Received 1 December 2022, accepted 11 December 2022, date of publication 16 December 2022,
date of current version 21 December 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3230065

RESEARCH ARTICLE

Data-Driven Haptic Modeling and Rendering of Viscoelastic Behavior Using Fractional Derivatives

HOJUN CHA[1], (Associate Member, IEEE), AMIT BHARDWAJ[2], (Member, IEEE), AND SEUNGMOON CHOI[1], (Senior Member, IEEE)
[1]Department of Computer Science and Engineering, Pohang University of Science and Technology (POSTECH), Pohang 37673, South Korea
[2]Department of Electrical Engineering, IIT Jodhpur, Karwar, Rajasthan 342030, India

Corresponding author: Seungmoon Choi (choism@postech.ac.kr)

This work was supported in part by the Morphable Haptic Controller for Manipulating Virtual Reality (VR).Augmented Reality (AR) Contents under Grant 2020-0-00594; in part by the Development of Non-Wearable Visual-Tactile Digital Twin Platform Technology to provide various interpretation of digital objects from Institute of Information and Communications Technology Planning and Evaluation (IITP) under Grant 2022-0-01005; and in part by the Data-Driven Haptic Modeling and Rendering of Normal Interactions on Inhomogeneous Viscoelastic Deformable Objects from the Department of Science and Technology (DST), Science and Engineering Research Board (SERB), India, under Grant SRG/2020/000539.

This work involved human subjects or animals in its research. Approval of all ethical and experimental procedures and protocols was granted by the Institutional Review Board of POSTECH under Application No. PIRB-2021-E054.

ABSTRACT Data-driven modeling and rendering is a general approach in haptics aiming to provide highly accurate haptic perceptual experiences simulating complex real physical dynamics, such as deformable or textured objects. A prevalent problem in the present methods for data-driven haptics is that the computational cost for modeling grows rapidly, even becoming intractable, as the interaction complexity or the number of data increases. This paper proposes one data-driven method featured with greatly improved computational efficiency for modeling viscoelastic deformable objects. This advantage is enabled by the use of fractional derivatives for modeling features and regression forests for data-interpolation models. For the benchmark of normal interaction on deformable objects, we describe a computational framework for data-driven haptic modeling and rendering. Its performance is validated by physical experiments for modeling accuracy and cost and a perceptual experiment for the similarity between real and virtual objects. The experiments demonstrate that our method offers highly realistic haptic perceptual experiences with markedly better modeling cost (at least ten times) than other state-of-the-art methods.

INDEX TERMS Haptics, contact modeling, rendering, data-driven, deformable object, virtual reality, fractional calculus.

I. INTRODUCTION

This paper revisits the fundamental research problem of *data-driven haptic modeling and rendering*, nicknamed "haptic camera" [1] or "haptography" [2], for viscoelastic deformable objects. As illustrated in Figure 1, this approach captures the complex physical dynamics of a real object when a human finger touches and explores it, expresses the real object's dynamics by a black-box model from the input (finger position and its derivatives) to the output (response force), and then simulates the black-box model to present the identical haptic experiences to a user in a virtual environment. The black-box models for input-output interpolation are built empirically by applying appropriate machine learning techniques to the captured real interaction data.

The data-driven approach in haptics has advantages for objects exhibiting complex physical phenomena, e.g., deformable or textured objects. Such physical properties are difficult to represent accurately using the alternative of physics-based modeling and simulation, especially because of the strict real-time computation requirement of haptic rendering. The rule of thumb is that update rates of no less than 1 kHz are necessary to provide stable haptic interaction with a stiff virtual object using a force-feedback interface. In fact, the faster the update rate is, the better the stability of haptic interaction becomes [3]. Besides, the physics-based methods

The associate editor coordinating the review of this manuscript and approving it for publication was Yangmin Li.

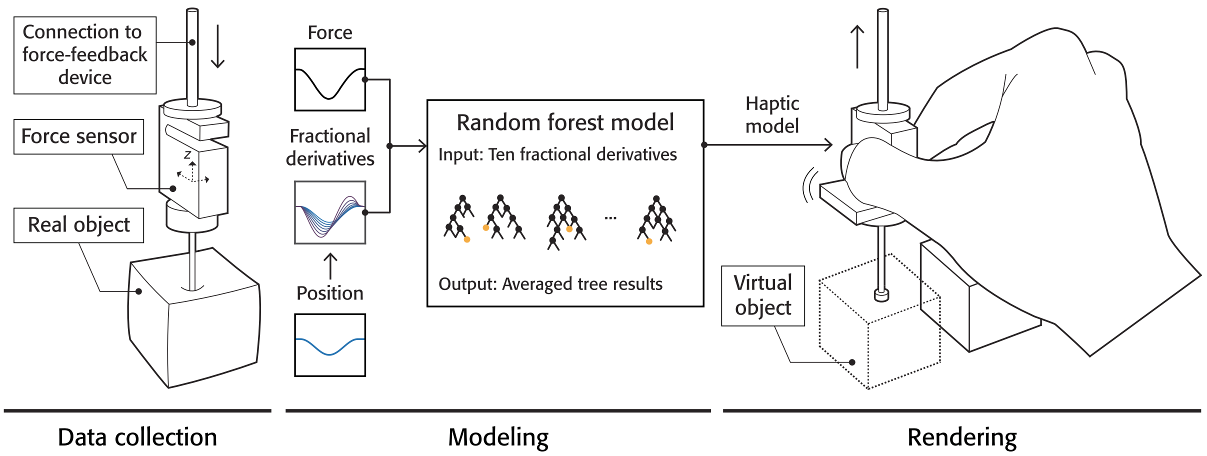

FIGURE 1. While a user haptically interacts with a real deformable object, the user's interaction motion and the object's response force are recorded, as if taking "haptic photographs". The recorded data is processed and used to make a blackbox computational model, e.g., a random forest model, between the motion (input) and the force (output). This data-driven model is used to recreate the interaction dynamics of the real object with a haptic interface in a virtual environment.

often require an exhaustive identification or tuning procedure of their model parameters for the faithful simulation.

A critical issue present in data-driven haptics for deformable objects is that the amount of real interaction data and its processing time, i.e., the modeling cost, easily becomes intractable when the interaction patterns are complex. Examples include an object pressed by multiple fingers and a case exhibiting significant inhomogeneous behaviors depending on the contact position; see Section II. In this paper, we present a new data-driven approach based on *fractional derivatives (FDs)* to improve the modeling efficiency.

Given a mathematical function, a fractional derivative is its derivative of arbitrary order (not necessarily integer); see Appendix A. FDs have been used to describe complex mechanical phenomena, where viscoelasticity is one of the main targets (Section II). In our framework, while pressing and releasing a real viscoelastic object using an instrumented probe, we measure the probe's normal position to the object surface, which represents the extent of deformation, and the response force generated by the object (Section III). Then, we compose an input feature vector for machine learning models using FDs of the normal position (Section IV). Each FD has an order between 0 and 1, corresponding to the position and velocity. This is one of the crucial differences from the previous data-driven methods, which used either the position and velocity pair or the current and several past position samples as input features (Section II). As interpolation models to predict the response force, we test both regression forests (RFs) and radial basis functions (RBFs). The computational performance of our new approach is evaluated experimentally in comparison with other state-of-the-art methods. We employ the trained models for rendering deformable objects in a virtual environment (Section V). Finally, the perceptual accuracy of our data-driven framework is assessed by a human-subject experiment (Section VI. Conclusions are summarized in Section VII with a plan for future work.

Contributions of our work consist of: (1) the first application of fractional derivatives to haptic data-driven modeling and rendering of deformable objects; (2) the proposal of a regression forest-based interpolation model leading to substantially reduced modeling cost; and (3) the physical and perceptual validations of the modeling and rendering performance of the developed methods.

II. RELATED WORK

The recent advances in machine learning has allowed data-driven modeling to be actively used for describing complex phenomena in many fields of science and engineering, such as physical modeling, material modeling, weather forecasting, and financial forecasting [4]. In physics, researchers have developed physics-informed machine learning models trained on the additional information more than the physical laws [5]. For example, Hatfield et al. [6] has highlighted the importance of data-driven methods for high-energy density physics. In [7], the authors have used deep learning frameworks to solve nonlinear partial differential equations. Cenedese et al. [8] has developed a data-driven modeling and prediction method for non-linear dynamic systems. Researchers have also started employing the data-driven methods in material science Pollice et al. [9]. Forecasting or predicting is also one of the promising topics in data-driven modeling [10]. For example, the data-driven methods are employed for predicting electricity consumption of a building [11] and wild fire forecasting [12].

Likewise, data-driven haptics is concerned with model-free methods that employ nonparametric machine learning techniques to learn input-output mapping functions. The aim is to model the complex physical behaviors of real objects and simulate them *in real time* for rendering in a virtual environment. The data-driven approach has been successfully employed for deformable objects [13], [14], [15], [16], [17], surface textures [18], [19], [20], [21], [22], and thermal interaction [23]. The review in this section focuses on data-driven haptics for deformable objects.

Höver et al. [13] were among the first who applied the data-driven approach to accounting for the viscoelastic behavior of a deformable object. They used the position and velocity of a haptic tool as the input features, as in the physics-based models. They employed an RBF-based interpolation method to predict the output of response force. This work was extended to modeling slip and inhomogeneous interactions on deformable objects [14], [15]. Yim et al. [16] presented a complete data-driven framework for inhomogeneous objects handling both viscoelastic and frictional responses in a unified way. For friction identification, they estimate sliding yield surfaces and anisotropic friction cones that depend on the contact position from the measured data. The frictional behavior during rendering is simulated using a virtual proxy, and its position is updated using the sliding yield surfaces while crossing the stick and slip regions. The position-dependent deformation of the inhomogeneous object is modeled using an RBF formulation that includes an additional input variable of virtual proxy. This notable improvement, however, significantly increases the amount of data required and the modeling complexity. This often makes the RBF training intractable for complex inhomogeneous real objects.

This problem of high computational cost for modeling has been addressed by a few groups. Sianov and Harders [17] proposed a feature-based learning scheme for RBF models to reduce the dimensionality of datasets. They select features using greedy kernel selection, which effectively reduces the modeling time for large datasets. This method was successfully applied to viscoelastic objects explored by two fingers.

As an alternative to RBF, we previously proposed using regression forests, a well-known nonparametric machine learning technique, to model viscoelastic deformable objects [24]. In this method, the input features consist of the current and past position samples. The RF-based model outperformed RBF-based models in terms of prediction accuracy, amount of training data, and computational time. However, when tested for rendering virtual objects, we found that the RF method did not provide stable interactions. We observed that the RF model often simulated a negative damping behavior, which essentially accumulates the energy during haptic interaction and can make the rendering unstable [3], [25]. This experience motivated us to explore better input features for RF-based models.

Recently, Abdulali et al. [26] proposed data-driven methods for hyper-viscoelastic material. In this work, deformation data are captured using external cameras. The relationship between applied force and deformation is represented by a highly efficient FEM (finite element method) model. Then, a real-time predictor is built from the FEM model and used for haptic rendering. In their next paper [27], the authors extended the framework to model the plasticity in deformation using inverse reinforcement learning.

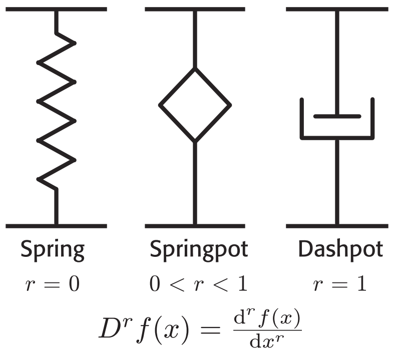

FIGURE 2. Concept of a springpot.

In this paper, we use fractional derivatives to define input features. In mechanics, FDs have long been used as an alternative method to describe viscoelastic behaviors [28], [29]. The key component is springpot (Figure 2), which can be combined with other traditional mechanical elements to model the complex dynamics of many real world objects [30]. FDs have been applied to many other research problems to explain the viscoelastic property of deformable objects [31], [32], elastomers [33], [34], and tissues [35], [36], [37]. FDs have recently been introduced to haptics. Caldiran et al. [38] studied the perceptual properties of the springpot model when rendered with different amplitudes and phases in the frequency domain.

III. DATA COLLECTION

In this paper, we focus on modeling of the viscoelastic deformation behavior of a real soft object when it is pressed and released in the normal direction. The aim is to provide highly realistic haptic responses by simulating the model and rendering the results using a force-feedback interface. Our experimental setup is designed accordingly.

A. HARDWARE

As shown in Figure 3, the experimental setup for data collection consists of a force-feedback device (Omega 3.0, Force Dimension), a load cell (DBCM-2kg, Bongshin), and a deformable object. The software is implemented using an open-source haptic rendering library, CHAI3D. The load cell is attached to the end-effector of the force-feedback device. The other end of the load cell has a tip of 6 mm diameter for interaction with the soft object. The tip is always located at the center of the object's top surface for data collection. The deformation depth is measured by the haptic device, and the force is by the load cell. The data (position and force) is updated at a sampling rate of 1 kHz.

For haptic rendering, the load cell is replaced with an aluminum handle that has a very similar size and weight to the load cell. The upper image in Figure 4 compares the sensor and the handle. The gravity compensation constant included

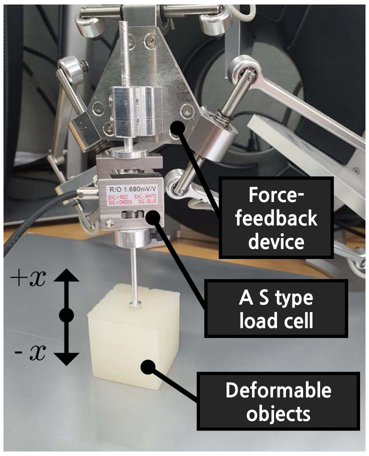

FIGURE 3. Configuration of data collection.

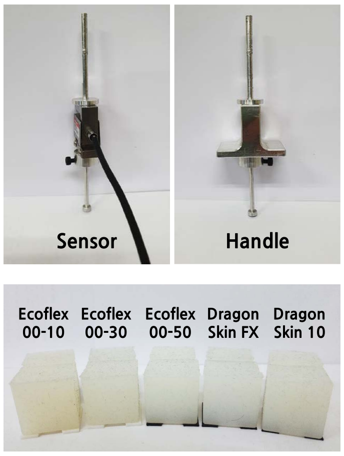

FIGURE 4. Our data collection and rendering setup (top). Cubes made of five silicon materials from Ecoflex 00-10 to Dragon Skin 10 (bottom).

TABLE 1. Silicon materials used in our work.

Material	Shore Hardness	Max. Modeling Depth* (mm)
Ecoflex 00-10	00-10	17.7
Ecoflex 00-30	00-30	14.5
Ecoflex 00-50	00-50	10.4
Dragon Skin FX	2A	7.5
Dragon Skin 10	10A	7.1

in CHAI3D for force rendering is adjusted according to the changed mass of the end effector.

We consider five different silicone materials for data collection. The materials are Ecoflex 00-10, Ecoflex 00-30, Ecoflex 00-50, Dragon Skin FX, and Dragon Skin 10, from the softest to the hardest, all from Smooth-On, Inc. Their detailed information is available in Table 1. From each silicon material, a cube with an edge length of 3.5 cm is molded and used in this work; see the lower image in Figure 4.

B. TRAINING DATA

Our data collection procedure is automated using the force-feedback device. An inverted cosine signal in (1) is commanded as a force control signal to the device's tip:

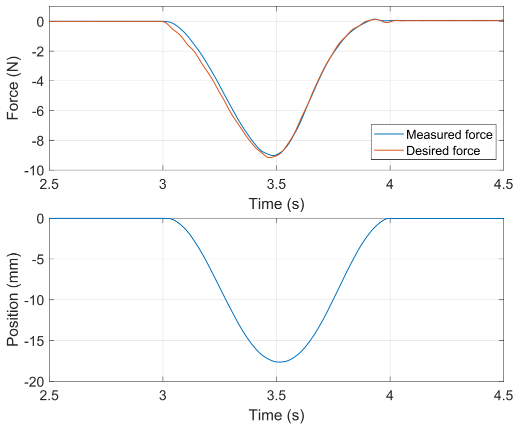

FIGURE 5. Example of collected data. The input force profile in (1) with $a = 10$ N and $\nu = 1$ Hz is applied to the object made of Ecoflex 00-10. The upper panel compares the commanded force with the measured force. The lower panel shows the measured contact position on the surface.

$$f(t) = -\frac{a}{2}(\cos(2\pi\nu t) - 1), \tag{1}$$

where a is the target (peak-to-peak) amplitude in N and ν is the indentation frequency in Hz, respectively. To characterize the object's rate-dependent deformation property for users' general exploratory behavior, we choose three amplitudes, $a \in \{2, 6, 10\}$ (in N), and nine frequencies, $\nu \in \{0.2, 0.25, 0.33, 0.5, 1, 2, 3, 4, 5\}$ (in Hz). For each pair (a, ν), we collect the interaction data for a single indentation cycle. The output force is controlled by position-derivative (PD) control to ensure stable data collection.

We collect interaction data for each object in Figure 4 using the 27 input force profiles (3 amplitudes $\times$ 9 frequencies). The total length of the training dataset is 48.85 s. An example of the collected data is provided in Figure 5. These datasets are used for training interpolation models (Section IV).

C. VALIDATION DATA

We collect another dataset for each object to validate the trained interpolation models. The validation set must include sufficiently complex and general conditions that can be interpolated from the range of the training data. We generate a complex force signal to indent the object by computing

$$f_i(t) = -\frac{a_i}{2}(\cos(2\pi\nu_i t) - 1), \tag{2}$$

for $t \in [0\text{s}, 15\text{s}]$. a_i and ν_i are randomly chosen between 0.2 N and 1 N and between 0.2 Hz and 5 Hz, respectively, for $i \in \{1, 2, \cdots, 5\}$. The values of $f_i(t)$ after the last zero-crossing before 15 s are zero-padded to make any sum of multiple $f_i(t)$ terms end with zero. Then, we compute

$$f_{sum}(t) = \sum_{i=1}^{5} f_i(t), \tag{3}$$

$$f(t) = \frac{10}{\max f_{sum}(t)} f_{sum}(t). \tag{4}$$

The five element signals are added in (3). The result is normalized in (4) by the maximum force amplitude (10 N) used for training data collection. We make ten validation input force signals using (2)–(4) and then collect interaction data for 15 s for each input signal. We apply a zero-phase moving average filter (window size 25) to smooth both the measured position and force data. See Figure 7 for examples.

IV. MODELING

Using the collected interaction data, we learn a nonparametric mapping from the input (position) to the output (force). In this paper, we propose new input features using fractional derivatives, which is free from the rendering instability problem for RF models aforementioned in Section II.

A. INPUT FEATURES

The input features are derived from the fractional derivatives of deformation position. Here, we use the sampled sequence of position $x[n]$ and force $f[n]$ from the continuous position $x(t)$ and response force $f(t)$, where n is the discrete time index. We make an input feature vector as follows:

$$\mathbf{X}[n] = \left(D^{r_1}x[n], D^{r_2}x[n], \cdots, D^{r_{10}}x[n]\right), \tag{5}$$

where $D^r x[n]$ denotes the r^{th}-order fractional derivative of $x[n]$ and its order $r_i \in [0, 1]$ $(i = 1, 2, \cdots, 10)$. See Appendix A for how we compute $D^r x[n]$. The measured response force sequence is denoted by $f[n]$.

The training dataset of each deformable object includes the 27 position-force signal pairs measured for the 27 input force profiles of different amplitudes and frequencies (Section III). We represent them by $x_k[n]$ and $f_k[n]$, respectively $(k = 1, 2, \cdots, 27)$. Then, we compute $\mathbf{X}_k[n]$ using (5):

$$\mathbf{X}_k[n] = \left(D^{r_1}x_k[n], D^{r_2}x_k[n], \cdots, D^{r_{10}}x_k[n]\right), \tag{6}$$

for all $n \in \{1, 2, \cdots, N_k\}$ where N_k is the number of the recorded samples in $x_k[n]$. For each n, $\mathbf{X}_k[n]$ (input feature) is paired with $f_k[n]$ (output force). For compact notation, we stack $\mathbf{X}_k[n]$ and $f_k[n]$ for all n, such that

$$\mathbf{X}_k = (\mathbf{X}_k[1], \mathbf{X}_k[2], \cdots, \mathbf{X}_k[N_k]), \tag{7}$$
$$\mathbf{f}_k = (f_k[1], f_k[2], \cdots, f_k[N_k]). \tag{8}$$

Finally, $\mathbf{X}_k$ and $\mathbf{f}_k$ are stacked over k, i.e., all the 27 training signal sets, such that

$$\mathbf{X}^* = (\mathbf{X}_1, \mathbf{X}_2, \cdots, \mathbf{X}_{27}), \tag{9}$$
$$\mathbf{f}^* = (\mathbf{f}_1, \mathbf{f}_2, \cdots, \mathbf{f}_{27}), \tag{10}$$

where $\mathbf{X}^* \in \mathbb{R}^{(10\times N_k\times 27)}$ and $\mathbf{f}^* \in \mathbb{R}^{(1\times N_k\times 27)}$. Consequently, $(\mathbf{X}^*, \mathbf{f}^*)$ constitutes the full training dataset.

B. MODEL TRAINING

For each object, we estimate an interpolation function Ψ,

$$f[n] = \Psi\left(\mathbf{X}[n]\right), \tag{11}$$

which predicts the response force $f[n]$ from the FD feature vector $\mathbf{X}[n]$. To this end, we learn a regression forest

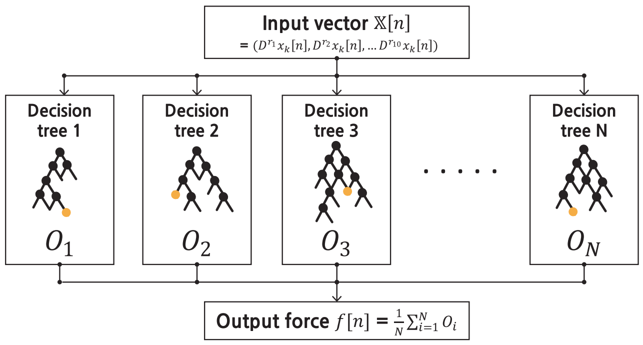
Input vector $\mathbb{X}[n]$
$= (D^{r_1}x_k[n], D^{r_2}x_k[n], \ldots D^{r_{10}}x_k[n])$
Decision tree 1
Decision tree 2
Decision tree 3
Decision tree N
O_1
O_2
O_3
O_N
Output force $f[n] = \frac{1}{N}\sum_{i=1}^{N} O_i$

FIGURE 6. Structure of regression using a random forest.

model on the training dataset prepared for the object. An RF consists of many decision trees and can be used for regression [39]. As illustrated in Figure 6, an input vector is fed to many decision trees in parallel, and their output values are averaged to obtain the final output; refer to [39] for more on RFs.

For RF model training, we choose the following parameters: ten orders of FDs in (5) $= \{0.05, 0.10, \cdots, 0.50\}$, the number of decision trees $= 100$, and the stopping criteria $=$ minimum five samples at leaf nodes. We use FDs of many different orders to fully utilize the strength of RFs for automatically searching for meaningful features in a large feature space. Each of the ten orders explicitly represents the different time-dependent behavior (see Figure 19 in Appendix A) so that RF training can choose the best rate-related features. For implementation, we use the source codes available in [40]. The resulting interpolation models are denoted by RF-FD models.

For comparison, we also train a radial basis function on input features consisting of FDs:

$$\mathbf{X}[n] = \left(D^{r_1}x[n], D^{r_2}x[n]\right). \tag{12}$$

To choose the best two orders for each object, we train RBF models with all pairs of orders chosen from the ten orders used for the RF-FD models. Then, we select the pair that results in the lowest root mean squared error (RMSE) for the validations datasets. The chosen FD orders are shown in Table 2, and the resulting models are called RBF-FD models. Note that including more FD terms in the input features significantly increase the computation time for RBF model training. The computational results of such cases are not compared in this paper.

TABLE 2. Fractional derivative orders chosen for RBF models.

Material	r_1	r_2
Ecoflex 00-10	0.30	0.35
Ecoflex 00-30	0.15	0.20
Ecoflex 00-50	0.10	0.15
Dragon Skin FX	0.10	0.15
Dragon Skin 10	0.10	0.15

Lastly, we train another RBF model on the conventional input features of position and velocity:

$$\mathbf{X}[n] = \left(D^0 x[n], D^1 x[n]\right) = (x[n], v[n]) \,. \qquad (13)$$

The velocity $v[n]$ is estimated using the first-order adaptive windowing method [41]. These models are named RBF-PV, as an implementation of the previous state-of-the-art model [13] for comparison.

All the models are trained on a reduced dataset randomly sampled from the original training dataset, as in [13]. We use only 20% of the training data after repeated tests. The size of the training dataset critically affects the time required for model training. The 20% ratio of the training data to use was determined by empirical tests. Using larger ratio tended to make training of RBF models infeasible.

The two RBF models, RBF-FD and RBF-PV, use the following settings for training: the number of kernel points = 100 and the cubic spline interpolation method. These numbers, as well as the number of trees for RF models, were chosen considering the update rate of haptic rendering of the trained models. As the model size increases, the computation time of rendering becomes longer, which degrades the stability of haptic rendering.

C. MODELING PERFORMANCE

Examples of the force curves predicted by the RF-FD, RBF-FD, and RBF-PV models for all the objects are presented in Figure 7 with the measured force curves. They are chosen from the modeling results of the ten validation input profiles (Section III-C) to represent the average performance. The lower panels in Figure 7 show the errors between the measured and predicted force curves. Generally, the output force profiles closely match the measured force responses for all of the three data-driven models. The absolute prediction errors are less than 0.4 N for all the objects, except Ecoflex 00-10 (the softest material). For Ecoflex 00-10, the absolute errors sometimes increase to around 1 N.

We consider the absolute percentage error (APE) as the difference between the measured and predicted forces divided by the measured force. For perceptual similarity, the APE should lie below the just noticeable difference (JND) of human force perception. Figure 8 presents the APE vs. the measured force for each object's trained models. Each plot also shows the JND curve of force perception for comparison. The JND curve is initially very large and then rapidly decreases as the reference force increases, finally plateauing to the constant Weber fraction (10%) [42]. The APEs for all the models lie below the JND curve for the reference force larger than 2 N for all the objects, which is generally acceptable [26], [43]. Among the three models, the two FD methods show lower APEs than RBF-PV.

For each object, we also compute the RMSEs averaged over the ten validation datasets. The results are shown in Figure 9. RF-FD and RBF-FD, trained on the fractional derivatives, result in slightly lower RMSEs than RBF-PV trained on the standard position and velocity data. RF-FD and RBF-FD have very similar RMSEs for all the objects.

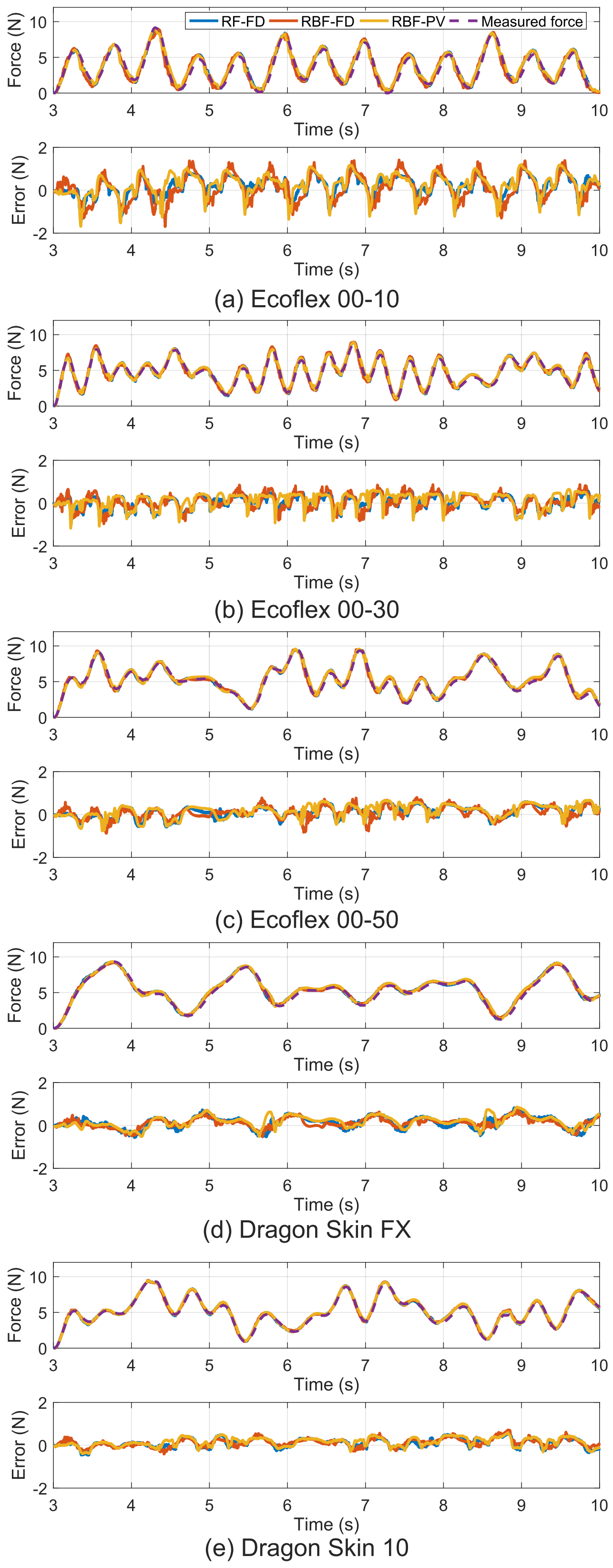

FIGURE 7. Exemplar modeling results on validation signals.

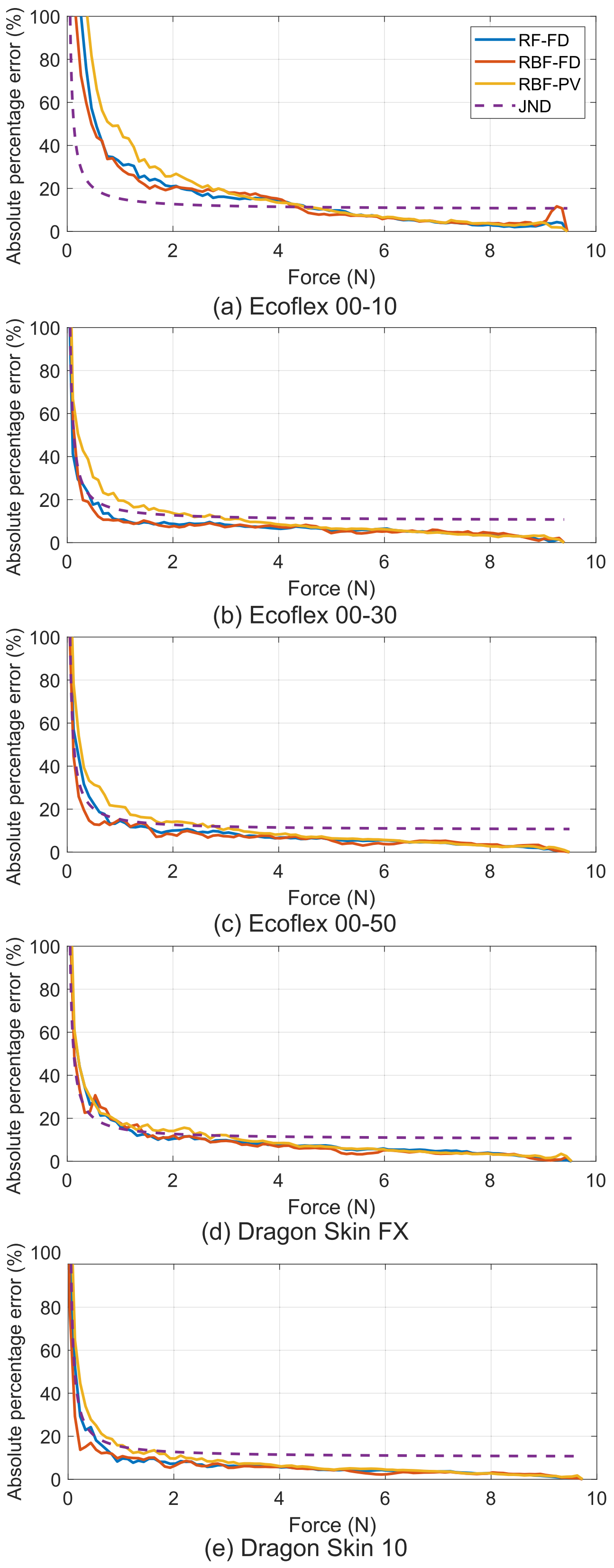

FIGURE 8. **Absolute percentage errors on the validation dataset.**

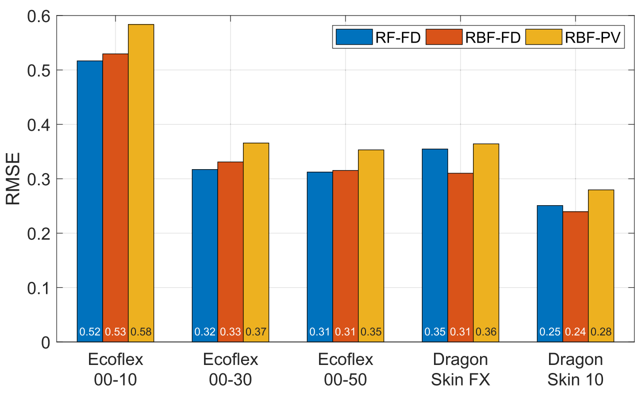

FIGURE 9. **RMSEs of the three interpolation models.**

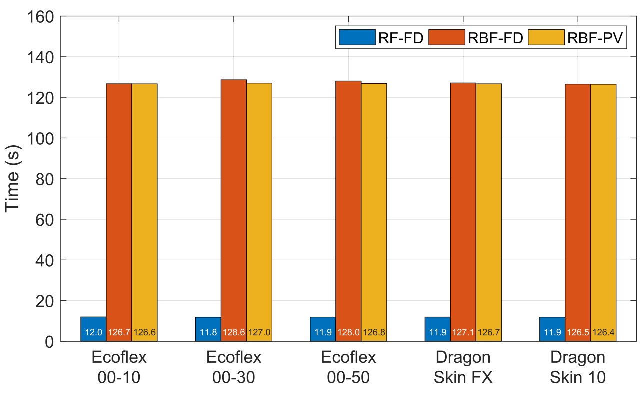

FIGURE 10. **Average modeling times of the three interpolation models.**

Last, we report the modeling computation time for both feature computation and model training. The measurement was performed on a regular PC (Windows 10, i7-11700k CPU, 32 GB memory) using Matlab. Results are shown in Figure 10. The computation times were similar regardless of the real object for all three models. Their means were 11.9 s, 126.7 s, and 126.7 s for RF-FD, RBF-FD, and RBF-PV, respectively. Therefore, the RF-FD method requires less computation by more than ten times than the two RBF methods. The actual modeling time will decrease in proportion to the number of surface contact points when we extend our modeling approach to inhomogeneous objects as in [16].

The above results indicate that RF-FD provides similar accuracy to the two RBF methods at greatly superior modeling cost. Note that the prediction accuracy and computation time can be further improved if the parameters are optimized per object. For example, Figure 11 shows the variable importance of the ten FD features obtained from the RF training for the five real objects. A variable importance is a relative indicator of how much information the corresponding feature provides to a random forest. The plots show that fractional derivatives of low orders are dominant for hard objects (low viscosity), whereas those of medium orders (0.3-0.5) are more important for soft objects (high viscosity). Including higher order terms close to 1.0 is not necessary for the objects tested, although it may be for highly viscous objects. We may further improve the prediction and/or computation

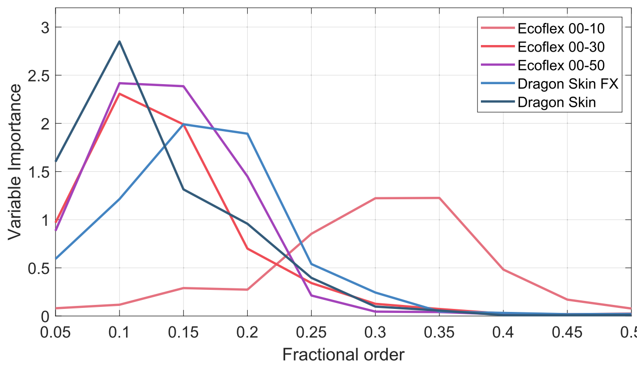

FIGURE 11. Variable importance from RF training results.

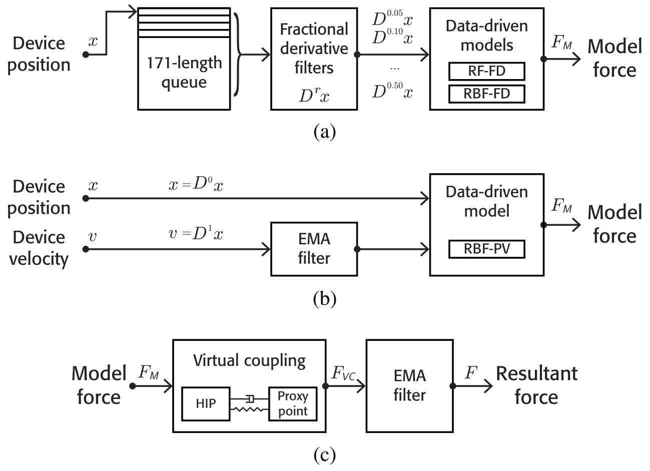

FIGURE 12. Procedure of haptic rendering. Data-driven model simulation for RF-FD and RBF-FD (a) and for RBF-PV (b). Virtual coupling to improve rendering stability (c).

performance by using only the features of high importance for each object.

V. HAPTIC RENDERING

For haptic rendering, we recreate the deformation dynamics of a real object by simulating its data-driven model and then generating the response force. To this end, the end-effector of the haptic device is replaced by the handle shown in Figure 4.

Our haptic rendering program is implemented under Microsoft Windows using C++ and CHAI3D, similar to the data collection program. Since we train RF and RBF models using MATLAB, the resulting models are loaded into the haptic rendering program. A loader file for RF models includes all information required to construct and simulate a regression forest. Given an input feature vector, all decision trees in the RF are traversed, and their results at the leaf nodes are averaged to determine the final output force. A loader file for RBF models delivers information about the centers of radial basis functions and their weights. Using these values, our program computes the rendering force for an input feature vector of RBF models.

The overall procedure of haptic rendering is depicted in Figure 12. The position of the haptic tool is denoted by x. As shown in Figure 12a, the past position samples for 171 ms are queued in a vector. It is convolved with a fractional derivative filter of the same length to compute fractional derivative features for RF-FD and RBF-FD models. The FD filter can be precomputed because our models use the fixed fractional orders (0.05, 0.1, $\cdots$, 0.5), fixed time difference (1 ms), and limited length (171); see Appendix A. The FD features are input to a data-driven model (RF-FD or RBF-FD) to compute the virtual model force f_M. For RBF-PV computation, shown in Figure 12b, we use the tool velocity v provided by the haptic device's driver. Its output is smoothed using an exponential moving average (EMA) filter with the weighting factor α for the current term of 0.7. These position and velocity are the input to an RBF-PV model, which calculates the output force f_M.

Algorithm 1 Virtual Coupling Algorithm

```
function virtual_coupling($x, v, f_M$)
    if $x_{proxy} = NULL$ then
        $x_{proxy} \leftarrow x$
        $v_{proxy} \leftarrow v$
        $f_{VC} \leftarrow 0$
    else
        $f_{VC} \leftarrow -k(x_{proxy} - x) - b(v_{proxy} - v)$
        $f_{proxy} \leftarrow f_M + f_{VC}$
        $a_{proxy} \leftarrow f_{proxy}/m$
        $v_{proxy} \leftarrow v_{proxy} + a_{proxy}\Delta t$
        $x_{proxy} \leftarrow x_{proxy} + v_{proxy}\Delta t$
    end if
    return $f_{VC}$
end function
```

The data-driven model output f_M is used as the input to a virtual coupling algorithm to improve rendering stability, as depicted in Figure 12c. Its computational steps are specified in Algorithm 1[1]. In the pseudocode, x_{proxy}, v_{proxy}, and a_{proxy} denote the virtual proxy position, velocity, and acceleration, respectively. x_{proxy} is initialized with $NULL$ before the initial contact. f_{proxy} is the force applied to the virtual proxy to update its movement, and f_{VC} is the force output of the virtual coupling. Δt is the sampling time. The algorithm simulates the simple dynamics of a virtual spring-damper system connecting the device position and the virtual proxy. The virtual proxy is treated as a quasi-static point mass, and its movement is simulated accordingly. The algorithm returns the coupling force between the two points. We use the following parameters: spring constant $k = 1.5$ N/mm, damping coefficient $b = 20$ Ns/mm, and mass $m = 50$ g, which are manually tuned. The output force f_{VC} of virtual coupling is smoothed further by an EMA filter with $\alpha = 0.7$. This final output force f is sent to the force-feedback device for haptic rendering. The effects of virtual coupling and EMA filtering are illustrated in Figure 13 using an example.

[1]We thank Arsen Abdulali and Seokhee Jeon, the authors of [26], for providing the working codes of virtual coupling for deformable objects.

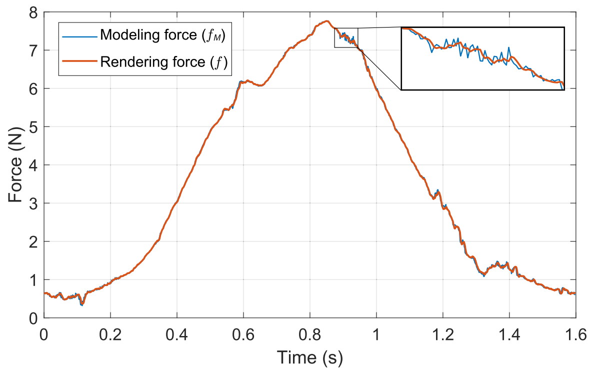

FIGURE 13. Comparison between the force computed from data-driven models and its smoothed force by virtual coupling and EMA filtering.

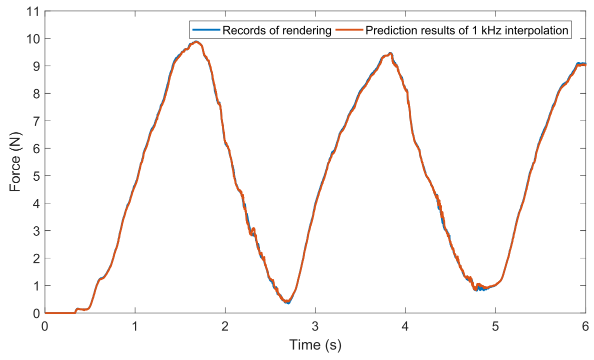

FIGURE 14. Time-force curves for the actual rendering results of the RF-FD model for Ecoflex 00-50 and the resampled results at the sampling time of 1 ms.

TABLE 3. Update rates (Hz) of the three data-driven models.

Material	RF-FD	RBF-FD	RBF-PV
Ecoflex 00-10	787.5	993.4	1984.7
Ecoflex 00-30	794.7	988.6	1980.1
Ecoflex 00-50	792.7	990.3	1983.4
Dragon Skin FX	781.4	990.3	1982.1
Dragon Skin 10	785.0	988.9	1982.9
Mean	788.3	990.3	1982.6

The above procedure is executed repeatedly in a dedicated thread. Table 3 shows the three data-driven models' update rates measured using a regular PC (Windows 11, i7-11700k CPU, 32 GB memory). The update rates are similar regardless of the real object used for modeling. RBF-PV shows the fastest update rate (mean 1982.6 Hz), followed by RBF-FD (mean 990.3 Hz). This difference is due to the use of FD features. RF-FD shows the lowest update rate (mean 788.3 Hz), indicating that rendering force computation using RF models using 10 FD features is slower than RBF models using 2 FD features by approximately 20%. Nonetheless, all of these update rates are sufficient for the haptic rendering of relatively soft deformable objects.

When computing FD features, we use the FD coefficients precomputed assuming 1-ms sampling time. The average update times for RF-FD and RBF-FD are 1.27 ms and 1.01 ms (Table 3). Thus, we need to examine whether the force output errors caused by the sampling time difference in RF-FD are significant. RF-FD has a sampling time of 1.27 ms, and the derivative orders that have high variable importance are lower than 0.4 (Figure 11). The errors in the value of the fractional derivative are negligible, according to Figure 20 in Appendix A. Thus, the slightly lower sampling time of RF-FD is expected not to cause practically noticeable problems. For confirmation, we resample the actual rendering results of position and force to have the exact 1-ms sampling time by linear interpolation and compare them with the rendering results. An example is shown in Figure 14 for the RF-FD model of Ecoflex 00-50. The actual rendering results with approximately a 1.27 ms sampling time and the resampled results with the exact 1-ms sampling time show very similar time-force curves. This close similarity is also observed for the other objects. Therefore, the RF-FD model is quite robust to the variability in the update rate of haptic rendering, at least in the range of real materials tested in this work.

VI. PERCEPTUAL EXPERIMENT

We performed a user study to validate the perceptual performance of our data-driven haptic modeling and rendering framework for viscoelastic deformable objects. Participants' task was to compare the similarity between real and virtual objects and then represent it using a number. This similarity rating paradigm has been frequently used in related studies to data-driven haptics [16], [18], [19], [21], [26], [44].

A. METHODS

1) PARTICIPANTS

We recruited 18 adults (5 females and 13 males, average age 24.2 years) for this experiment. None of them had a history of neurophysiological disorders. They were provided with written and verbal instructions about the experiment. They were paid KRW 15,000 ($\simeq$ USD 12) for compensation.

2) TASK

Participants sat comfortably in front of a computer monitor. They controlled the force-feedback device (Omega.3) with their right hands and used their left hands to enter responses using a keyboard; see Figure 15. A physical wall was placed between the haptic device and the monitor screen to block any visual cues regarding the device movements and the real/virtual objects.

Figure 16 illustrates the experimental setup. When a real object was placed on the left side, the corresponding virtual object was rendered on the right side, and vice versa. Participants explored both objects while pressing and releasing them in the vertical direction using the haptic device without a time limit. To guide the haptic exploration, the monitor screen displayed the haptic interface point (HIP), a visual guideline, and a contact line. The HIP represents the position of the device tool tip in the virtual environment. The contact line was aligned with the top faces of the real and virtual

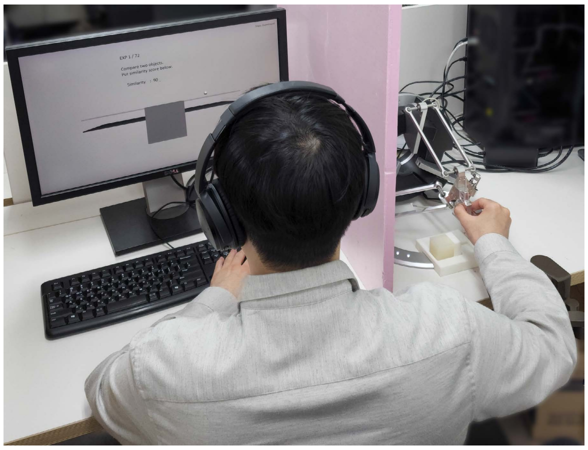

FIGURE 15. Setup of the perceptual experiment. A participant controls the force-feedback device while watching the monitor screen. The force-feedback device and deformable objects are visually blocked by the physical wall.

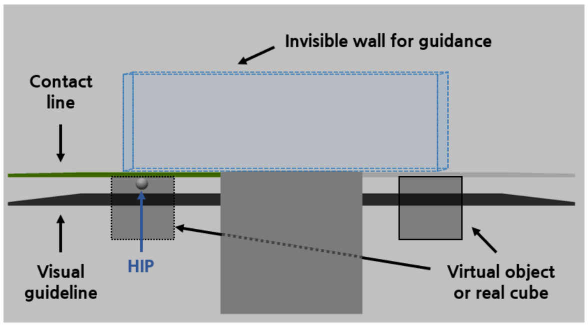

FIGURE 16. Configuration of real and virtual objects used in the perceptual experiment. Note that on the monitor screen, real or visual objects are not displayed.

objects. The visual guideline indicated the maximum permissible penetration depth (7.5 to 11 mm) used for training the models. An invisible virtual wall was also rendered to define the workspace limit while helping participants interact around the centers of the real/virtual object's top faces.

We provided two specific instructions to participants to ensure the rendering to take place in the range covered by the data-driven models. First, they were asked not to penetrate the objects exceeding the visual guideline. Otherwise, a warning message was given. This precaution was to keep the rendering force stay within the force range of the training data. Second, they were asked not to move too fast while touching the objects. An error message was shown when their movement velocity was four times larger than the maximum velocity used to generate the training data. Additionally, participants wore noise-canceling headphones that played white noise to prevent any possible effect of external sound on the experimental results.

3) CONDITIONS

We tested 21 experimental conditions: the 5 deformable objects (Ecoflex 00-10, Ecoflex 00-30, Ecoflex 00-50, Dragon Skin FX, and Dragon Skin 10) $\times$ the 3 trained models (RF-FD, RBF-FD, and RBF-PV) + 5 upper bound conditions (per material) + 1 lower bound condition. The upper bound conditions were to compare two identical real objects included to obtain maximum similarity scores. In the lower bound condition, participants compared the two most different real objects: Ecoflex 00-10 (the softest) and Dragon Skin 10 (the hardest). This condition provided a perceptual anchor for the most dissimilar case.

The haptic rendering functions for RF-FD and RBF-FD were executed as fast as possible at the update rates shown in Table 3. For RBF-PV, the update rate was fixed at 1000 Hz.

4) PROCEDURE

The experiment consisted of three sessions of identical design. In one session, each experimental condition was tested once, except the low bound condition. The low-bound condition was repeated four times. Thus, one session included 24 trials. The order of the experimental conditions was randomized per session and participant. Each participant completed all of the three sessions in a within-subject design.

In each trial, participants compared two deformable (real or virtual) objects. The experimenter manually switched the real objects under the tip of the haptic device. The haptic device generated no force while a real object was explored. The positions (left or right) of the real and virtual objects were randomly selected per trial. Participants compared the haptic responses of the two objects sufficiently without a time limit. Then, they rated the perceptual similarity between the two objects by entering a number between 0 and 100 using a keyboard.

Before performing the main sessions, participants had a training session. The training session had the same procedure as that of the main session. Participants familiarized themselves with the haptic device and the experimental task. The experimenter provided feedback on participants' behavior of executing the task whenever necessary.

At the end of the experiment, participants had a debriefing session and answered a questionnaire consisting of the following five questions:

- Q1 Was there any unstable or abnormal behavior from the haptic device during the experiment?
- Q2 What criteria have you opted for rating similarity?
- Q3 Could you distinguish between real and virtual objects?
- Q4 If your answer to Q3 is yes, then what were your criteria?
- Q5 Which factors/aspects can improve the rendering?

In addition, we recorded the trajectory of the HIP in all trials and sessions.

B. RESULTS

1) OUTLIER REMOVAL

Some participants showed inconsistent responses in the experiment. For objective assessment, we computed the standard deviations (SDs) of the similarity scores collected in

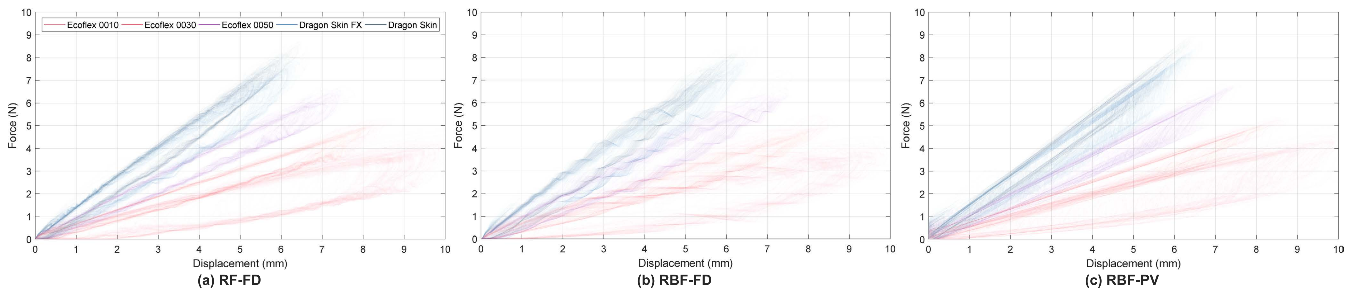

FIGURE 17. Displacement-force hysteresis curves constructed from the HIP trajectories of all participants.

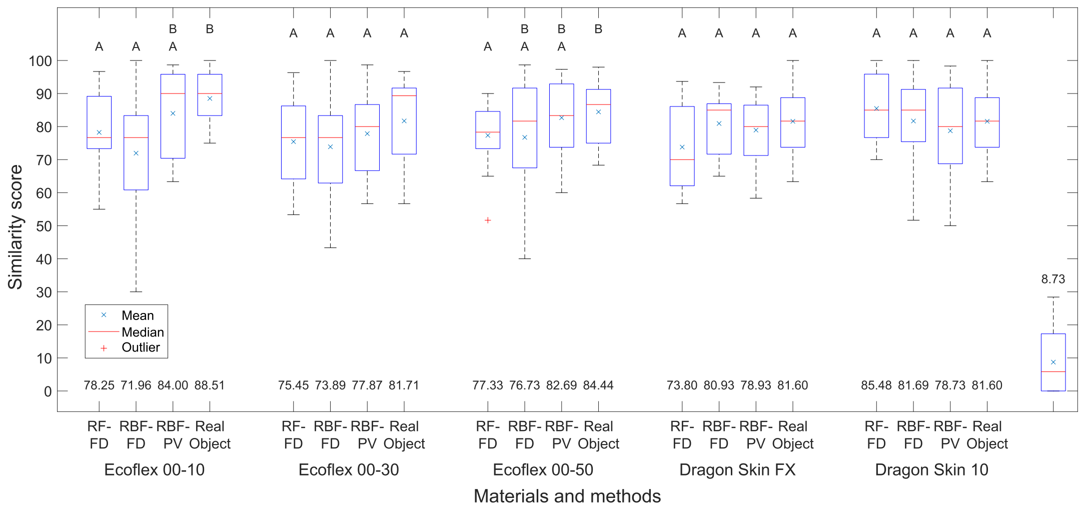

FIGURE 18. Similarity rating results of the perceptual experiment. The means of each condition are specified right above the horizontal axis. For each material, conditions without statistically significant differences are grouped using the same letters.

the same experimental conditions for each participant. Two participants, P4 and P13, showed very high inconsistency. Their SDs averaged across the 21 experimental conditions exceeded 30, while the mean SDs of the other participants were between 3.87 and 24.33. We also checked possible inconsistency due to the stimulus position by averaging the similarity scores collected when the trained model was placed left or right. Two participants, P6 and P13, failed this test. Their left-right difference scores were over 30, while the average of the other participants was 5.47. As a result, we removed the experimental data of the three participants, P4, P6, and P13, and proceeded with the data of the other 15 participants.

2) HIP TRAJECTORIES

Figure 17 shows the displacement-force hysteresis curves combining all the HIP trajectories and the command forces, collected from the interaction data of one participant, for each virtual object and data-driven model. A hysteresis curve for a typical viscoelastic object exhibits different behaviors when the object is pushed (loading) or released (unloading). In Figure 17, each curve's upper and lower parts represent the loading and unloading behaviors, forming an elliptical curve. Harder objects, e.g., Dragon Skin 10, shows a smaller hysteresis curve with higher stiffness, while softer objects, e.g., Ecoflex 00-10, shows a larger hysteresis curve with low stiffness. These behaviors are consistent with those reported in the literature for real and virtual viscoelastic objects [13], [43], [45]. Other participants resulted in similar displacement-force curves.

3) SIMILARITY SCORES

Figure 18 presents the collected similarity scores *SS* in box plots for all the 21 experimental conditions. In the upper-bound conditions (comparison between the same real objects), the mean *SS* ranged from 81.6 to 88.5, with a grand mean of 84.2. In the lower-bound condition (comparing the softest and hardest real objects), the mean *SS* was very low at 8.7. The mean *SS* values of the three data-driven models were substantially greater than the lower bound, and they were comparable to the upper bounds. For the five materials, the RF-FD models resulted in the mean *SS* values between 73.8 and 85.5, RBF-FD between 72.0 and 81.7, and RBF-PV between 77.9 and 82.7.

For statistical analysis, we applied two-way repeated-measures ANOVA with the independent variables of *Method* (RF-FD, RBF-FD, RBF-PV, or REAL) and *Material* (Ecoflex

00-10, Ecoflex 00-30, Ecoflex 00-50, Dragon Skin FX, and Dragon Skin 10) to *SS*. The data of the lower-bound condition were not included. *Method* ($F(3, 45) = 6.74, p < 0.001$) and the interaction between *Method* and *Material* ($F(12, 180) = 2.84, p = 0.001$) had significant effects on *SS*, but *Material* did not ($F(4, 60) = 2.08, p = 0.094$).

To further examine the significant interaction term, we conducted simple effect tests on the *SS* of each *Material*. *Method* had a significant effect at Ecoflex 00-10 ($F(3, 42) = 7.30, p < 0.001$), Ecoflex 00-50 ($F(3, 42) = 3.40, p = 0.026$), and Dragon Skin FX ($F(3, 42) = 2.87, p = 0.047$). We ran Tukey's post-hoc tests for the significant cases. For Ecoflex 00-10, a significant difference in *SS* existed between the two FD models and the upper bound condition. For Ecoflex 00-50, RF-FD and the upper bound condition showed a significant difference. For Dragon Skin FX, no significant differences were found among the four conditions of *Method*.

4) QUESTIONNAIRE

The participants' responses to the five questions are summarized as follows.

- Q1 Eleven out of the 15 participants sometimes felt (unstable) vibrations while exploring the virtual objects.
- Q2 All of the participants regarded the response force as the main criteria for similarity rating. Three participants also considered the change in response force when they varied the haptic exploration velocity.
- Q3 Eight participants could distinguish virtual objects from real objects.
- Q4 The major sensory cue for detecting virtual objects was oscillations that the participants sometimes experienced at deep penetration.
- Q5 Ten participants provided suggestions to improve the haptic rendering. Six of them were about removing unrealistic oscillations at deep penetration.

C. DISCUSSION

The similarity scores were bounded below by 8.7 (between the most different real objects) and above by 84.2 (between the same real objects). This resulted from the participants' general tendency to avoid extreme responses. All of the data-driven models led to high similarity scores comparable to or slightly smaller than the upper bound. The mean similarity scores of RF-FD, RBF-FD, and RBF-PV across all the objects were 78.1, 77.0, and 80.4, respectively. No significantly different cases were found among the pairs of the three data-driven models for any of the five materials (Figure 18). Therefore, all of the data-driven models and rendering methods are highly and similarly effective in replicating the deformation dynamics of a real object.

In one of the related studies [46], similar perceptual similarity scores were obtained for the same materials using the FEM-based modeling approach. The similarity scores of our RF-FD model were 75.4 and 77.3 for Ecoflex 00-30 and Ecoflex 00-50, respectively (Section VI-B). The FEM-based approach in [46] provided the similarity scores of 82.7 and 84.9 for the same two material cases. The scores of both studies are very high, although the absolute scores cannot be compared in a statistically meaningful way due to the difference in the detailed methods and environment in the experimental setups. Therefore, our work demonstrates that viscoelasticity can be modeled on a smaller amount of data in a greatly shorter time than any other models from the literature, while preserving the high accuracy.

The unrealistic oscillation problem mentioned by many participants generally occurred when the user touched a virtual object with a very fast velocity or penetrated very deep into the object. In our data-driven models, the dynamics information for such high velocity or deformation is not defined, and the rendering is likely to exhibit unexpected behaviors. In fact, this problem is common to all data-driven methods. One remedy can be extrapolation with a well-defined physics-based model if the input variables exceed the data collection ranges.

VII. CONCLUSION

In this paper, we have presented a new data-driven approach for haptic modeling and rendering of viscoelastic deformable objects. Our approach is to train a regression forest model along with features consisting of fractional derivatives of the haptic device position. This new method can reduce the computational cost of modeling to a great extent while maintaining modeling accuracy and rendering quality. This advantage is demonstrated by physical and perceptual experimental data in reference to the other state-of-the-art methods that use radial basis functions to interpolate between the position and velocity data.

For future work, we plan to extend our methods to model inhomogeneous deformable objects and multiple finger exploration. Modeling of frictional behaviors on the objects may also be considered. We also plan to adopt more up-to-date machine learning techniques, such as physics-informed deep neural networks.

APPENDIX A
COMPUTATION OF FRACTIONAL DERIVATIVES

A fractional derivative, first proposed in 1695 by L'Hospital, is a derivative of arbitrary order, real or complex. As with integer-order derivatives, the r^{th}-order fractional derivative of a function $f(x)$ is described as:

$$D^r f(x) = \frac{\mathrm{d}^r f(x)}{\mathrm{d}x^r}, \tag{14}$$

where D^r represents the operator for the r^{th}-order fractional differentiation.

For efficient computation, we use the Grünwald–Letnikov fractional derivative [47], as defined below:

$$D^r f(x) = \lim_{h \to 0} \lim_{m \to \infty} \sum_{k=0}^{m} c(r, k) f(x + (r - k)h),$$

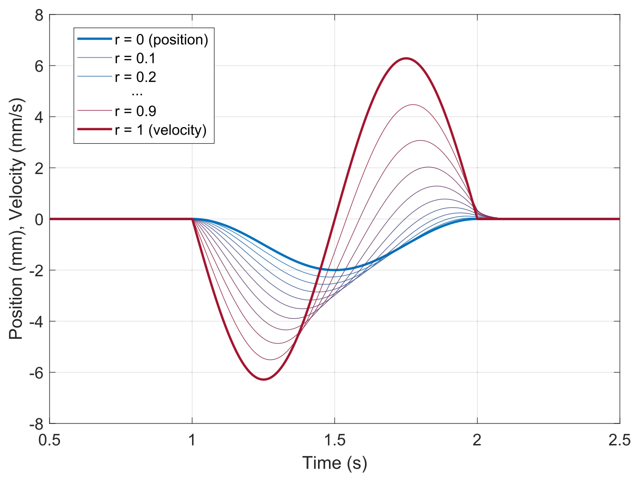

FIGURE 19. Examples of fractional derivative $D^r f(t)$ with orders from 0 to 1.

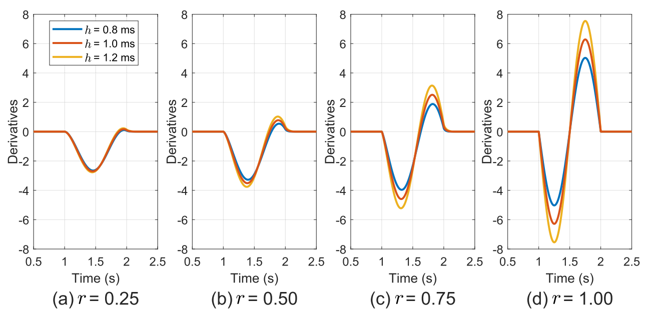

FIGURE 20. Effects of different sampling intervals on the computed values of fractional derivatives.

$$c(r,k) = \frac{(-1)^k}{h^r} \frac{\Gamma(r+1)}{\Gamma(k+1)\Gamma(r-k+1)} \tag{15}$$

where Γ is the gamma function.

In our modeling and rendering processes, the independent variable is the time t, and we approximate (15) by taking small h and large m, such that

$$D^r f(t) = \sum_{k=0}^{m} c(r,k) f(t + (r-k)h). \tag{16}$$

We set $h = 1$ ms, corresponding to the update (sampling) time of data collection and haptic rendering. We use $m = 171$ as larger values cause overflow in computing the gamma functions (double floating-point type, 64-bit CPU).

Given r, the coefficients $c(r,k)$ are precomputed for all k $(0 \leq k \leq m)$. Then, (16) works as a filter on $f(t)$ and can be easily computed in real time.

An example that demonstrates the effects of fractional derivatives is shown in Figure 19 computed using (16). The fractional derivatives show gradual transitions between $f(t) = D^0 f(t)$ and its first-order derivative $f'(t) = D^1 f1(t)$.

During haptic rendering, the update rate may vary depending on many internal and external factors. Figure 20 shows changes in the computed values of fractional derivatives for different sampling times. Compared to the reference at $h = 1$ ms, the values remain very similar for the shorter (0.8 ms) and longer (1.2 ms) sampling time when the derivative order r is low until $r = 0.50$. The differences become more notable as r increases further to 1.00.

REFERENCES

[1] K. E. MacLean, "The 'haptic camera': A technique for characterizing and playing back haptic properties of real environments," in *Proc. Annu. Symp. Haptic Interfaces Virtual-Environ. Teleoperator Syst. (HAPTICS)*, vol. 58, 1996, pp. 459–467.

[2] K. J. Kuchenbecker, J. Romano, and W. McMahan, "Haptography: Capturing and recreating the rich feel of real surfaces," in *Robotics Research*, vol. 70. Berlin, Germany: Springer, 2011, pp. 245–260.

[3] J. E. Colgate, M. C. Stanley, and J. M. Brown, "Issues in the haptic display of tool use," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst., Hum. Robot Interact. Cooperat. Robots*, Aug. 1995, pp. 140–145.

[4] F. J. Montns, F. Chinestab, R. Gómez-Bombarelli, and J. N. Kutzd, "Data-driven modeling and learning in science and engineering," *Comp. Rendus Mécanique*, vol. 347, no. 11, pp. 845–855, 2019.

[5] G. E. Karniadakis, I. G. Kevrekidis, L. Lu, P. Perdikaris, S. Wang, and L. Yang, "Physics-informed machine learning," *Nature Rev. Phys.*, vol. 3, pp. 422–440, Jan. 2021.

[6] P. W. Hatfield et al., "The data-driven future of high-energy-density physics," *Nature*, vol. 593, pp. 351–361, May 2021.

[7] M. Raissi, P. Perdikaris, and G. E. Karniadakis, "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations," *J. Comput. Phys.*, vol. 378, pp. 686–707, Mar. 2019.

[8] M. Cenedese, J. Axå, B. Bäuerlein, K. Avila, and G. Haller, "Data-driven modeling and prediction of non-linearizable dynamics via spectral submanifolds," *Nature Commun.*, vol. 13, no. 1, pp. 1–13, Feb. 2022.

[9] R. Pollice, G. dos Passos Gomes, M. Aldeghi, R. J. Hickman, M. Krenn, C. Lavigne, M. Lindner-D'Addario, A. Nigam, C. T. Ser, Z. Yao, and A. Aspuru-Guzik, "Data-driven strategies for accelerated materials design," *Accounts Chem. Res.*, vol. 54, no. 4, pp. 849–860, Feb. 2021.

[10] A. Inteha, F. Hussain, and I. A. Khan, "A data driven approach for day ahead short term load forecasting," *IEEE Access*, vol. 10, pp. 84227–84243, 2022.

[11] F. N. Soelami, P. H. K. Utama, I. N. Haq, J. Pradipta, E. Leksono, and M. Wasesa, "Data driven building electricity consumption model using support vector regression," *J. Eng. Technol. Sci.*, vol. 53, no. 3, Jul. 2021, Art. no. 210313.

[12] S. Cheng, I. C. Prentice, Y. Huang, Y. Jin, Y.-K. Guo, and R. Arcucci, "Data-driven surrogate model with latent data assimilation: Application to wildfire forecasting," *J. Comput. Phys.*, vol. 464, Sep. 2022, Art. no. 111302.

[13] R. Hover, G. Kosa, G. Szekely, and M. Harders, "Data-driven haptic rendering—From viscous fluids to visco-elastic solids," *IEEE Trans. Haptics*, vol. 2, no. 1, pp. 15–27, Jan. 2009.

[14] R. Hover and M. Harders, "Measuring and incorporating slip in data-driven haptic rendering," in *Proc. IEEE Haptics Symp.*, Mar. 2010, pp. 175–182.

[15] A. Sianov and M. Harders, "Data-driven haptics: Addressing inhomogeneities and computational formulation," in *Proc. World Haptics Conf. (WHC)*, Apr. 2013, pp. 301–306.

[16] S. Yim, S. Jeon, and S. Choi, "Data-driven haptic modeling and rendering of viscoelastic and frictional responses of deformable objects," *IEEE Trans. Haptics*, vol. 9, no. 4, pp. 548–559, Oct. 2016.

[17] A. Sianov and M. Harders, "Exploring feature-based learning for data-driven haptic rendering," *IEEE Trans. Haptics*, vol. 11, no. 3, pp. 388–399, Jul. 2018.

[18] J. M. Romano and K. J. Kuchenbecker, "Creating realistic virtual textures from contact acceleration data," *IEEE Trans. Haptics*, vol. 5, no. 2, pp. 109–119, Apr. 2012.

[19] H. Culbertson, J. Unwin, and K. J. Kuchenbecker, "Modeling and rendering realistic textures from unconstrained tool-surface interactions," *IEEE Trans. Haptics*, vol. 7, no. 3, pp. 381–393, Jul./Sep. 2014.

[20] S. Shin and S. Choi, "Geometry-based haptic texture modeling and rendering using photometric stereo," in *Proc. IEEE Haptics Symp. (HAPTICS)*, Mar. 2018, pp. 262–269.

[21] R. H. Osgouei, S. Shin, J. R. Kim, and S. Choi, "An inverse neural network model for data-driven texture rendering on electrovibration display," in *Proc. IEEE Haptics Symp. (HAPTICS)*, Mar. 2018, pp. 270–277.
[22] J. B. Joolee and S. Jeon, "Data-driven haptic texture modeling and rendering based on deep spatio–temporal networks," *IEEE Trans. Haptics*, vol. 15, no. 1, pp. 62–67, Jan. 2022.
[23] H. Choi, S. Cho, S. Shin, H. Lee, and S. Choi, "Data-driven thermal rendering: An initial study," in *Proc. IEEE Haptics Symp. (HAPTICS)*, Mar. 2018, pp. 344–350.
[24] A. Bhardwaj, H. Cha, and S. Choi, "Data-driven haptic modeling of normal interactions on viscoelastic deformable objects using a random forest," *IEEE Robot. Autom. Lett.*, vol. 4, no. 2, pp. 1379–1386, Apr. 2019.
[25] J. E. Colgate and G. G. Schenkel, "Passivity of a class of sampled-data systems: Application to haptic interfaces," *J. Robot. Syst.*, vol. 14, no. 1, pp. 37–47, Jan. 1997.
[26] A. Abdulali, I. R. Atadjanov, S. Lee, and S. Jeon, "Realistic haptic rendering of hyper-elastic material via measurement-based FEM model identification and real-time simulation," *Comput. Graph.*, vol. 89, pp. 38–49, Jun. 2020.
[27] A. Abdulali and S. Jeon, "Data-driven haptic modeling of plastic flow via inverse reinforcement learning," in *Proc. IEEE World Haptics Conf. (WHC)*, Jul. 2021, pp. 115–120.
[28] G. W. S. Blair, "Analytical and integrative aspects of the stress-strain-time problem," *J. Sci. Instrum.*, vol. 21, no. 5, pp. 80–84, May 1944.
[29] G. W. S. Blair, "The role of psychophysics in rheology," *J. Colloid Sci.*, vol. 2, no. 1, pp. 21–32, Feb. 1947.
[30] L. B. Eldred, W. P. Baker, and A. N. Palazotto, "Kelvin-Voigt versus fractional derivative model as constitutive relations for viscoelastic materials," *AIAA J.*, vol. 33, no. 3, pp. 547–550, Mar. 1995.
[31] R. L. Bagley and P. J. Torvik, "A theoretical basis for the application of fractional calculus to viscoelasticity," *J. Rheol.*, vol. 27, no. 3, pp. 201–210, 1983.
[32] R. L. Bagley and P. J. Torvik, "Fractional calculus-a different approach to the analysis of viscoelastically damped structures," *AIAA J.*, vol. 21, no. 5, pp. 741–748, May 1983.
[33] M. Sasso, G. Palmieri, and D. Amodio, "Application of fractional derivative models in linear viscoelastic problems," *Mech. Time-Dependent Mater.*, vol. 15, no. 4, pp. 367–387, Nov. 2011.
[34] F. C. Meral, T. J. Royston, and R. Magin, "Fractional calculus in viscoelasticity: An experimental study," *Commun. Nonlinear Sci. Numer. Simul.*, vol. 15, no. 4, pp. 939–945, Apr. 2010.
[35] R. L. Magin, "Fractional calculus models of complex dynamics in biological tissues," *Comput. Math. Appl.*, vol. 59, no. 5, pp. 1586–1593, 2010.
[36] A. Bonfanti, J. L. Kaplan, G. Charras, and A. Kabla, "Fractional viscoelastic models for power-law materials," *Soft Matter*, vol. 16, no. 26, pp. 6002–6020, 2020.
[37] Y. Kobayashi, P. Moreira, C. Liu, P. Poignet, N. Zemiti, and M. G. Fujie, "Haptic feedback control in medical robots through fractional viscoelastic tissue model," in *Proc. Annu. Int. Conf. IEEE Eng. Med. Biol. Soc.*, Aug. 2011, pp. 6704–6708.
[38] O. Caldiran, H. Z. Tan, and C. Basdogan, "Visuo-haptic discrimination of viscoelastic materials," *IEEE Trans. Haptics*, vol. 12, no. 4, pp. 438–450, Oct. 2019.
[39] L. Breiman, *Machine Learning*. Cham, Switzerland: Springer, 2001, ch. 1, pp. 5–32.
[40] A. Jaiantilal. (Accessed: Dec. 4, 2018). *Randomforest-MATLAB*. [Online]. Available: https://code.google.com/archive/p/randomforest-MATLAB/
[41] F. Janabi-Sharifi, V. Hayward, and C.-S. J. Chen, "Discrete-time adaptive windowing for velocity estimation," *IEEE Trans. Control Syst. Technol.*, vol. 8, no. 6, pp. 1003–1009, Nov. 2000.
[42] R. Höver, M. D. Luca, and M. Harders, "User-based evaluation of data-driven haptic rendering," *ACM Trans. Appl. Perception*, vol. 8, no. 1, pp. 1–23, Oct. 2010.
[43] S. Jeon, "Real stiffness augmentation for haptic augmented reality," *Presence: Teleoperators Virtual Environ.*, vol. 20, no. 4, pp. 337–370, Aug. 2011.
[44] S. Shin and S. Choi, "Hybrid framework for haptic texture modeling and rendering," *IEEE Access*, vol. 8, pp. 149825–149840, 2020.
[45] S. Jeon and S. Choi, "Haptic augmented reality: Taxonomy and an example of stiffness modulation," *Presence: Teleoperators Virtual Environ.*, vol. 18, no. 5, pp. 387–408, Oct. 2009.
[46] A. Abdulali, I. Atadjanov, S. Lee, and S. Jeon, "Measurement-based hyper-elastic material identification and real-time FEM simulation for haptic rendering," in *Proc. 25th ACM Symp. Virtual Reality Softw. Technol.*, Nov. 2019, pp. 1–10.
[47] R. Scherer, S. L. Kalla, Y. Tang, and J. Huang, "The Grünwald–Letnikov method for fractional differential equations," *Comput. Math. Appl.*, vol. 62, pp. 902–917, Aug. 2011.

HOJUN CHA (Associate Member, IEEE) received the B.S. degree in computer science and engineering from the Pohang University of Science and Technology (POSTECH), in 2014, where he is currently pursuing the Ph.D. degree with the Interaction Laboratory. His research interests include modeling and rendering of deformable objects and human–computer interaction.

AMIT BHARDWAJ (Member, IEEE) received the Ph.D. degree in electrical engineering from IIT Bombay, in 2016. He was an Alexander von Humboldt Postdoctoral Fellow at the Technical University of Munich (TUM), Germany. He was a Postdoctoral Researcher at the Pohang University of Science and Technology (POSTECH), South Korea. He is currently an Assistant Professor with the Department of Electrical Engineering, IIT Jodhpur. His research interests include computer haptics, haptics for teleoperation, human haptics, signal processing, and applications of machine learning.

SEUNGMOON CHOI (Senior Member, IEEE) received the B.Sc. and M.Sc. degrees in control and instrumentation engineering from Seoul National University, in 1995 and 1997, respectively, and the Ph.D. degree in electrical and computer engineering from Purdue University, in 2003. He is currently a Professor of computer science and engineering with the Pohang University of Science and Technology (POSTECH). His research interests include haptic rendering and perception, both in kinesthetic and tactile aspects, mobile devices, automobiles, virtual prototyping, and motion-based remote controllers.