative models,” in *Proc. IEEE Conf. Comput. Vis. Pat. Recog.*, 2020, pp. 2437–2445.

[36] N. Hansen, “The cma evolution strategy: A tutorial,” *arXiv preprint arXiv:1604.00772*, 2016.

[37] S. Bensmaïa and M. Hollins, “Pacinian representations of fine surface texture,” *Perception & Psychophysics*, vol. 67, no. 5, pp. 842–854, 2005.

[38] R. L. Klatzky, D. Pawluk, and A. Peer, “Haptic perception of material properties and implications for applications,” *Proc. of the IEEE*, vol. 101, no. 9, pp. 2081–2092, 2013.

[39] S. Lu, Y. Chen, and H. Culbertson, “Towards multisensory perception: Modeling and rendering sounds of tool-surface interactions,” *IEEE Transactions on Haptics*, vol. 13, no. 1, pp. 94–101, 2020.

[40] J. M. Romano and K. J. Kuchenbecker, “Creating realistic virtual textures from contact acceleration data,” *IEEE Transactions on Haptics*, vol. 5, no. 2, pp. 109–119, 2011.