Tactile Rendering Based on Skin Stress Optimization

MICKEAL VERSCHOOR, DAN CASAS, and MIGUEL A. OTADUY, Universidad Rey Juan Carlos

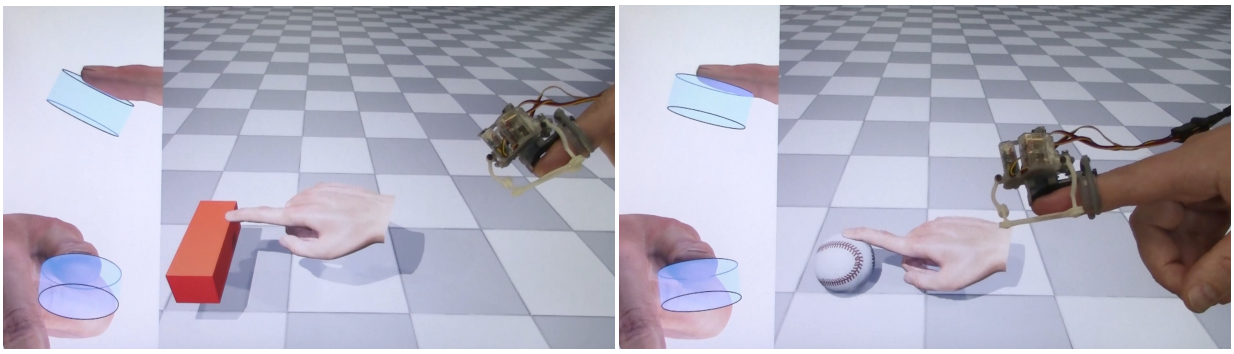


Fig. 1. Our tactile rendering method in action. A virtual hand follows the user and interacts with virtual objects. On each frame, we compute the tactile stimulus (i.e., skin stress) in this simulation, and use it to find the tactile device configuration that produces the best-matching stimulus (see insets). Then, we render this device configuration to the user.

We present a method to render virtual touch, such that the stimulus produced by a tactile device on a user's skin matches the stimulus computed in a virtual environment simulation. To achieve this, we solve the inverse mapping from skin stimulus to device configuration thanks to a novel optimization algorithm. Within this algorithm, we use a device-skin simulation model to estimate rendered stimuli, we account for trajectory-dependent effects efficiently by decoupling the computation of the friction state from the optimization of device configuration, and we accelerate computations using a neural-network approximation of the device-skin model. Altogether, we enable real-time tactile rendering of rich interactions including smooth rolling, but also contact with edges, or frictional stick-slip motion. We validate our algorithm both qualitatively through user experiments, and quantitatively on a BioTac biomimetic finger sensor.

CCS Concepts: • **Computing methodologies** → *Physical simulation*; *Virtual reality*.

Additional Key Words and Phrases: haptics, skin simulation, optimization methods

ACM Reference Format:
Mickeal Verschoor, Dan Casas, and Miguel A. Otaduy. 2020. Tactile Rendering Based on Skin Stress Optimization. *ACM Trans. Graph.* 39, 4, Article 1 (July 2020), 13 pages. https://doi.org/10.1145/3386569.3392398

1 INTRODUCTION

Tactile rendering stands for the computer-based generation of virtual touch sensations on the skin. Most works in haptic rendering simulate tool-based interaction using robotic devices, which produce kinesthetic stimulation of muscles, tendons and joints. In contrast, tactile rendering simulates direct touch interaction, by stimulating skin mechanoreceptors directly. Progress on different tactile actuation technologies in the last ten years has opened the door to virtual reality experiences where users touch virtual environments (VEs) directly with their hands [Otaduy et al. 2016].

Most research in tactile haptics has aimed at the design of actuation devices, with little attention to the design of rendering algorithms for VEs. In previous methods, tactile rendering is simplified by approximating VE interaction to some primitive with the same degrees of freedom (DoFs) as the device. However, tactile perception is a very high-dimensional spatiotemporal process, where the stimuli of mechanoreceptors are mapped to high-level percepts in a complex and yet largely unknown way. By limiting the interaction with VEs to simple primitives, previous tactile rendering methods limit the ability to explore the richness of tactile perception. To name some examples, with wearable thimbles, each finger pad is approximated as one 3D point, and the interaction of this point is directly programmed as a force vector or a surface orientation [Minamizawa et al. 2007; Prattichizzo et al. 2013; Solazzi et al. 2011]; active surfaces are programmed to reproduce predefined geometric shapes [Leithinger et al. 2015; Stanley and Okamura 2015]; and air jets or ultrasound haptics are programmed to produce touch sensations at target locations in space [Long et al. 2014; Sodhi et al. 2013].

We follow a different approach to tactile rendering. We simulate a realistic interaction in the VE, using a model of the user's hand and fingers, and thus we compute a rich target stimulus on each simulation frame. Once a target stimulus is computed, we solve an inverse mapping to the tactile device configuration, and we render this configuration to the user. We choose a mechanical characterization of tactile stimuli, using the stress distribution in the skin, and

Authors' address: Mickeal Verschoor, mickeal.verschoor@gmail.com; Dan Casas, dan.casas@urjc.es; Miguel A. Otaduy, miguel.otaduy@urjc.es, Universidad Rey Juan Carlos.

0730-0301/2020/7-ART1 $15.00
https://doi.org/10.1145/3386569.3392398

we formulate and solve the inverse mapping as skin stress optimization. To the best of our knowledge, we propose the first method that formulates and solves tactile rendering as the optimization of a mechanical characterization of tactile stimuli. Nevertheless, the optimization of the stress distribution in the skin is not trivial; we face four major challenges, which we solve through corresponding major contributions.

i) Due to hardware limitations, a tactile device can explore only a subspace of the tactile stimuli computed in a VE simulation, hence the inverse mapping from tactile stimulus to device configuration is unfortunately not well defined. To solve this issue, we formulate a constrained optimization to search the device configuration space, while producing the best-matching stress.

ii) The full stimulus, i.e., the stress distribution, produced by a tactile device on the user's skin cannot be directly measured. As an alternative, we embed a skin simulation model within our solution. Then, in the context of numerical optimization, we estimate the skin stress distribution for a particular device configuration using a simulation of device-skin interaction.

iii) The skin stress produced by a tactile device is trajectory-dependent, due to device-skin friction. To address this challenge, we stagger the computation of friction state and device configuration. On each rendering frame, we first search for an optimal device configuration while considering the friction state from the previous frame as a known input, and next we update the friction state while considering the device configuration fixed.

iv) The optimization must be executed at tens of frames per second to produce a smooth rendering output, and each optimization requires the evaluation of multiple device-skin simulations while searching for the optimal device configuration. To avoid the high computational cost of full device-skin simulations, we have designed a data-driven model of skin mechanics. This model takes as input the device configuration and the friction state, and outputs the skin stress distribution. We implement the data-driven model using a neural network, trained on simulated device-skin interactions.

Our novel tactile rendering formulation is general, but some choices of feature descriptors and system components are motivated by the capabilities of the particular device being used. We have tested our tactile rendering algorithm using a type of thimble tactile device [Chinello et al. 2015], which is capable of rendering smooth geometry under interactions of low temporal bandwidth. In Section 3 we provide detailed descriptions of the main input-output variables of our algorithm, namely the device configuration, the device-skin friction state, and the tactile stimulus (i.e., the skin stress distribution), for our benchmark tactile device.

In Section 4 we describe our data-driven model of skin biomechanics, and in Section 5 we provide the full description of our optimization algorithm for tactile rendering. In Section 7 we discuss multiple examples of tactile rendering simulations. We have tested contact with smooth and sharp features, frictional contact, and interactions that range from exploration to fine grasping. Fig. 1 depicts example VE interactions, the device configurations that produce best-matching skin stress, and the actual rendering with the physical device. In contrast to previous tactile rendering methods, our approach enables a rich VE interaction and maximizes the rendering capabilities of the tactile device. We have validated these conclusions through two perceptual discrimination experiments.

To evaluate quantitatively the accuracy of our tactile rendering algorithm, we have also tested it on a BioTac [Syntouch 2018], a biomimetic finger-shaped tactile sensor with distributed force measurement. We have recorded interactions of the BioTac with various shapes, and we have then rendered those same interactions using our algorithm and the thimble tactile device. As discussed in Section 6, we demonstrate that the stimuli produced by the device match closely the original stimuli recorded when interacting with real-world objects.

2 RELATED WORK

Haptic Rendering. Haptic rendering can be largely classified into two categories: tool-based and direct-touch interaction. The first category has dominated algorithmic research, due to the availability of stylus-type devices to render tool-based interactions with VEs. Tool-based rendering algorithms control the mechanical impedance (position or force) displayed to the user, and they are subject to stability constraints due to the delay introduced between the kinesthetic action of the device/algorithm and the action of the user [Colgate and Brown 1994]. Most algorithms approach the design of stable rendering by coupling the action of the user through a virtual spring-damper system to a virtual tool in contact with the rest of the VE. Stability is easily enforced by limiting the stiffness of the virtual coupling spring [Colgate et al. 1995], while the maximum allowed stiffness, and hence the rendering quality, can be increased by maximizing the update rate of the rendering algorithm.

Many tool-based rendering algorithms have been proposed, covering interaction with rigid objects [McNeely et al. 1999], but also deformable objects [Barbič and James 2008; Duriez et al. 2006], and differences are discussed in a survey on the topic [Otaduy et al. 2013]. Recently, research attention has been devoted to maximizing the performance and quality of the virtual tool simulation through various collision detection and resolution techniques [Wang et al. 2013; Xu and Barbič 2017; Zhao et al. 2018]. The virtual-tool rendering methodology has also been explored to provide kinesthetic interaction on the hand through exoskeletal gloves, by simulating a virtual replica of the user's hand [Garre et al. 2011].

In direct-touch interaction, skin is stimulated directly, using a large variety of stimulation technologies. Three prevalent technologies are active surfaces, ultrasound devices, and wearable thimbles. Note that we focus on technologies capable of rendering shape, and we leave out other vibrotactile devices [Choi and Kuchenbecker 2013], which aim at rendering surface texture or friction, or even shape, contact, and friction in a combined manner [Yem et al. 2016]. The spatiotemporal bandwidth of device-skin mechanical interaction in vibrotactile devices is beyond the scope of our work.

Active surfaces expose a deformable surface, composed of an array of vertical pins [Leithinger et al. 2015], or pushed by pneumatic actuators and hardened through particle jamming [Stanley and Okamura 2015]. For the latter case, Stanley and Okamura [2017] designed a pre-processing optimization method to approximate a

desired output surface using a mechanical simulation model of the device.

Ultrasound devices modulate the activation of an array of ultrasound transducers to create focal pressure points in space. Inoue et al. [2015] studied methods to pre-process the activation patterns of the transducers to optimize the pressure amplitude of pre-defined focal points, and thus reproduce the touch sensation of 3D shapes. Long et al. [2014] found a more efficient solution to this optimization problem, allowing for real-time definition of the location and amplitude of the pressure focal points. Barreiro et al. [2019] developed a method to optimize dynamically these focal points based on the pressure exchange with a fluid simulation.

Wearable thimbles move a small end-effector against the finger pad, producing a local sensation of touch. Due to hardware limitations, they offer a small number of DoFs, between one and three. The choice of DoFs favors different types of dominant feedback sensations, such as one-line contact location [Provancher et al. 2005], surface orientation [Chinello et al. 2015; Frisoli et al. 2008; Prattichizzo et al. 2013] or sliding contact and friction forces [Gleeson et al. 2010; Leonardis et al. 2015; Minamizawa et al. 2007; Schorr and Okamura 2017]. To provide virtual touch interaction with a VE, the device is mapped to a virtual point, and the position, force magnitude, and/or force direction of this virtual point are directly commanded to the device. Such simplistic rendering algorithms ignore the full stimulus produced by the device on the user's skin. Therefore, a device with one dominant feedback sensation fails to render other sensations, e.g., a device designed for rendering contact orientation [Chinello et al. 2015] fails to render sliding contact, and similarly a device designed for rendering sliding contact [Schorr and Okamura 2017] fails to render surface orientation. In our results, we demonstrate that a device designed to render surface orientation [Chinello et al. 2015] can also produce other sensations, such as sliding frictional contact, when the full stimulus on the skin is accounted for.

Some aforementioned tactile rendering methods share with ours the methodology of computing the device command by solving an optimization problem. In addition, Perez et al. [2017] proposed a timid approach to controlling a thimble device by optimizing contact conditions. They approximated with a thimble device the local contact geometry around a simulated finger. However, they did not account for the mechanical interaction between device and skin, and hence completely ignored aspects such as trajectory dependency or contact stiffness. Our method optimizes instead the tactile stimuli produced by the device on the skin, which is a far more complex problem. To the best of our knowledge, ours is the first solution that solves tactile rendering by optimizing stimuli of the skin, accounting for mechanical device-skin interaction.

With wearable thimbles and ultrasound devices, haptic stimulation is limited to the passive outer layer of skin. Under this condition, tactile rendering is not subject to stability problems due to latency; therefore, the update rate of the rendering algorithm affects the bandwidth of the interactions that can be rendered, but it does not affect stability as in kinesthetic rendering. This is an important aspect for the design and refresh rate of our algorithm.

Inverse Modeling and Motion Control. Our overall computational problem can be characterized by an outer optimization loop and an inner mechanical simulation model. Such nested problems are not unfamiliar to computer graphics, and arise at least in mechanical parameter estimation works, computational design for fabrication of objects with desired mechanical properties, or motion control of animations or robots. Since the problems may vary strongly, so do the particular solutions, but we borrow from the general knowledge in these areas to design our tactile rendering algorithm. There are two general strategies that are often applied, and which we also apply: approximation of mechanical models and staggered optimization.

Staggering is applied when the optimization must solve different sets of variables, e.g., model parameters and deformation. The optimization algorithm iterates solving for one variable set at a time, keeping the rest fixed. In our work, we apply a staggering approach to compute friction forces and the device configuration separately, and thus we handle trajectory dependency efficiently.

The use of approximate mechanical models accelerates computations within the optimization loop. Moreover, the use of the model for a particular application often enables the design of highly efficient, application-specific approximations. There are multiple approaches to the design of approximate models, mainly numerical coarsening, subspace model reduction, and data-driven models. Some example applications of model approximation in inverse modeling or motion control are numerical coarsening for fast optimization-based design of objects [Chen et al. 2017], subspace model reduction for fast interactive design of animations [Barbič et al. 2012; Hildebrandt et al. 2012], or subspace model reduction for the design of heterogeneous deformable objects [Xu et al. 2015]. We devote to data-driven models a subsection of their own below.

In our work, we solve a new optimization problem per rendering frame, on the fly. Therefore, the approximate mechanical model is subject to extreme efficiency requirements. We opt for a highly efficient data-driven model based on neural networks.

Data-Driven Models. The high computational power of modern neural networks has opened the door to accurate and efficient data-driven deformable models. The application of neural networks to accelerate deformable simulation includes approaches such as learning of constitutive material models [Wang et al. 2019], learning nonlinear corrections to linear deformations [Luo et al. 2018], and the design of nonlinear subspace deformation models using autoencoders [Fulton et al. 2019].

Another relevant use of data-driven models is hand and grasp animation. Examples include the use of hand motion data to initialize animations followed by physics-based control [Zhao et al. 2013], the animation of fingers using motion segments from pre-recorded databases [Jörg et al. 2012], or the creation of tentative data-driven grasping animations followed by spacetime optimization [Ye and Liu 2012].

To conclude, we highlight two works that build data-driven methods from tactile stimuli recorded with BioTac sensors. One is a method to predict stability of robotic grasping [Garcia-Garcia et al. 2019]. The other one is a method for vibrotactile feedback during teleoperation [Pacchierotti et al. 2016]. We perform a quantitative

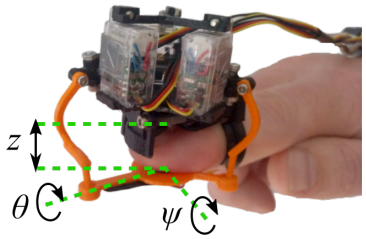


Fig. 2. The 3 DoFs thimble tactile device used as a testbed in this work. It allows control of the pitch ψ and roll θ angles, together with the distance z of a disk oriented against the finger pad.

validation of our algorithm using BioTac measurements to represent tactile stimuli, and to this end we also build a data-driven model of BioTac response.

3 DEVICE, SKIN AND FRICTION DESCRIPTORS

Our tactile rendering algorithm takes as input a target tactile stimulus, and computes as output a device configuration that produces the best-matching stimulus. Internally, the algorithm staggers the computation of the device configuration and the friction state, to handle efficiently the dependency between the device trajectory and the resulting stimulus. Furthermore, the algorithm leverages various device-skin simulation models to produce computational descriptors of tactile stimulus and friction state. Before describing the algorithm in full detail, in this section we motivate and discuss its major elements. We start with a description of the tactile device we use as testbed, and we continue with the reference skin mechanics model. Then, we motivate the choice of stress distribution as descriptor of tactile stimulus, as well as the choice of friction force distribution as descriptor of friction state.

3.1 Tactile Device

We denote with $\mathbf{x}$ the controlled DoFs of a tactile device. These DoFs may describe a kinematic configuration (as in the case of our testbed device, which we discuss next), forces, or some other type of command. We also consider arbitrary workspace constraints of the type $\mathbf{c}(\mathbf{x}) \geq 0$.

We have used as testbed a thimble tactile device with 3 DoFs [Chinello et al. 2015], shown in Fig. 2. Its end effector is a disk that is pressed and oriented against the finger pad, and it is controlled in pitch ψ and roll θ angles, together with the distance z from the finger pad. We formally define the DoFs of our tactile rendering problem as $\mathbf{x} = (\theta, \psi, z) \in \mathbb{R}^3$, with a reference system fixed on the nail.

The disk is actuated through a parallel mechanism, with three servomotors. The motors are placed on the fixed part of the device, which rests on top of the nail. The three servomotors are connected through articulated links to the end-effector disk, and they are constrained to rotations of ±40 degrees. Due to the parallel nature of the device mechanism, inverse kinematics are expressed in closed-form, and the rotational constraints of the servomotors can be expressed as 6 explicit constraints $\mathbf{c}(\mathbf{x}) \geq 0$. The full kinematics of the device are described by Chinello et al. [2015].

3.2 Hand and Finger Simulation Models

Since skin stress cannot be directly measured, we leverage instead a skin mechanics model to estimate skin stress in the various components of the rendering algorithm. In particular, we use two different models: a full-hand model in the VE simulation, to define the target tactile stimulus; and a finger-device contact model in the numerical optimization, to evaluate the stimulus produced by tentative device configurations.

The first step in the rendering algorithm is to simulate the interaction of skin with the objects in the VE, and thus estimate the target stress. We adopt a full-hand simulation method [Verschoor et al. 2018], where the virtual hand follows the tracked hand of the user, and the interaction with virtual objects is computed by solving frictional contact mechanics in real time. The hand is composed of an articulated skeleton and surrounding soft tissue. The skeleton consists of 16 bones and 70 constraints, while the soft tissue is modeled as a linear corotational material discretized into 1474 tetrahedra and augmented with strain-limiting constraints. Contact is modeled using the penalty method and springs with sliding anchors to model Coulomb friction. The full-hand simulation is computed at a visual rendering rate (30 fps), and outputs the target tactile stimulus to our tactile rendering algorithm.

In the computation of the optimal device configuration, we estimate tentative rendered stimuli by simulating contact between a finger skin model and a model of the tactile device. In this context, it is sufficient to simulate the proximity of skin that is in contact with the device. Specifically, we use a portion of the full model described earlier, limited to the distal phalanx of the index finger. This local model contains a single bone and is meshed with 154 tetrahedra. It is fixed in the nail area, to represent the mounting of the tactile device. We compute skin-device interactions by modeling the device end-effector as a rigid disk, and simulating frictional contact mechanics between this disk and the local finger model, using the same method of Verschoor et al. [2018] as mentioned above. In our staggered optimization, we use two different instances of the skin-device simulation model. A full instance of the model computes the friction state given an optimal device configuration. A data-driven approximation of the model allows fast evaluation of the rendered stimulus within the numerical optimization loop. This data-driven approximation is described in detail in Section 4.

The testbed tactile device, described above, is capable of rendering smooth geometry under interactions of low temporal bandwidth. Then, our soft-tissue simulation model provides a good compromise between visuo-haptic fidelity and real-time constraints. It captures the relevant nonlinear behavior of skin, represents well deformations of low spatial and temporal bandwidth, and handles friction smoothly. The model is, however, not suited for vibrotactile rendering, which would require much higher spatial and temporal bandwidth, and is out of the scope of this work.

3.3 Skin Stress as Descriptor of Tactile Stimuli

Four types of mechanoreceptors characterize glabrous (i.e., hairless) skin: Merkel cells, which respond to static indentations or slowly moving stimuli; Ruffini corpuscles, which respond to skin stretch; Meissner corpuscles, which respond to low-frequency skin

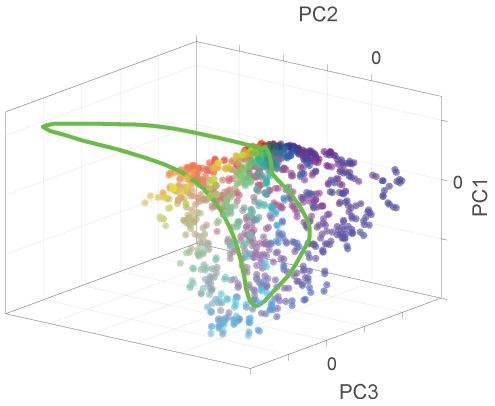


Fig. 3. Visualization of the stimulus descriptor, a.k.a. *3-PCA stimulus viz*. Each dot corresponds to the 3 main PCA components of an instance of stress descriptor. The dots have been generated by simulating device-skin interactions that span the workspace of the device, and their color represents the device configuration. The green trajectory is the 3-PCA stimulus viz for a finger simulation generated by touching a virtual object, not the simulated device. It becomes apparent that, in this case, the stimulus descriptor reaches well outside the range produced by the device.

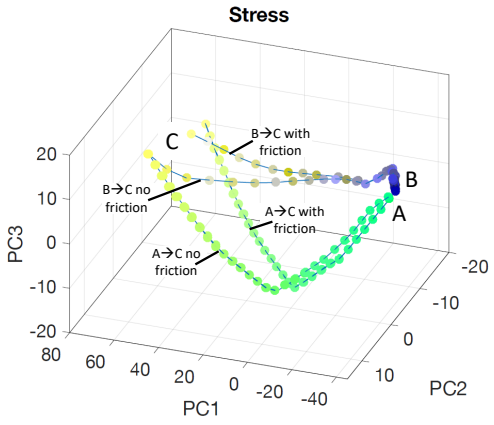


Fig. 4. 3-PCA stimulus viz of two device trajectories with and without friction. One trajectory starts from configuration A (green) and the other one from B (dark blue), and they both reach C (yellow). Without device-skin contact friction, the stress at C is the same in both cases. With friction, on the other hand, the final stress depends on the path followed by the device.

vibrations and movements across the skin; and Pacinian corpuscles, which respond to high-frequency vibrations and finely textured surfaces [Bensmaia and Manfredi 2012; Yau et al. 2016]. These mechanoreceptors are distributed throughout different parts of the skin of the finger pad: Merkel cells cover most of the base of intermediate epidermal ridges; Ruffini corpuscles are detected at the base of finger nails; Meissner corpuscles arise mainly in dermal papillary ridges; and Pacinian corpuscles are sparsely distributed in the deep dermis [Nolano et al. 2003; Paré et al. 2002]. When restricted to interactions of low temporal bandwidth, it is sufficient to consider slow adapting mechanoreceptors, i.e., Merkel cells and Ruffini corpuscles.

The response of mechanoreceptors is induced by their local deformation, which is a result of the combined forces applied on the finger. The full connection between the spatiotemporal distribution of skin deformation and the cognitive response to tactile perception is arbitrarily complex, and still unknown, but previous works have documented the influence of skin stress on the activation of both Merkel cells [Gerling et al. 2014] and Ruffini corpuscles [Grigg and Hoffmann 1982]. Furthermore, several studies on the neuroscience of touch suggest that similarity of skin stress (not necessarily an exact match) leads to perceptual similarity. In particular, similarity in the evolution of the skin deformation field has explained the perceptual similarity between different types of interactions, which induces *tactile illusions*. These include: the haptic barberpole illusion where diagonal stripes moving in one direction are perceived as moving in a different direction [Bicchi et al. 2008], and the illusion of finger motion resulting from changes of fingertip contact area [Moscatelli et al. 2014]. For these reasons, without loss of generality, we select the full distribution of stress on the finger pad skin as descriptor of tactile stimuli. Our rendering method could be easily adapted to account for perceptually inspired weighting of the stress distribution.

In practice, we sample the stress field of the finger pad. Given the soft-tissue simulation of the finger, described above, we evaluate the stress tensor on each element of the local finger mesh, rotated to the local reference frame of the nail, and we concatenate all stress tensor values (6 per element for a symmetric stress) into a large vector σ.

Throughout the paper, we show several visualizations of the stimulus descriptor. To generate the visualizations, we first execute a large number of device-skin simulations (see more details in Section 4.1), we record the stress descriptors, and perform PCA on the data. Then, we plot a stimulus descriptor as a 3D point, by clamping it to its 3 main components. We refer to this visualization as the *3-PCA stimulus viz*. Note that tactile stimuli cannot be well represented with just 3 PCA components, and in our optimization-based formulation we use a much more accurate approximation of the full stimulus, but this simple visualization suffices to convey differences across methods and settings.

Fig. 3 shows the 3-PCA stimulus viz for multiple device-skin simulations that span the workspace of the device, with the 3D points color-coded according to the device configuration $\mathbf{x}$. The limits of the workspace are evident in the plot. Fig. 3 also shows, in green, the 3-PCA stimulus viz for a finger simulation generated by touching a virtual object, not the simulated device. It becomes apparent that the stimulus descriptor reaches well outside the range produced by the device, hence the inverse mapping from stimulus descriptor to device configuration is undefined. As anticipated, we formulate and solve this inverse mapping as an optimization problem.

3.4 Friction

The stress field generated by the device depends on the current device configuration, but also on its trajectory. For low-bandwidth deformations, i.e., neglecting skin vibration, trajectory-dependent

effects are dominated by contact friction. In other words, frictional contact plays a dominant role in determining the quasi-static equilibrium configuration [Ly et al. 2018], and hence the internal skin stress. Fig. 4 shows the influence of device-skin friction on a 3-PCA stimulus viz. Two trajectories start from different device configurations and reach the same final configuration. Without friction, the final skin stress is the same in both cases. With friction, on the other hand, the final stress depends on the path followed by the device.

We identify the friction state as a notable descriptor of the effect of the device trajectory on the tactile stimulus. Specifically, using the local finger model described above in Section 3.2, we compute the friction forces between the device and the finger at any given device state. And we use as friction descriptor $\mathbf{f}$ a concatenation of the contact forces at all the surface nodes of the finger model, rotated to the local reference system of the nail.

The skin stress and the friction forces at an instant t depend on the full previous device trajectory, which we can formally express as $(\sigma(t), \mathbf{f}(t)) = f(\mathbf{x}(0), \dots \mathbf{x}(t))$. However, using a staggered strategy, if we assume that the friction state is known, we can consider the skin stress an instantaneous function of the device configuration and the friction state, i.e., $\sigma(t) = f(\mathbf{x}(t), \mathbf{f}(t))$. In essence, the effect of the device trajectory is embedded in the friction state. We leverage this staggered strategy in our algorithm to efficiently evaluate skin stimuli.

4 DATA-DRIVEN SKIN MODEL

The finger simulation model described in Section 3.2 provides the right accuracy for tactile rendering, but its high computational cost makes it impractical for our optimization-based algorithm. Instead, we observe that, for a given device configuration and friction state, we can safely ignore skin dynamics, and then the deformation of the finger, and hence its stress distribution, can be well approximated through a quasi-static function. We propose to model this function in a data-driven manner, using a neural network.

4.1 Data Collection and Subspace Projection

To train the data-driven model, we have used simulation data from device-finger interactions under random device trajectories. To generate each trajectory, we sample a pair of device configurations $(\mathbf{x}_a, \mathbf{x}_b)$ by drawing randomly from a uniform distribution of the device DoFs, and we discretize the path in between. We discard trajectories where the initial configuration $\mathbf{x}_a$ is in contact. For each step along a trajectory, we simulate device-finger contact, and if contact exists we store the device configuration $\mathbf{x}_i$, the friction descriptor $\mathbf{f}_i$, and the stress descriptor σ_i. We stop a trajectory if a maximum contact force is reached. Our data set contains 28279 samples generated from 2400 trajectories, and we have used half of the data for training and half for testing.

Since the device visits only a subspace of the stress and friction distributions, as shown in Fig. 3, we propose to project the stress and friction descriptors to respective subspaces. In this way, the data-driven skin model becomes compact and can be learned more effectively.

Given all stress descriptor samples $\{\sigma_i\}$, we execute PCA, and retain the main components that satisfy a predefined accuracy (10% in our experiments). We downscale the stress descriptor from a full-space of size 924 to a subspace of size 10. Similarly, we execute PCA on the friction descriptor samples $\{\mathbf{f}_i\}$. We downscale the friction descriptor from a full-space of size 1182 to a subspace of size 15. Without loss of generality, in the rest of the paper we rename the stimulus and friction descriptors σ and $\mathbf{f}$ to represent their corresponding subspace projections.

4.2 Neural Network Model

As discussed in Section 3.4, the friction state embeds information about device trajectory. Then, we can model skin deformation as a quasi-static problem, considering the friction state as boundary conditions. Using a neural network, we learn the function $\sigma(\mathbf{x}, \mathbf{f})$, which approximates quasi-static finger deformation mechanics. It takes as input the device configuration $\mathbf{x}$ and the (subspace) friction descriptor forces $\mathbf{f}$, to compute an estimate of the (subspace) stress distribution σ. We choose a fully-connected single-layer neural network in our implementation. For the particular problem at hand, the dimensions of the network are: 18 inputs (3 for $\mathbf{x}$ and 15 for $\mathbf{f}$), 10 outputs (for σ), and 260 neurons in the hidden layer.

Given the full set of precomputed device-finger simulations, we have used half to train the neural network and half to test its accuracy. We achieve an error of 4.2% on the reconstruction of the test skin-stress data set. We have also compared the accuracy of an alternative data-driven model $\sigma(\mathbf{x})$, which ignores the influence of the device trajectory on the resulting stress due to friction. In this case, we reach an error of 7.4% on the test skin-stress data set. As observed in Section 3.4, the influence of friction is confirmed, and it translates into a less discriminative neural network when friction is ignored.

Our numerical optimization algorithm requires the evaluation of the gradient of stimulus estimates w.r.t. device configurations, $\frac{\partial \sigma}{\partial \mathbf{x}}$. To ensure that the neural network is differentiable w.r.t. its inputs, we have chosen *softplus* as the nonlinear activation function, as a smooth counterpart to the common ReLU unit.

5 SKIN STRESS OPTIMIZATION ALGORITHM

Once all the pieces of our tactile rendering approach are defined, we present the runtime optimization algorithm. The key insight of our algorithm is to apply the staggered strategy discussed in Section 3.4, and decouple the update of device configuration and friction state. We first describe the overall algorithm, and then the numerical optimization within.

5.1 Overall Algorithm

Fig. 5 depicts schematically our rendering algorithm. On each frame, we perform a simulation of contact interaction with a VE, using the full-hand model described in Section 3.2. Thanks to this simulation, we obtain the stress distribution on the finger, which defines the target stimulus σ^*.

Given the target stimulus, we stagger the computation of the friction state and the optimal device configuration. To compute the friction state, we use a simulation of contact between a model of the tactile device and a local finger model, also described in Section 3.2. Using the device configuration $\mathbf{x}$ from the previous rendered frame,

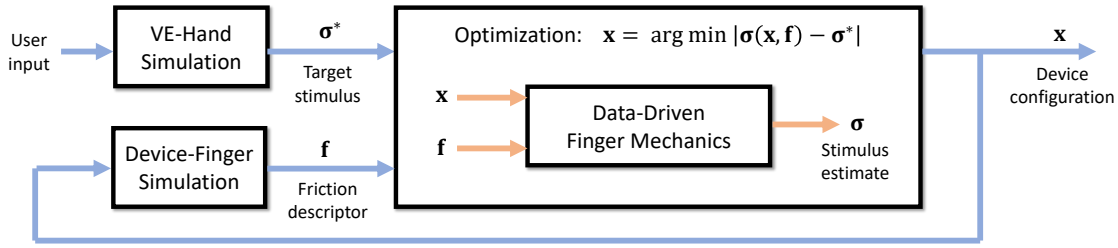


Fig. 5. Overview of our tactile rendering algorithm. On every frame, we obtain a target stimulus σ^* (i.e., skin stress) from the VE simulation. Using the device configuration $\mathbf{x}$ from the previous frame, we also compute a friction state $\mathbf{f}$. Using both as input, we search for the device configuration that produces the best-matching stimulus. This search is formulated as a constrained optimization, which evaluates a data-driven model of skin mechanics on each iteration.

we simulate the deformation of the finger and compute the current friction state descriptor $\mathbf{f}$. Assuming high temporal coherence of the friction state, we apply our proposed staggered strategy, and we can estimate tactile stimuli (i.e., skin stress) σ for various device configurations using the friction state as known external forces.

In our rendering algorithm, we search for the device configuration whose tactile stimulus matches best the target stimulus. To search efficiently for such device configuration, we leverage the data-driven skin model $\sigma(\mathbf{x}, \mathbf{f})$ described in Section 4. Formally, we formulate the search for the optimal device configuration as the constrained optimization problem

$$\mathbf{x} = \arg\min \left|\sigma(\mathbf{x}, \mathbf{f}) - \sigma^*\right|, \qquad \text{s.t. } \mathbf{c}(\mathbf{x}) \geq 0, \tag{1}$$

where $\mathbf{c}(\mathbf{x}) \geq 0$ are device workspace constraints as described in Section 3.1.

The stress and friction descriptors in (1) are expressed in the PCA subspace. Thanks to orthonormality of the PCA subspace projection, the result to the optimization (1) is exactly the same if the stimulus error is formulated in the subspace or in the full space of the stress descriptor. Therefore, at the beginning of each rendering frame we project the target stimulus from the VE simulation to the subspace, and then we perform the optimization directly in the subspace, with no need to lift stress descriptors to the full space.

5.2 Numerical Optimization Method

We solve the constrained nonlinear least-squares problem (1) using an active-set Gauss-Newton method. On each Gauss-Newton iteration, we start by identifying the active constraints, and then we linearize both the estimated stimulus and the active device constraints. As a result, we solve the iteration as a quadratic program with Lagrange multipliers. This problem has a guaranteed unique solution, as the constraint set is always feasible and the Gauss-Newton approximation of the Hessian ensures a convex quadratic objective.

Given the tentative step, we apply line-search to ensure that the cost function is reduced. At the first Gauss-Newton iteration of each rendering frame, we perform a warm start by initializing the device configuration to the result of the previous frame.

The linearization of the estimated stimulus requires the gradient $\frac{\partial \sigma}{\partial \mathbf{x}}$. To this end, we execute the differentiation of the neural-network data-driven skin model, as discussed in Section 4.2.

6 EVALUATION WITH A BIOTAC SENSOR

We have validated our rendering algorithm, both qualitatively and quantitatively, using a BioTac [Syntouch 2018] in combination with the testbed thimble tactile device. The BioTac is filled with a conductive fluid, and it contains a set of 19 electrodes that detect impedance changes as the sensor is deformed. The electrode signals form a descriptor of tactile stimulus that is comparable to the skin stress distribution, yet it can be physically measured for validation purposes.

6.1 Quantitative Validation

We have followed the same procedure as in Section 4.1 to generate training data of the interaction between the thimble device and a BioTac sensor. We have computed a subspace representation for tactile stimuli consisting of 10 PCA components (with 5% error, compared to the original 19 electrode signals). Then, we have trained a neural network that allows us to efficiently estimate tactile stimuli from device configurations.

For validation, we have captured a set of test sequences by manually interacting with real-world objects (See Fig. 6) using the BioTac sensor, while recording sensor readings. We have fed these values (*i.e.,* the target stimuli) into our tactile rendering optimization-based pipeline to obtain optimal device configurations. Finally, we have inserted the BioTac sensor into the thimble device, played back the optimized device sequences, and compared the produced stimuli with the target stimuli.

Results of the validation procedure described above are presented in Fig. 7, where we use the 3-PCA stimulus viz to compare the target (*i.e., ground truth*) and obtained stimuli of 4 different sequences. As discussed earlier in Section 3.3, color dots represent training samples for the data-driven BioTac model, and are also useful to depict the limit capabilities of the used thimble device: the dotted area is the range capable of being reproduced by our tactile device. In each subfigure, the green curve represents the sensor measurements when interacting with real world objects (*i.e.,* the target values), the red curve represents the estimated best-matching sensor readings, and the blue curve represents the actual readings of the BioTac sensor produced by playing the optimized tactile rendering sequence with the thimble device. Notice that for a training sequence (*i.e.,* for target stimuli produced with the device itself), the curves almost overlap. For interactions outside the workspace of the device,

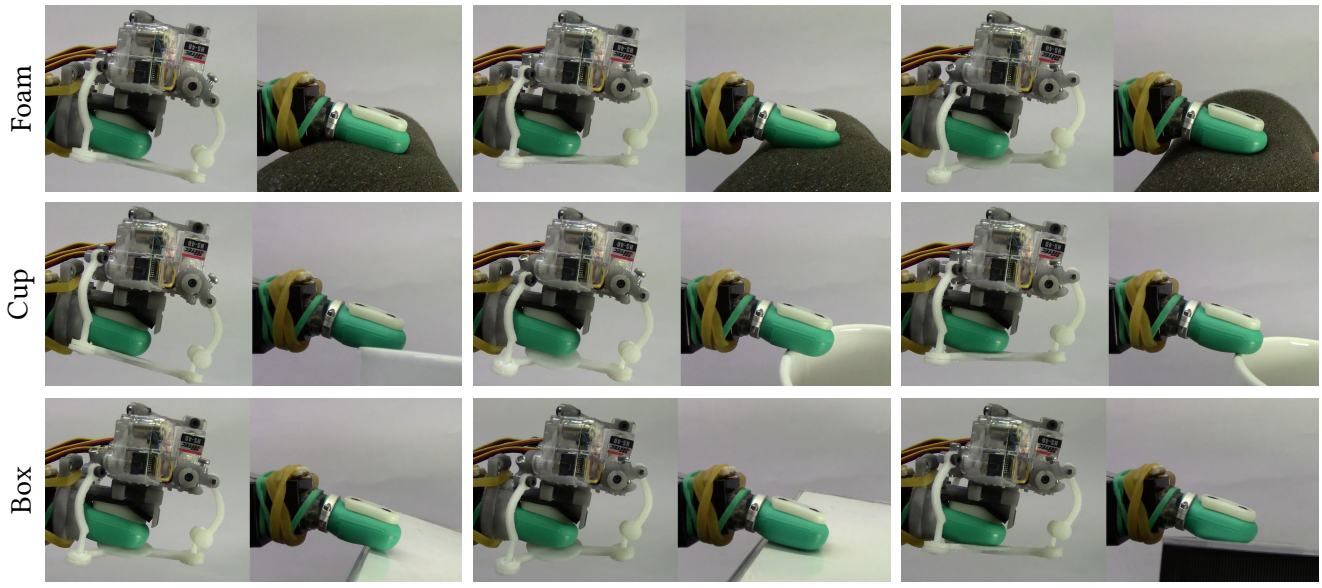
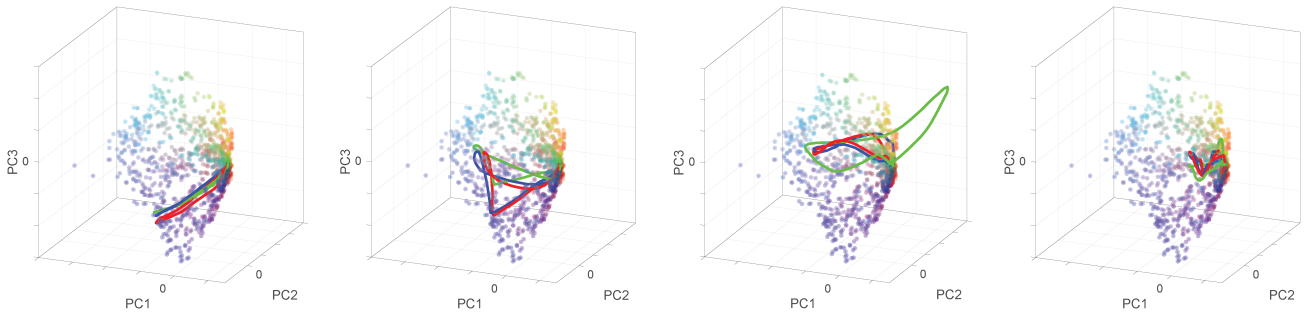


Fig. 6. Qualitative validation of tactile rendering on a BioTac sensor. We use the BioTac to interact with a set of real objects (from to top to bottom: a cylindrical foam, a cup, and a box) to obtain a sequence of target tactile stimuli. Then, we run our optimization-based algorithm to compute tactile device configurations that best match those stimuli. We show side-by-side comparisons of the rendered device configuration next to the real-world interaction that generated each target stimulus. We trained the rendering algorithm using only interactions of the BioTac with the tactile device, yet it succeeds to produce plausible renderings for unseen situations, such as contact with edges or deformable objects.



(a) Device (b) Flat surface (c) Curves and edges of a cup (d) Foam

Fig. 7. Quantitative validation using the BioTac sensor. 3-PCA stimulus viz for interactions of the BioTac with 4 different objects. The curves show: target stimuli recorded during real-world interaction (green), best-matching stimuli produced by our rendering algorithm (red), and actual stimuli measured by the sensor when playing back the optimal device configurations (blue). For data in the training set (Device), the stimuli are almost identical. And even for data far from the training set (Edge of a cup), not reachable within the workspace of the device, our algorithm produces a best-matching approximation.

our optimization produces best-matching values. The relative RMS errors between target (green) and rendered values (blue) on the 4 cases are: 7.9% (device), 9.6% (flat surface), 33.7% (cup), and 7.4% (foam). Notice also that the error between the estimated result and the actual rendered result is very small. Differences are due to a combination of error of the neural network and the PCA projection, sensor noise, and difficulties to exactly reproduce the mounting of the thimble device.

Overall, we show that the stimuli produced by the device match closely the target stimuli recorded when interacting with real-world objects, which effectively validates our method. Importantly, notice that our method also succeeds in highly challenging scenarios where the target stimulus is far from the capabilities of the device, as shown for the interaction with the edges of a cup.

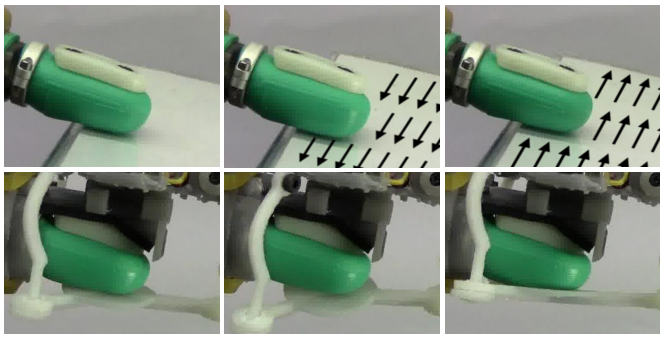


Fig. 8. Even if the testbed thimble device cannot move laterally, our rendering algorithm succeeds to approximate tangential forces through rolling motion. The arrows show the direction of motion of the box.

6.2 Qualitative validation

For the same interactions reported above, we have also carried out qualitative evaluations by observing side by side the target interactions with the real-world objects and the rendering interactions with the tactile device. As shown in Fig. 6, our algorithm succeeds to produce plausible renderings for complex unseen situations which cannot be exactly reproduced by the device, such as contact with edges or deformable objects.

One particular phenomenon where our algorithm produces interesting rendering results is sliding frictional contact, or even sticking contact with skin sliding. The testbed thimble device is simply not designed to render tangential forces. Other thimble devices replace the pitch and roll DoFs with two translational DoFs tangent to the finger pad, to produce lateral skin stretch by pulling [Leonardis et al. 2015]. However, the stimuli produced by pushing laterally on skin (or the BioTac in this case) thanks to a change in roll angle are similar to those produced by pulling. Since such data is present in the data-driven model, our optimization algorithm succeeds to approximate purely tangential forces through rolling motion of the device. An example is shown in Fig. 8.

7 VIRTUAL ENVIRONMENT RENDERING

We have also evaluated our algorithm's ability to render diverse virtual interactions. After a discussion of runtime performance, we present results of two user experiments where we compare the discrimination ability of our method and two other methods. To conclude, we discuss qualitative results during free interaction with virtual objects.

7.1 Performance

We have carried out all experiments on a 3.38 GHz Intel Core i7-4720HQ CPU with 16 GB of RAM, with the VE simulation running in parallel to the optimization algorithm. We use TensorFlow [Abadi et al. 2016] on the CPU for neural-network implementation, and a Leap Motion to track the user's hand.

We run the VE simulation, including frictional contact with the full soft hand of Section 3.2 at 30 fps. We do this in parallel to the actual rendering algorithm, which renders the simulation result from the last available simulation frame. The rendering algorithm computes the optimization of the device configuration, followed by the update of the friction state, as shown in Fig. 5. The finger simulation for the friction update takes just 5 ms.

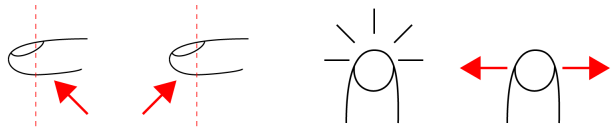


Fig. 9. Schematic representations of the contact scenarios presented to subjects of the experiments. The first two images indicate contact on the back and the front of the finger pad respectively. The last two images indicate pressing versus sliding contact.

Recall that the nonlinear constrained optimization problem for our testbed is of size 3 ($\mathbf{x}$), with 6 constraints, and each iteration requires the evaluation of a neural network with 18 inputs ($\mathbf{x}$ and $\mathbf{f}$), 10 outputs (σ), and 260 hidden neurons. Keeping the problem size compact enables successful completion of the optimization for real-time tactile rendering. Per rendering frame, the optimization takes on average 7.5 ms (std. dev. = 3.8, max = 21) and 6.3 iterations (std. dev. = 3.7, max = 16). Within each optimization iteration, the evaluation of the neural network takes on average 0.22 ms, and the evaluation of its Jacobian takes 0.57 ms.

A full simulation-based optimization would require multiple executions of the finger simulation per rendering frame. Our Gauss-Newton optimization would require a minimum of 4 executions per optimization step to evaluate the stimulus error in (1) and its Jacobian using finite differences. Moreover, given that the optimization converges on average on 6.3 iterations, a full simulation-based optimization would require 126 ms on average per rendering frame. This is in contrast to the 12.5 ms (optimization plus friction update) of our data-driven approach. The error incurred by our approach, on the other hand, amounts to 10% due to PCA compression (Section 4.1) and 7.4% due to the neural network (Section 4.2).

7.2 User Experiments

We have evaluated our rendering algorithm on two user experiments, and we have compared it to two other methods. One method is a downgraded version of our algorithm, where the skin stress descriptor is replaced by the total contact force on the finger pad, with the rest of the pipeline left unchanged. This method falls in the category of approaches that limit interaction to simple primitives, e.g., [Prattichizzo et al. 2013]. The other method is a geometric optimization of the contact interaction [Perez et al. 2017].

The experiments test the ability of methods to discriminate contact scenarios. The first experiment evaluates the success rate of discrimination of contact location (dependent variable) vs. rendering method (independent variable). The experiment displays to the user a contact with a thin cylinder at the back or at the front of the finger pad, as shown in the two leftmost images in Fig. 9. The second experiment evaluates the success rate of discrimination of pressing or sliding contact (dependent variable) vs. rendering method (independent variable). The experiment displays to the user a contact with the same cylinder, either as a simple press and release, or as a lateral

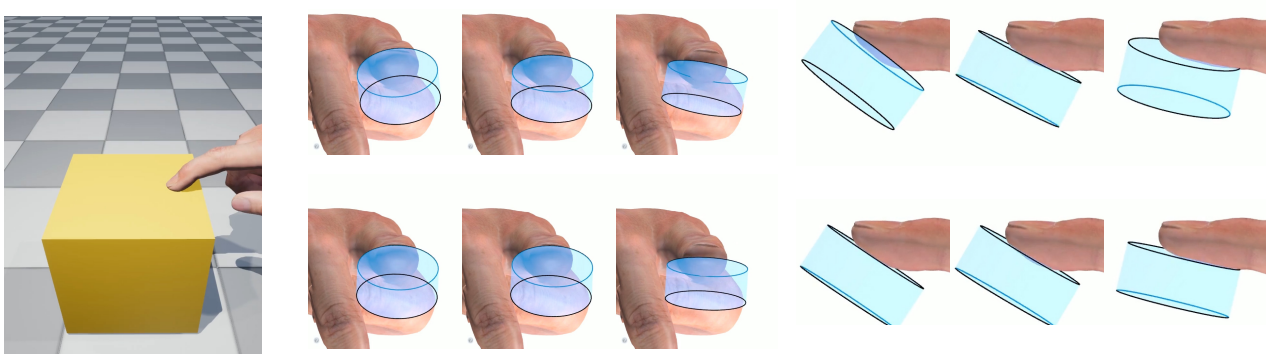


Fig. 10. Front and side visualizations of the tactile thimble device when performing a *sliding* finger motion in the scenario shown on the left. Ignoring friction in the data-driven skin stress model (bottom), the tactile rendering method cannot reproduce path-depending deformations of the finger; therefore, the tactile device remains quasi static. In contrast, our model accounts for friction and is capable of modeling path-depending deformations, and hence the device rolls to approximate the sliding sensation (top). Please see the supplementary video.

sliding motion, as shown in the two rightmost images in Fig. 9. In both experiments, we have tested three conditions, i.e., the three rendering methods, but we only compare our method separately to the other two methods, not all three conditions to each other.

In both experiments, subjects followed the same protocol. First, the device was calibrated to determine the angles and normal distance of first contact with each subject's finger pad. Then, subjects performed some free exploration of the ball scene (Fig. 11-left), with visual and haptic feedback, to get familiar with the device. Then, they executed both experiments, first contact-location and then press-vs-slide, with only one of the methods. We decided to design the experiments as independent measures (a.k.a. between groups) because, during pilot tests, we observed confounding effects of mixing the various methods for the same subject, as detectable differences between the methods could lead to erroneous cues.

In each experiment, each subject was presented with 2 tests, corresponding to the 2 stimuli to discriminate, with 8 slightly different repetitions each. All the 16 tests of each experiment had roughly the same duration, around 3 seconds, and they were presented in random order. Subjects were instructed to keep their hand and finger approximately still, and the stimuli were presented to them with haptic feedback but no visual feedback. For each test, the experimenter asked one of two possible questions: "Did you experience contact at the back of the finger pad?" or "Did you experience contact at the front of the finger pad?" in the contact-location experiment, and "Did you experience a pressing contact?" or "Did you experience a sliding contact?" in the pressing-vs-sliding experiment. The question was accompanied of a schematic image of the scenario on a monitor. The four schematic images are shown in Fig. 9. Subjects took typically less than 5 seconds to answer, and the experimenter annotated the responses.

Thirty (30) people participated in the experiments, 10 with each method. The majority of the participants were computer science students, with no previous experience with haptic devices. Table 1 shows the success rate per experiment and method. We have used Welch's t-test to compare statistically the success rate of our method vs. the other two methods. Table 1 also shows the statistical significance (p-value) of the comparisons. The results validate our initial hypotheses, and confirm the strong improvement in rendering quality achieved by our method. The use of total contact force as stimulus descriptor fails to provide any ability to discriminate the pairs of scenarios. The answers of participants are close to random. The geometric optimization method [Perez et al. 2017] performs well for the discrimination of contact location, but fails completely for the discrimination of pressing vs. sliding contact. This is no surprise, as the contact-location experiment is dominated by geometric information, whereas in the press-vs-slide experiment the skin stress distribution can vary strongly for the same contact geometry. Our method exhibits high success rates, slightly better for contact-location, which is an experiment that exploits better the degrees of freedom of the device. In this experiment, performance is slightly higher than with geometric optimization, but not statistically significant.

	Ours	Force	p-val	Geometry	p-val
contact loc.	82% (11%)	51% (8%)	4e-9	78% (11%)	0.17
press/slide	73% (14%)	53% (14%)	8e-4	53% (9%)	4e-5

Table 1. Comparison of success rates of three rendering methods on two subjective discrimination experiments ('contact location' and 'press vs. slide'). The columns show the average success rate per participant (and standard deviation) for our method, total contact force optimization, and geometric optimization [Perez et al. 2017]. The columns also include p-values comparing the performance of our method vs. the other two.

7.3 Additional Experiments

We have also performed experiments to test our tactile rendering algorithm on diverse *exploratory procedures* [Klatzky and Lederman 2003]. Fig. 1 shows camera recordings of live demos, while Fig. 10, Fig. 11 and Fig. 12 show screen captures of real-time interactions. Please see the supplementary video.

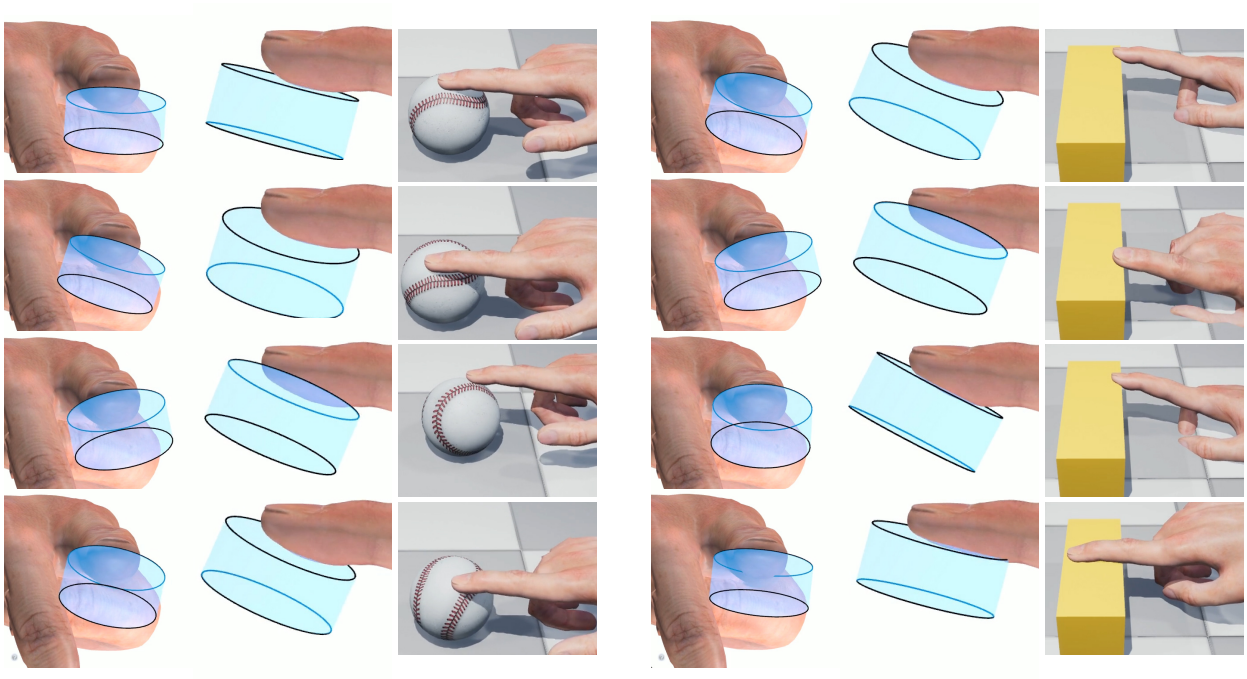


Fig. 11. Tactile exploration of objects. Our rendering algorithm is responsive to subtle changes in the stimuli computed in the VE, as shown in these two interactions with a round object and an edge. The ball is a best-case scenario for the testbed tactile device, as the interaction is dominated by the location and direction of contact. The edge, on the other hand, is a worst-case scenario, as the target stimulus falls well outside the workspace of the device. Even in such challenging case, our algorithm succeeds to approximate the stimuli, with the pitch of the device changing to render a *rounded edge* effect.

Fig. 11 and Fig. 12 demonstrate that our rendering algorithm is responsive to subtle changes in the stimuli computed in the VE. The scenarios include less challenging cases such as rolling the finger on a round object, but also challenging ones such as sliding over an edge, changing the grasping pose, or shaking a grasped object. These examples demonstrate that our rendering algorithm extrapolates beyond the expected operational space of the device, and hence it maximizes its rendering capabilities.

Fig. 10 demonstrates that the device is capable of approximating the stimuli perceived during frictional sliding motion in a VE, similar to the results found with the BioTac (See Fig. 8). Moreover, in this example we compare the rendering quality when friction is not accounted for as part of the algorithm. As discussed in Section 4.2, the neural-network data-driven model is less discriminative when friction is omitted, and in the example this translates into less smooth device motion.

8 LIMITATIONS AND FUTURE WORK

We have presented the first method to render virtual tactile stimuli by matching the stimuli computed in a simulation of a VE. Our method is based on a numerical optimization formulation, and leverages important observations about frictional contact and quasi-static deformations to achieve very high computational performance under good accuracy. We show that, with our method, even a simple 3-DoF tactile device can approximate complex contact situations such as interaction with edges, deformable objects, or sliding frictional contact.

Our work is not free of limitations, and they set interesting avenues for future work. Although our method is general, we have demonstrated it only on a device of low dimensionality, and it remains to test how it scales as the dimensionality of the device grows. However, note that, even though the device has only 3-DoFs, our data-driven skin model is of higher dimensionality, as we account for the friction state too. With multiple devices, one per finger, accuracy would be the same and the cost would simply scale linearly; such extension would enable tactile feedback of full grasping. The method could also be extended to ultrasound devices, which exhibit higher dimensionality, but the extension would require the addition of pressure-based skin deformation.

The data-driven skin deformation model relies strongly on preprocessing. Moreover, since the finger model should be adapted to each user for maximum accuracy, the training step should be redone. Statistical hand models [Romero et al. 2017] are an interesting complement, which could simplify preprocessing for multiple users.

In our experiments, we have carried out quantitative validation of the approach on the BioTac sensor. Quantitative validation on users is complex, as it would require defining meaningful metrics that can be simulated but also measured. Our work could be further

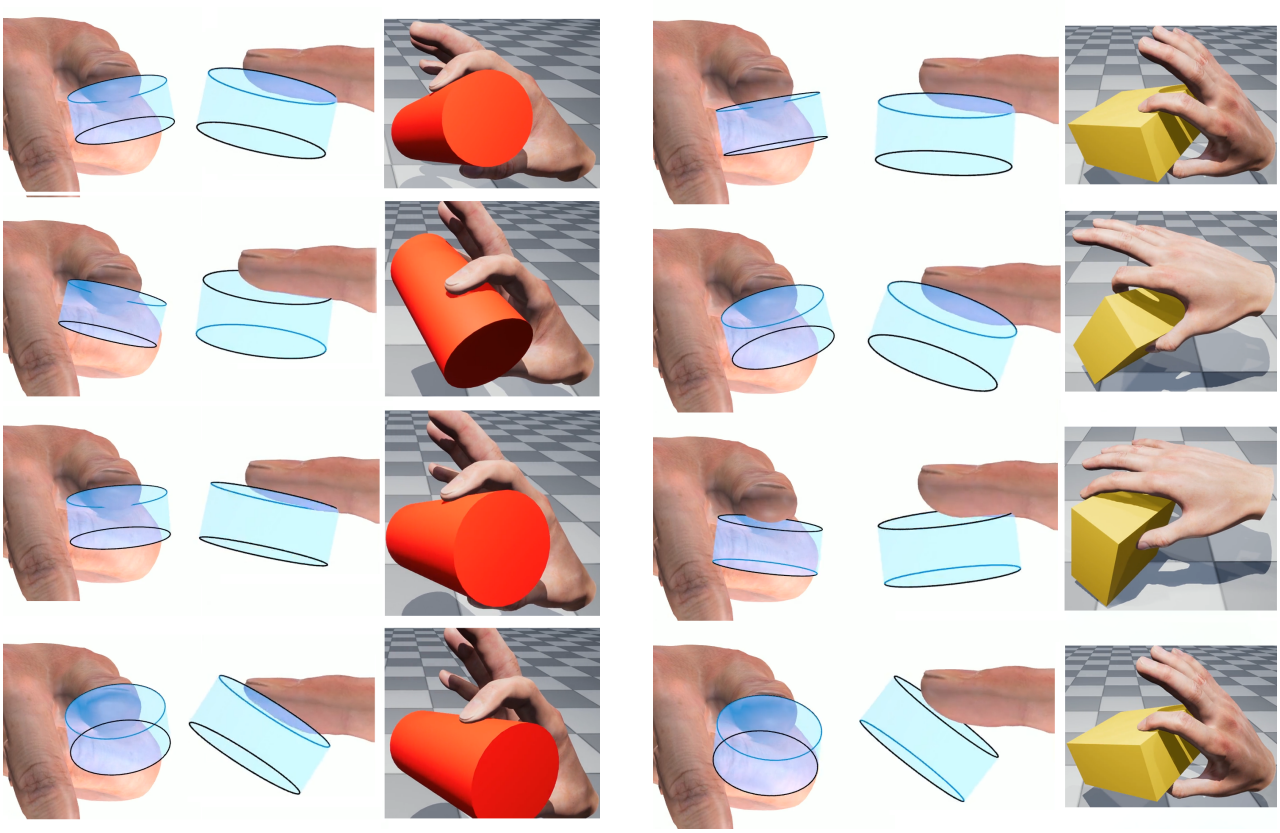


Fig. 12. Grasping objects. Our rendering algorithm succeeds to provide key tactile feedback when grasping and lifting objects. In the cylinder example, subtle changes in grasping pose produce smooth device motion. In the block example, shaking the object produces changes in friction forces and hence device motion.

extended, however, through perceptual evaluation. We have limited the evaluation to simple cases because, as shown by the results, they help to clearly extract the benefits of the method. Our experience is that the method notably outperforms other methods in full 3D interactions, such as the ones shown in the video. We did not identify cases where our method fails, but the response cannot be considered realistic when the target stimuli extend far beyond the capabilities of the device. This limitation could distort evaluation experiments on complex tasks. Perceptual evaluation could also help refine the descriptor of tactile stimuli, e.g., by weighting the stress tensor in a spatially-varying or task-dependent manner, by accounting for the stress distribution on a time window, or to search for alternative descriptors of stimuli.

Our method is limited to render effects of low spatial and temporal bandwidth, resulting from quasi-static simulation in the VE. Higher spatial bandwidth would require a finer and more accurate simulation model, probably with better handling of local nonlinearities. The subspace representations of the descriptors would also need to be lifted to higher dimensionality. Higher temporal bandwidth imposes even more severe computational difficulties, as the model should account for vibration and dynamic effects. Altogether, the extension of our approach to vibrotactile feedback, of high spatiotemporal bandwidth, is nontrivial.

ACKNOWLEDGMENTS

We wish to thank the anonymous reviewers, the participants of the study, and the members of the MSLab at URJC for their support. In particular, Stephen Sinclair for technical discussions, Alberto Martín for video production, Daniel Lobo for support with the thimble device and the VR simulation, José San Martín for help with the BioTac, and Héctor Barreiro and Jessica Illera for the user experiments. This work was funded in part by the European Research Council (ERC Consolidator Grant 772738 *TouchDesign*) and the Spanish Ministry of Science (grant RTI2018-098694-B-I00 *VizLearning*).

REFERENCES

Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, Manjunath Kudlur, Josh Levenberg, Rajat Monga, Sherry Moore, Derek G. Murray, Benoit Steiner, Paul Tucker, Vijay Vasudevan, Pete Warden, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. 2016. TensorFlow: A System for Large-Scale Machine Learning. In *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*. USENIX Association, Savannah, GA, 265–283. https://www.usenix.org/conference/osdi16/technical-sessions/presentation/abadi

J. Barbič and D. L. James. 2008. Six-DoF Haptic Rendering of Contact Between Geometrically Complex Reduced Deformable Models. *IEEE Transactions on Haptics* 1, 1 (2008), 39–52.

Jernej Barbič, Funshing Sin, and Eitan Grinspun. 2012. Interactive Editing of Deformable Simulations. *ACM Trans. Graph.* 31, 4 (2012), 70:1–70:8.

H. Barreiro, S. Sinclair, and M. A. Otaduy. 2019. Ultrasound Rendering of Tactile Interaction with Fluids. In *2019 IEEE World Haptics Conference (WHC)*. 521–526.

Sliman Bensmaia and Louise Manfredi. 2012. The Sense of Touch. In *The Encyclopedia of Human Behavior*, VS Ramachandran (Ed.). Elsevier, Amsterdam.

A. Bicchi, E. P. Scilingo, E. Ricciardi, and P. Pietrini. 2008. Tactile flow explains haptic counterparts of common visual illusions. *Brain Research Bulletin* 75, 6 (2008), 737–741.

Desai Chen, David I. W. Levin, Wojciech Matusik, and Danny M. Kaufman. 2017. Dynamics-aware Numerical Coarsening for Fabrication Design. *ACM Trans. Graph.* 36, 4 (2017), 84:1–84:15.

F. Chinello, M. Malvezzi, C. Pacchierotti, and D. Prattichizzo. 2015. Design and development of a 3RRS wearable fingertip cutaneous device. In *Advanced Intelligent Mechatronics (AIM), 2015 IEEE International Conference on*. 293–298.

S. Choi and K. J. Kuchenbecker. 2013. Vibrotactile Display: Perception, Technology, and Applications. *Proc. IEEE* 101, 9 (2013), 2093–2104.

J. E. Colgate and J. M. Brown. 1994. Factors affecting the Z-Width of a haptic display. In *Proceedings of the 1994 IEEE International Conference on Robotics and Automation.*

J. E. Colgate, M. C. Stanley, and J. M. Brown. 1995. Issues in the haptic display of tool use. *Proc. of IEEE/RSJ International Conference on Intelligent Robots and Systems* (1995), pp. 140–145.

Christian Duriez, Frederic Dubois, Abderrahmane Kheddar, and Claude Andriot. 2006. Realistic Haptic Rendering of Interacting Deformable Objects in Virtual Environments. *Proc. of IEEE TVCG* 12, 1 (2006).

A Frisoli, M Solazzi, F Salsedo, and M Bergamasco. 2008. A Fingertip Haptic Display for Improving Curvature Discrimination. *Presence* 17, 6 (Dec 2008), 550–561. https://doi.org/10.1162/pres.17.6.550

Lawson Fulton, Vismay Modi, David Duvenaud, David I. W. Levin, and Alec Jacobson. 2019. Latent-space Dynamics for Reduced Deformable Simulation. *Computer Graphics Forum* (2019).

Alberto Garcia-Garcia, Brayan Stiven Zapata-Impata, Sergio Orts-Escolano, Pablo Gil, and Jose Garcia-Rodriguez. 2019. Tactilegcn: A graph convolutional network for predicting grasp stability with tactile sensors. *arXiv preprint arXiv:1901.06181* (2019).

Carlos Garre, Fernando Hernandez, Antonio Gracia, and Miguel A. Otaduy. 2011. Interactive Simulation of a Deformable Hand for Haptic Rendering. In *Proc. of World Haptics Conference.*

G. J. Gerling, I. I. Rivest, D. R. Lesniak, J. R. Scanlon, and L. Wan. 2014. Validating a Population Model of Tactile Mechanotransduction of Slowly Adapting Type I Afferents at Levels of Skin Mechanics, Single-Unit Response and Psychophysics. *IEEE Transactions on Haptics* 7, 2 (2014), 216–228.

B.T. Gleeson, S.K. Horschel, and W.R. Provancher. 2010. Design of a Fingertip-Mounted Tactile Display with Tangential Skin Displacement Feedback. *Haptics, IEEE Transactions on* 3, 4 (2010), 297–301.

P. Grigg and A. H. Hoffmann. 1982. Properties of Ruffini afferents revealed by stress analysis of isolated sections of cat knee capsule. *Journal of Neurophysiology* 47, 1 (1982), 41–54.

Klaus Hildebrandt, Christian Schulz, Christoph von Tycowicz, and Konrad Polthier. 2012. Interactive Spacetime Control of Deformable Objects. *ACM Trans. Graph.* 31, 4 (2012), 71:1–71:8.

S. Inoue, Y. Makino, and H. Shinoda. 2015. Active touch perception produced by airborne ultrasonic haptic hologram. In *World Haptics Conference (WHC), 2015 IEEE.* 362–367.

Sophie Jörg, Jessica Hodgins, and Alla Safonova. 2012. Data-driven Finger Motion Synthesis for Gesturing Characters. *ACM Trans. Graph.* 31, 6 (2012), 189:1–189:7.

Roberta L. Klatzky and Susan J. Lederman. 2003. *Touch.* American Cancer Society, Chapter 6, 147–176.

D. Leithinger, S. Follmer, A. Olwal, and H. Ishii. 2015. Shape Displays: Spatial Interaction with Dynamic Physical Form. *Computer Graphics and Applications, IEEE* 35, 5 (2015), 5–11.

Daniele Leonardis, Massimiliano Solazzi, Ilaria Bortone, and Antonio Frisoli. 2015. A wearable fingertip haptic device with 3 DoF asymmetric 3-RSR kinematics. In *World Haptics Conference (WHC), 2015 IEEE.* 388–393.

Benjamin Long, Sue Ann Seah, Tom Carter, and Sriram Subramanian. 2014. Rendering Volumetric Haptic Shapes in Mid-air Using Ultrasound. *ACM Trans. Graph.* 33, 6, Article 181 (2014), 181:1–181:10 pages.

R. Luo, T. Shao, H. Wang, W. Xu, X. Chen, K. Zhou, and Y. Yang. 2018. NNWarp: Neural Network-based Nonlinear Deformation. *IEEE Transactions on Visualization and Computer Graphics* (2018).

Mickaël Ly, Romain Casati, Florence Bertails-Descoubes, Mélina Skouras, and Laurence Boissieux. 2018. Inverse Elastic Shell Design with Contact and Friction. *ACM Trans. Graph.* 37, 6 (2018), 201:1–201:16.

William A. McNeely, Kevin D. Puterbaugh, and James J. Troy. 1999. Six Degrees-of-Freedom Haptic Rendering Using Voxel Sampling. In *Proc. of SIGGRAPH 99 (Computer Graphics Proc.).* 401–408.

K. Minamizawa, H. Kajimoto, N. Kawakami, and S. Tachi. 2007. A Wearable Haptic Display to Present the Gravity Sensation - Preliminary Observations and Device Design. In *EuroHaptics Conference, 2007 and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems. World Haptics 2007. Second Joint.* 133–138. https://doi.org/10.1109/WHC.2007.15

Alessando Moscatelli, Meike Scheller, Gabriele Joanna Kowalski, and Marc Ernst. 2014. The Haptic Analog of the Visual Aubert-Fleischl Phenomenon. In *Haptics: Neuroscience, Devices, Modeling, and Applications*, Malika Auvray and Christian Duriez (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 34–40.

Maria Nolano, Vincenzo Provitera, Claudio Crisci, Annamaria Stancanelli, Gwen Wendelschafer-Crabb, William Robert Kennedy, and Lucio Santoro. 2003. Quantification of myelinated endings and mechanoreceptors in human digital skin. *Annals of Neurology* 54, 2 (2003), 197–205.

M.A. Otaduy, C. Garre, and M.C. Lin. 2013. Representations and Algorithms for Force-Feedback Display. *Proc. IEEE* 101, 9 (2013), 2068–2080.

Miguel A. Otaduy, Allison Okamura, and Sriram Subramanian. 2016. Haptic Technologies for Direct Touch in Virtual Reality. In *ACM SIGGRAPH 2016 Courses.* ACM, New York, NY, USA, 13:1–13:123.

C. Pacchierotti, D. Prattichizzo, and K. J. Kuchenbecker. 2016. Cutaneous Feedback of Fingertip Deformation and Vibration for Palpation in Robotic Surgery. *IEEE Transactions on Biomedical Engineering* 63, 2 (2016), 278–287.

Michel Paré, Allan M. Smith, and Frank L. Rice. 2002. Distribution and terminal arborizations of cutaneous mechanoreceptors in the glabrous finger pads of the monkey. *Journal of Comparative Neurology* 445, 4 (2002), 347–359.

A. G. Perez, D. Lobo, F. Chinello, G. Cirio, M. Malvezzi, J. S. Martín, D. Prattichizzo, and M. A. Otaduy. 2017. Optimization-Based Wearable Tactile Rendering. *IEEE Transactions on Haptics* 10, 2 (2017), 254–264.

D. Prattichizzo, F. Chinello, C. Pacchierotti, and M. Malvezzi. 2013. Towards wearability in fingertip haptics: a 3-DoF wearable device for cutaneous force feedback. *IEEE Transactions on Haptics* 6, 4 (2013), 506–516.

William R. Provancher, Mark R. Cutkosky, Katherine J. Kuchenbecker, and Günter Niemeyer. 2005. Contact Location Display for Haptic Perception of Curvature and Object Motion. *International Journal of Robotics Research* 24, 9 (2005), 691–702.

Javier Romero, Dimitrios Tzionas, and Michael J. Black. 2017. Embodied Hands: Modeling and Capturing Hands and Bodies Together. *ACM Trans. Graph.* 36, 6 (2017), 245:1–245:17.

Samuel B. Schorr and Allison M. Okamura. 2017. Fingertip Tactile Devices for Virtual Object Manipulation and Exploration. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17).* Association for Computing Machinery, New York, NY, USA, 3115–3119.

Rajinder Sodhi, Ivan Poupyrev, Matthew Glisson, and Ali Israr. 2013. AIREAL: Interactive Tactile Experiences in Free Air. *ACM Trans. Graph.* 32, 4, Article 134 (2013), 134:1–134:10 pages.

M. Solazzi, W.R. Provancher, A. Frisoli, and M. Bergamasco. 2011. Design of a SMA actuated 2-DoF tactile device for displaying tangential skin displacement. In *IEEE World Haptics Conference.*

A.A. Stanley and A.M. Okamura. 2015. Controllable Surface Haptics via Particle Jamming and Pneumatics. *Haptics, IEEE Transactions on* 8, 1 (2015), 20–30.

A. A. Stanley and A. M. Okamura. 2017. Deformable Model-Based Methods for Shape Control of a Haptic Jamming Surface. *IEEE Transactions on Visualization and Computer Graphics* 23, 2 (2017), 1029–1041.

Syntouch. 2018. BioTac by Synthuch. https://www.syntouchinc.com/en/sensor-technology/

M. Verschoor, D. Lobo, and M. A. Otaduy. 2018. Soft Hand Simulation for Smooth and Robust Natural Interaction. In *2018 IEEE Conference on Virtual Reality and 3D User Interfaces (VR).* 183–190.

Bin Wang, Paul Kry, Yuanmin Deng, Uri Ascher, Hui Huang, and Baoquan Chen. 2019. Learning Elastic Constitutive Material and Damping Models. arXiv:cs.GR/1909.01875

Dangxiao Wang, Xin Zhang, Yuru Zhang, and Jing Xiao. 2013. Configuration-Based Optimization for Six Degree-of-Freedom Haptic Rendering for Fine Manipulation. *Haptics, IEEE Transactions on* 6, 2 (2013), 167–180.

H. Xu and J. Barbič. 2017. 6-DoF Haptic Rendering Using Continuous Collision Detection between Points and Signed Distance Fields. *IEEE Transactions on Haptics* 10, 2 (2017), 151–161.

Hongyi Xu, Yijing Li, Yong Chen, and Jernej Barbič. 2015. Interactive Material Design Using Model Reduction. *ACM Trans. Graph.* 34, 2 (2015), 18:1–18:14.

Jeffrey M. Yau, Sung Soo Kim, Pramodsingh H. Thakur, and Sliman J. Bensmaia. 2016. Feeling form: the neural basis of haptic shape perception. *Journal of Neurophysiology* 115, 2 (2016), 631–642.

Yuting Ye and C. Karen Liu. 2012. Synthesis of Detailed Hand Manipulations Using Contact Sampling. *ACM Trans. Graph.* 31, 4 (2012), 41:1–41:10.

Vibol Yem, Ryuta Okazaki, and Hiroyuki Kajimoto. 2016. FinGAR: Combination of Electrical and Mechanical Stimulation for High-Fidelity Tactile Presentation. In *ACM SIGGRAPH 2016 Emerging Technologies (SIGGRAPH '16).* Association for Computing Machinery, New York, NY, USA.

D. Zhao, Y. Li, and J. Barbič. 2018. 6-DoF Haptic Rendering of Static Coulomb Friction Using Linear Programming. *IEEE Transactions on Haptics* 11, 3 (2018), 325–337.

Wenping Zhao, Jianjie Zhang, Jianyuan Min, and Jinxiang Chai. 2013. Robust Realtime Physics-based Motion Control for Human Grasping. *ACM Trans. Graph.* 32, 6 (2013), 207:1–207:12.