Elastic Tactile Simulation Towards Tactile-Visual Perception

Yikai Wang[1], Wenbing Huang[1,2], Bin Fang[1], Fuchun Sun[1] ✉, Chang Li[3]
[1]Beijing National Research Center for Information Science and Technology (BNRist),
State Key Lab on Intelligent Technology and Systems,
Department of Computer Science and Technology, Tsinghua University; [2]Pazhou Laboratory; [3]JD Explore Academy
wangyk17@mails.tsinghua.edu.cn, hwenbing@126.com,
{fangbin, fcsun}@mail.tsinghua.edu.cn, lichang93@jd.com

ABSTRACT

Tactile sensing plays an important role in robotic perception and manipulation tasks. To overcome the real-world limitations of data collection, simulating tactile response in a virtual environment comes as a desirable direction of robotic research. In this paper, we propose Elastic Interaction of Particles (EIP) for tactile simulation, which is capable of reflecting the elastic property of the tactile sensor as well as characterizing the fine-grained physical interaction during contact. Specifically, EIP models the tactile sensor as a group of coordinated particles, and the elastic property is applied to regulate the deformation of particles during contact. With the tactile simulation by EIP, we further propose a tactile-visual perception network that enables information fusion between tactile data and visual images. The perception network is based on a global-to-local fusion mechanism where multi-scale tactile features are aggregated to the corresponding local region of the visual modality with the guidance of tactile positions and directions. The fusion method exhibits superiority regarding the 3D geometric reconstruction task. Our code for EIP is available at https://github.com/yikaiw/EIP.

CCS CONCEPTS

• **Computing methodologies** → Computer graphics; Physical simulation; Modeling and simulation.

KEYWORDS

Tactile Simulation; Tactile-Visual Perception; Robotics

ACM Reference Format:
Yikai Wang[1], Wenbing Huang[1,2], Bin Fang[1], Fuchun Sun[1] ✉, Chang Li[3]. 2021. Elastic Tactile Simulation Towards Tactile-Visual Perception. In *Proceedings of the 29th ACM International Conference on Multimedia (MM '21), October 20–24, 2021, Virtual Event, China.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3474085.3475414

1 INTRODUCTION

Tactile sensing is one of the most compelling perception pathways for nowadays robotic manipulation, as it is able to capture the physical patterns including shape, texture, and physical dynamics that are not easy to perceive via other modalities, *e.g.* vision. In recent years, data-driven machine learning approaches have exploited tactile data and exhibited success in a variety of robotic tasks, such as object recognition [12, 17], grasp stability detection [16, 34], and manipulation [4, 26] to name some. That being said, the learning-based methods, particularly those involving deep learning, usually require large datasets for training. Collecting a large real tactile dataset is not easy, since it demands continuous robot control which is time-consuming or even risky considering the hardware wear and tear. Another concern with the real tactile collection is that the data acquired by the sensors of different shapes/materials or under different control policies could be heterogeneously distributed, posing a challenge to fairly assess the effectiveness of different learning methods trained on different tactile datasets.

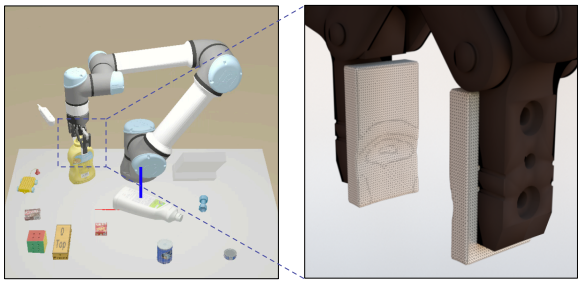

Figure 1: Illustration of the simulated tactile pattern when a robot is grasping a mustard bottle. The framework enables researchers to deal with tactile-visual perception in the simulated environment.

The simulation of tactile sensing can potentially help overcome these real-world limitations. Yet, establishing a promising tactile simulator is challenging since the tactile sensor needs to be geometrically and physically modeled. In addition, we should be capable of characterizing the physical interaction during the contact process between the sensor and the object which makes simulating tactile data more difficult than other modalities to some extent, such as vision that is solely geometrically aware.

Existing trails that consider simulating tactile interactions with manipulated objects [2, 14, 18, 21] usually model the tactile sensor as a combination of rigid bodies, and the collision between two objects is described by rigid multi-body kinematics provided by certain off-the-shelf physical engines (such as ODE in [14]). Despite its validity in some cases, considering the tactile sensor as a rigid multi-body will overlook the fact that common tactile sensors are usually elastic

MM '21, October 20–24, 2021, Virtual Event, China

ACM ISBN 978-1-4503-8651-7/21/10…$15.00
https://doi.org/10.1145/3474085.3475414

✉ Corresponding author: Fuchun Sun.

but not rigid. For example, the sensors invented by [33] leverages elastic materials to record the deformation to output tactile sensing. Moreover, in current methods, the segmentation of tactile sensor into rigid bodies is usually coarse and the interaction between rigid bodies is hard to capture the high-resolution sensor-object contact.

In this paper, we propose a novel methodology for tactile simulation, dubbed as Elastic Interaction of Particles (EIP). EIP first models the tactile sensor as a group of coordinated particles of certain mass and size. By assuming the sensor to be made of elastic materials, the elastic property is applied to constraint the movement of particles [23]. During the interaction between the sensor and the object, the deformation of particles is recorded as tactile data. An example of the simulated tactile perception is illustrated in Figure 1.

With fine-grained tactile patterns available, in this paper, we further propose a tactile-visual perception method that densely fuses features of both modalities, which exhibits great advantages on the 3D geometric reconstruction task. Multimodal learning can exhibit remarkable benefits against the unimodal paradigm if the patterns of different modalities are aligned and aggregated desirably [20, 27, 31]. However, tactile signals and visual images are not naturally aligned, since these two modalities are collected separately in different viewpoints. The discrepancy of viewpoints leads to geometrically unaligned tactile-visual feature maps. To this end, we design a global-to-local fusion network to integrate the learned tactile features into the visual counterparts. Since our tactile simulation is able to provide the position and direction for each touch, each feature map of the tactile modality is pooled into a global embedding and is then located to a local visual region for feature aggregation. The tactile-visual fusion is conducted densely in the architecture to capture multi-scale resolutions. Finally, inspired by Pixel2Mesh [28], the aggregated features of both modalities are further sent to a GNN-based network for vertices deformation.

To sum up, our contributions are two-fold:

- We propose EIP, a novel tactile simulating framework that is capable of modeling the elastic property of the tactile sensor and the fine-grained physical interaction between the sensor and the object. In contrast to existing methods that usually exploit the off-the-shelf physics engine for interaction simulation, the implementation of our method is formulated from scratch, which makes our framework more self-contained and easier to be plugged into downstream robotic applications.
- We propose a global-to-local fusion method to densely aggregates features of tactile and visual modalities. The designed per-pixel fusion method and the feature aggregation process alleviate the misalignment issue of tactile-visual features. Experimental results on 3D geometric reconstruction support the effectiveness of the proposed scheme. We combine EIP with a robotic grasping environment to acquire real-time tactile signals of the manipulated objects, which verifies its potential to downstream robotic tasks.

2 RELATED WORK

Tactile simulaton. The vision-based tactile sensors have become prominent due to their superior performance on robotic perception and manipulation. Data-driven approaches to tactile sensing are commonly used to overcome the complexity of accurately modeling contact with soft materials. However, their widespread adoption is impaired by concerns about data efficiency and the capability to generalize when applied to various tasks. Hence simulation approaches of vision-based tactile sensing are developed recently. Regarding the exploration of tactile simulation, early work [35] that directly adopts the elastic theory for the mesh interaction resorts to high computation costs. [18] represents the tactile sensor as a rigid body and calculates the interaction force on each triangle mesh. [7] models the tactile sensor as rigid elements and simulates their displacement by adding a virtual spring, with help of the commonly used Gazebo simulator. Modeling the tactile sensor as one or a combination of independent rigid bodies makes these methods difficult to obtain high-resolution tactile patterns, and these methods also overlook the fact that tactile sensors are mostly elastic materials. [2] implements the soft body simulation based on the Unity physics engine and trains a neural network to predict the contact information including positions and angles. [6] introduces an approach for simulating a GelSight tactile sensor in the Gazebo simulator by directly modeling the contact surface, which yet neglects the elastic material of the tactile sensor. Based on the finite-element analysis, [21] provides a simulation strategy to generate an entire supervised learning dataset for a vision-based tactile sensor, intending to estimate the full contact force distribution from real-world tactile images.

Tactile-visual perception. This paper mainly discusses the application of tactile-visual perception for the 3D reconstruction task. Several previous works combine vision and touch for shape reconstruction which rely on the given point cloud and depth data [1, 5, 13, 32]. Under such circumstances, shape reconstruction is less challenging since the real depth data or sparse point cloud at hand directly provides the global 3D information. When global depth is not available, [29] proposes to first estimate the depth and normal direction based on the vision and then predicts 3D structure with shape priors. Real-world tactile signals subsequently refine the predicted structure. Instead of predicting the global depth from the visual perception, [22] focuses on generating the local depth and point cloud from tactile signals, which provides local guidance for the chart deformation when a mass of tactile signals are obtained. [3, 25] discuss the tactile-visual perception based on real robotic hands from both functional and mechanistic perspectives. Different from existing methods, our tactile-visual perception framework is self-contained and end-to-end trainable without leveraging the global/local depth or point cloud information. We focus on how to combine geometrically unaligned feature maps for improving fusion performance.

3 PHYSICALLY TACTILE SIMULATION

In this section, we first introduce how to simulate the physical interaction between the tactile sensor and the targeted object. We then evaluate the effectiveness of our simulated tactile sensor by 3D geometric reconstruction with tactile-visual fusion.

3.1 Overall Framework

The basic idea of our method is to assume both the tactile sensor and the manipulated object to be solid and model them in the form of

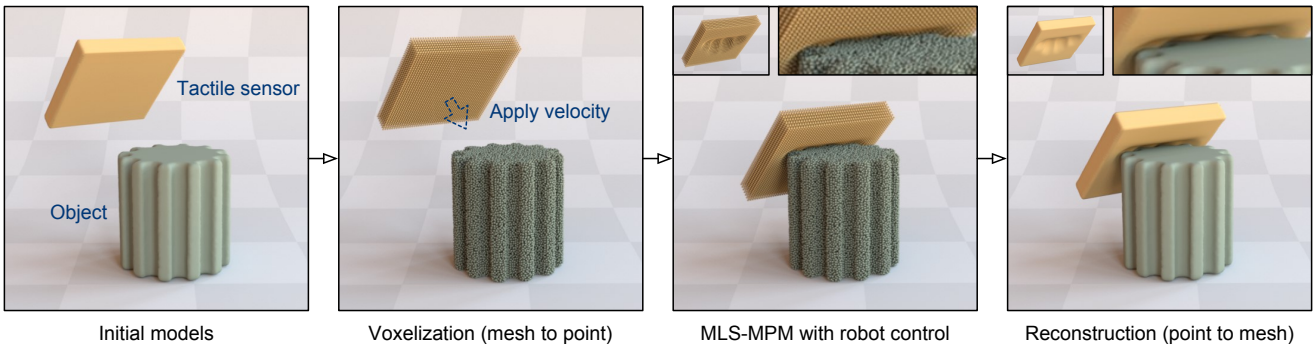

Figure 2: A brief framework of our tactile simulation process. The initial 3D meshes are first converted to particles by voxelization. Particles of the tactile sensor and the manipulated object are interacted by MLS-MPM, with additional control at the robot side.

particles. Differently, the sensor and the object are considered elastic and rigid respectively. In general, the tactile simulation process depicted in Figure 2 consists of 3 steps: voxelization from meshes to particles, interaction simulation, and reconstruction from particles to meshes. We detail each step below.

Voxelization. We first obtain the triangle meshes from the simulation environment that describe the geometric model of the object/sensor. The inside of each model is filled with dense voxel grids by voxel carving. Briefly, we first calculate the depth maps and then employ these depth maps to carve a dense voxel grid. We refer readers to [36] for more details. The center of each voxel grid is denoted as a particle, which is the fundamental unit for the following physical interaction simulation.

Interaction Simulation. We apply a certain velocity to the tactile sensor until it touches the object to a certain extent. The interaction process can be represented by the deformation of particles in the tactile sensor. Thus we simulate the deformation process based on Material-Point-Method (MPM) [24] and its modification MLS-MPM [10] considering both efficacy and efficiency, where we simultaneously apply the specific movement of the tactile sensor under robot control. Details of this step are provided in § 3.2.

Reconstruction. The final step is to reconstruct the meshes based on the positions of particles, which can be accomplished by using the method proposed by [15]. Note that this step is not necessary unless we want to render the interaction at each time step.

3.2 Interaction Simulation

This subsection presents details of how we simulate the interaction process. In practice, the tactile sensor is basically made of elastic materials, and our main focus is on the change of its shape, or called deformation. We apply the elastic theory to constrain the deformation of the particles in the tactile sensor during its interaction with the manipulated object, and the deformation at each time step will be recorded as the tactile sensing data.

Suppose that the sensor is composed of m particles. The coordinate of the p-th particle is represented by $\boldsymbol{x}_p \in \mathbb{R}^d$, where $d = 3$ throughout our paper. We define the deformation map as $\boldsymbol{\Phi} : \mathbb{R}^d \rightarrow \mathbb{R}^d$. The Jacobian of $\boldsymbol{\Phi}$ with respect to the p-th particle, denoted as $\boldsymbol{F}_p \in \mathbb{R}^{d\times d}$ (*a.k.a* deformation gradient), is obtained by

$$\boldsymbol{F}_p = \frac{\partial \boldsymbol{\Phi}}{\partial \boldsymbol{x}}(\boldsymbol{x}_p). \tag{1}$$

When the particle deforms, its volume may also change. The volume ratio by the deformation, denoted as J_p, is the determinant of $\boldsymbol{F}_p$,

$$J_p = \det(\boldsymbol{F}_p). \tag{2}$$

To describe the stress-strain relationship for elastic materials, we adopt a strain energy density function $\boldsymbol{\Psi}$, a kind of potential function that constrains the deformation $\boldsymbol{F}_p$. We follow a widely used method called Fixed Corotated [23], which computes $\boldsymbol{\Psi}$ by

$$\boldsymbol{\Psi}(\boldsymbol{F}_p) = \mu \sum_{i=1}^{d} (\sigma_{i,p} - 1)^2 + \frac{\lambda}{2}(J_p - 1)^2, \tag{3}$$

where $\mu = \frac{E}{2(1+\upsilon)}$ and $\lambda = \frac{E\upsilon}{(1+\upsilon)(1-2\upsilon)}$ are Lamé's 1st and 2nd parameters, respectively; E and υ are Young's modulus and Poisson ratio of the elastic material, respectively; $\sigma_{i,p}$ is the i-th singular value of $\boldsymbol{F}_p$. The derivative of $\boldsymbol{\Psi}$ (*a.k.a* the first Piola-Kirchhoff stress) will be utilized to adjust the deformation process, derived by

$$\boldsymbol{P}_p = \frac{\partial \boldsymbol{\Psi}}{\partial \boldsymbol{F}}(\boldsymbol{F}_p) = 2\mu(\boldsymbol{F}_p - \boldsymbol{R}_p) + \lambda(J_p - 1)J_p\boldsymbol{F}_p^{-\mathsf{T}}, \tag{4}$$

where $\boldsymbol{R}_p$ is obtained via the polar decomposition [9]: $\boldsymbol{F}_p = \boldsymbol{R}_p\boldsymbol{S}_p$.

In the following context, we will characterize how each particle deforms, that is, how its position $\boldsymbol{x}_p$ changes during the interaction phase. For better readability, we distinguish the position $\boldsymbol{x}_p$ and the quantities in Eq. (1-4) at each different time step by adding a temporal superscript, *e.g.* denoting the velocity at time step n as $\boldsymbol{x}_p^{(n)}$. We leverage MLS-MPM [10] to update $\boldsymbol{x}_p^{(n)}$, which divides the whole space into grids of a certain size. For each particle, its velocity is updated as the accumulated velocity of all particles within the same grid, which to some extent can emulate the physical interaction between particles. Specifically, we iterate the following steps for the update of $\boldsymbol{x}_p^{(n)}$. The flowchart is sketched in Algorithm 1.

Momentum Scattering. For each grid, we collect the mass and the momentum from the particles inside and those within its neighbors. The mass of the i-th grid is collected by

$$m_i' = \sum_{j\in\mathbb{G}_i} \sum_{p\in\mathbb{P}_j} w_{jp} m_p, \tag{5}$$

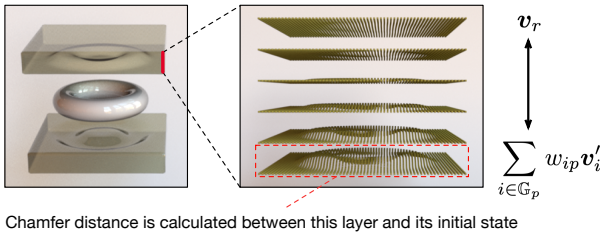

Figure 3: Deformation of pressing a torus mesh. To describe the transition from the velocity of the robot hand to the particle interacted velocity, we depict the deformation of each layer. The layer that directly contacts with the manipulated object has the largest extent of deformation, and we also use this layer to calculate the chamfer distance for terminal checking. Note that we increase the distance of tactile sensors for better visualization.

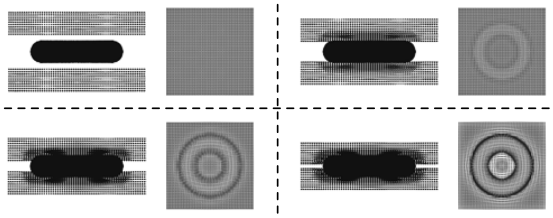

Figure 4: The change of tactile deformation during pressing a torus mesh, depicted using raw data.

where $\mathbb{G}_i$ denotes the $3\times3\times3$ neighbor grids surrounding grid i; here, we only consider the effects of particles in $\mathbb{G}_i$ and omit other distant particles due to the computational efficiency; $\mathbb{P}_j$ collects the indices of the particles located in grid j; m_p denotes the mass of the p-th particle; w_{jp} computes the B-Spline kernel negatively related to the distance between the j-th grid and the p-th particle.

The momentum of the i-th grid is derived as

$$(m'\boldsymbol{v}')_i = \sum_{j\in\mathbb{G}_i}\sum_{p\in\mathbb{P}_j} w_{jp}\left(m_p\boldsymbol{v}_p^{(n)} + \boldsymbol{C}_p^{(n)}(\boldsymbol{x}'_j - \boldsymbol{x}_p^{(n)})\right) - \sum_{j\in\mathbb{G}_i}\sum_{p\in\mathbb{P}_j} w_{jp}\gamma\boldsymbol{P}_p^{(n)}(\boldsymbol{F}_p^{(n)})^{\mathsf{T}}(\boldsymbol{x}'_j - \boldsymbol{x}_p^{(n)}), \tag{6}$$

where $\boldsymbol{x}'_j$ denotes the position of grid j; $v_p^{(n)}$ is the velocity of particle p; $\boldsymbol{C}_p^{(n)} \in \mathbb{R}^{d\times d}$, adopted as an approximated parameter in [10], is associated to the particle p whose update will be specified later; $\gamma = \frac{4\Delta t}{\Delta x'^2}V_p^0$ is a fixed coefficient where Δt is the time interval; $\Delta x'$ is the spatial interval between grids; V_p^0 is the initial particle volume.

Velocity Alignment. The velocity on the i-th grid can be obtained given the grid momentum and the grid mass by normalization, *i.e.*,

$$\boldsymbol{v}'_i = \frac{(m'\boldsymbol{v}')_i}{m'_i}. \tag{7}$$

Note that the grid velocity is only for later parameter updates, and the position of the grid will not change in the simulation.

Robot Hand Movement. Apart from the displacement caused by physical interaction, the sensor will also change its position due to the movement by the robot hand where the sensor is equipped. This additional velocity is a local parameter depending on the position of a particle, and we use a weight α_p to reflect it during the scattering process while keeping its sufficient physical interaction at the contact side. The velocity is thus updated as

$$\boldsymbol{v}_p^{(n+1)} = \alpha_p \sum_{i\in\mathbb{G}_p} w_{ip}\boldsymbol{v}'_i + (1-\alpha_p)\boldsymbol{v}_r, \tag{8}$$

where $\mathbb{G}_p$ is the set of $3\times3\times3$ grids that the particle p adheres to; $\boldsymbol{v}_r$ is the velocity of the robot hand; α_p is proportional to the distance between the particle and the robot hand as illustrated in Figure 3, and the deformation process is provided in Figure 4.

Parameters Gathering. With the updated velocity, we renew the values of the velocity gradient, position vector, and deformation gradient by

$$\boldsymbol{C}_p^{(n+1)} = \frac{4}{\Delta x^2}\sum_{i\in\mathbb{G}_p} w_{ip}\boldsymbol{v}_p^{(n+1)}(\boldsymbol{x}_i - \boldsymbol{x}_p^{(n)}), \tag{9}$$

$$\boldsymbol{x}_p^{(n+1)} = \boldsymbol{x}_p^{(n)} + \Delta t\boldsymbol{v}_p^{(n+1)}, \tag{10}$$

$$\boldsymbol{F}_p^{(n+1)} = (\boldsymbol{I} + \Delta t\boldsymbol{C}_p^{(n+1)})\boldsymbol{F}_p^{(n)}. \tag{11}$$

Terminal Checking. For safety and keeping consistent with the practical usage, we will terminate the robot hand movement once the deformation of the sensor is out of a certain scope. For this purpose, we use the chamfer distance to measure the distance between the deformed state and the original state of the particles in the contact surface, also illustrated in Figure 3. In form, we compute

$$l = \sum_{p\in\mathbb{S}}\min_{q\in\mathbb{S}}\|\hat{\boldsymbol{x}}_p^{(n+1)} - \hat{\boldsymbol{x}}_q^{(0)}\|_2^2 + \sum_{q\in\mathbb{S}}\min_{p\in\mathbb{S}}\|\hat{\boldsymbol{x}}_p^{(n+1)} - \hat{\boldsymbol{x}}_q^{(0)}\|_2^2, \tag{12}$$

where $\mathbb{S}$ denotes the contact surface between the sensor and the object; $\hat{\boldsymbol{x}} = \boldsymbol{x} - \bar{\boldsymbol{x}}$ for removing the effect of translation brought by $\boldsymbol{v}_r$, and $\bar{\boldsymbol{x}}$ denotes the center point of $\boldsymbol{x}$.

Algorithm 1 Elastic Interaction of Particles (EIP)

Input: 3D meshes of the manipulated object and the tactile sensor; values of Lamé's parameters including E and υ; robot hand velocity $\boldsymbol{v}_r$.
Output: Tactile interaction between the object and the sensor.

```
Convert meshes to particles using voxelization.
Initialize the values of x_p^(0), F_p^(0), C_p^(0), and v_p^(0).
Dividing the whole space into grids.
while not terminal do
    for each grid i do
        Scatter the mass and momentum of grid i by Eq. (5-6).
        Update the grid velocity by Eq. (7).
    end for
    for each particle p do
        Gather the velocity v_p^(n) by Eq. (8).
        Update parameters x_p^(n), F_p^(n), and C_p^(n) by Eq. (9-11).
    end for
    Terminal check of the robot control by Eq. (12).
end while
```

3.3 Tactile-Visual Multimodal Fusion

CNN-based architectures have shown the superiority in homogeneous multimodal fusion, *e.g.*, RGB-depth fusion [30], where feature

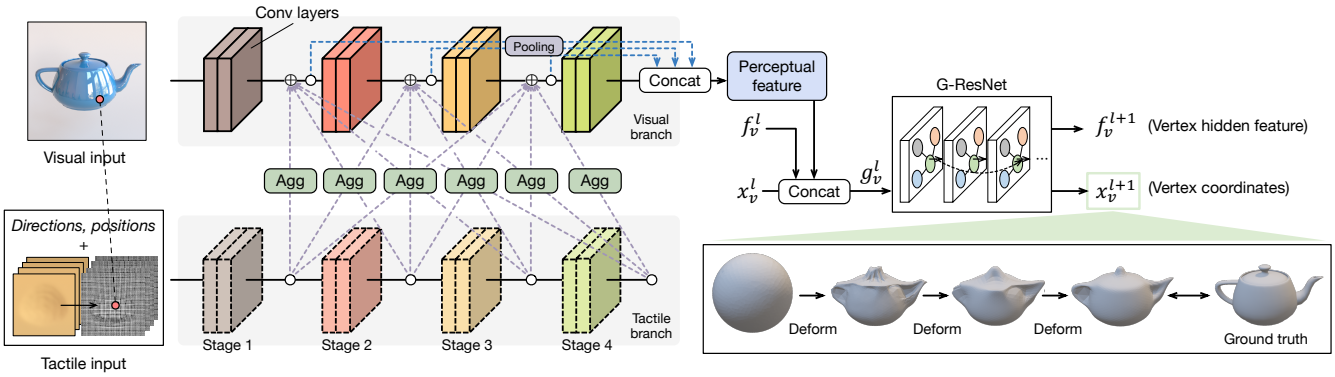

Figure 5: Training process of the proposed tactile-visual fusion method. Feature maps of the tactile branch are densely merged to the visual branch with Aggregation (Agg) blocks. Here, Agg transforms each tactile feature map into a 1×1 embedding, which is further added to 1×1 embeddings at the corresponding pixels in the visual feature map. More details are introduced in Figure 6. In the visual branch, multi-layer features are collected as the perceptual feature, which is then sent to the GNN (specifically, G-ResNet). Here, the average pooling technique is adopted for downsampling so that features from different stages can be concatenated. $\bigoplus$ indicates channel-wise addition of 1×1 embeddings. The learning process regarding GNN follows the method in Pixel2Mesh [28]. We provide an illustration of a teapot deforming from the initial sphere to the final prediction.

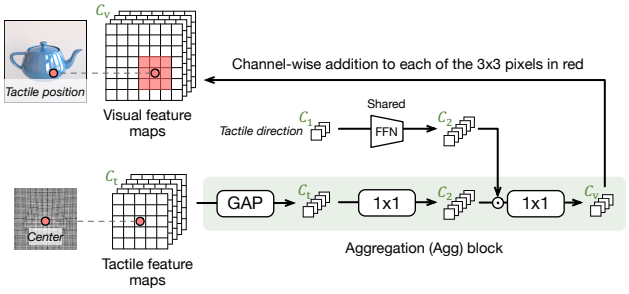

Figure 6: Illustration of the per-pixel fusion method and the feature transformation process with an Aggregation (Agg) block. GAP, 1×1 and FFN represent global average pooling, the 1×1 convolutional layer, and the feed-forward neural network, respectively. The channel number of each feature (map) is annotated in its upper left. $\odot$ indicates channel-wise dot production. Each aggregated tactile embedding is added to the 3×3 visual pixels which are colored in red, which is for amplifying the multimodal fusion effectiveness.

maps of different modalities are naturally aligned. But here, the feature maps of vision and touch are unaligned due to the discrepancy of viewpoint and tactile direction. Existing methods mostly leverage estimated global or local depth for multi-modal alignment. In this section, we propose an end-to-end tactile-visual perceptual framework that directly combines multi-modal features without explicitly predicting depth or sparse point cloud.

Specifically, we focus on the 3D geometric reconstruction of the manipulated objects, where we utilize the simulated tactile signals as complementary information to single-view visual images. During the prediction process, the network input is composed of an image and a set of tactile data obtained from different grasps based on different directions.

The structure of our tactile-visual perception framework is illustrated in Figure 5. We adopt a visual branch and a tactile branch to process visual and tactile inputs, respectively. We choose ResNet [8] as the backbone network for both branches, and thus there are four stages depicted. Multi-scale tactile features are extracted and densely connected to each network stage in the visual branch for multimodal feature fusion. We then concatenate multi-stage feature maps as a perceptual feature, which is finally sent to a Graph Neural Network (GNN) to predict the vertices deformation of the 3D mesh following the scheme in Pixel2Mesh [28].

The main contributions of our method include a designed Aggregation (Agg) block and the global-to-local feature fusion method to alleviate the misalignment issue. As illustrated in Figure 6, Agg transforms tactile features before integrating them into the visual branch. The transformation consists of a Global Average Pooling (GAP), which results in a global embedding, and two 1×1 Conv layers for feature extraction. Given that the touch direction plays an important role in tactile representation, we additionally involve the direction information into our tactile processing, by using a Feed-Forward Network (FFN) (that is shared in different Agg blocks) to produce the direction embedding which is integrated with the tactile global embedding via dot product. Note that in Agg, each 1×1 Conv layer is followed by a ReLU activation and a batch normalization layer. As described in Sec. 1, the tactile-visual feature maps are not naturally aligned due to the discrepancy of viewpoint and tactile direction. In our tactile generation, we are able to record the position (*i.e.* the coordinate center of tactile pixels). With the help of the tactile position, we then locate the corresponding local region (the 3×3 region colored in red in Figure 6) in each visual feature map and then perform addition between the global tactile embedding by Agg and every pixel within the local visual region. It is known that the different-stage layer in neural networks characterizes different-scale patterns. Hence, we conduct the above

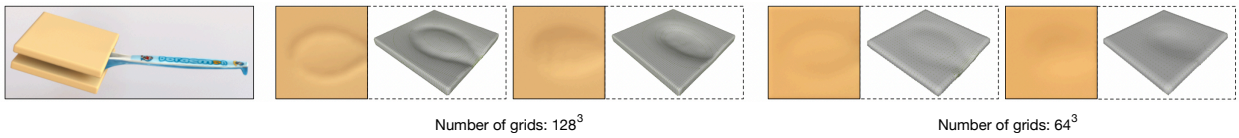

Figure 7: Comparison of tactile patterns when pressing a spoon, with different grid number settings.

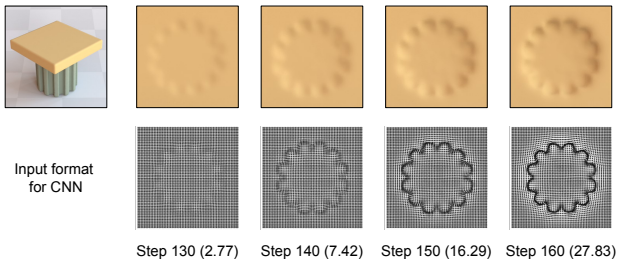

Figure 8: Comparison of tactile patterns at different time steps. The extent of deformation increases with the time step increases, followed by the increase of the chamfer distance l (10^{-5}) as shown in the brackets. We also depict the input format for CNN.

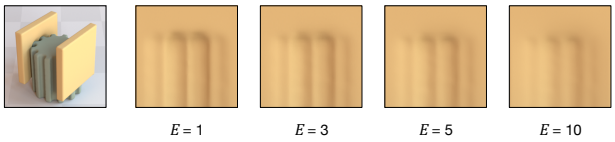

Figure 9: Comparison of tactile patterns with different elastic coefficients (Young's modulus E), at the same time step.

tactile-visual Agg between every two stages across the tactile and visual branches, to enable multi-scale and delicate fusion.

Inspired by [28], the perceptual feature is sent to a GNN to guide the deformation process of GNN vertices, which are also the vertices of the 3D geometric model to be reconstructed. The geometric model is initialized to an ellipsoid mesh and the deformation is realized by updating 3-dimensional vertices embeddings (coordinates) of GNN. Each vertex on the mesh would be projected to the closest visual feature [28]. The deformation update process is illustrated in the right part of Figure 5, and can be formulated as below,

$$\boldsymbol{g}_v^l = \text{concat}\left(\boldsymbol{f}_v^l, \boldsymbol{x}_v, \text{proj}(\boldsymbol{f}_p)\right), \tag{13}$$

$$\boldsymbol{f}_v^{l+1} = \boldsymbol{W}_{f0}\boldsymbol{g}_v^l + \sum_{v' \in \mathcal{N}(v)} \boldsymbol{W}_{f1}\boldsymbol{g}_{v'}^l, \tag{14}$$

$$\boldsymbol{x}_v^{l+1} = \boldsymbol{W}_{x0}\boldsymbol{g}_v^l + \sum_{v' \in \mathcal{N}(v)} \boldsymbol{W}_{x1}\boldsymbol{g}_{v'}^l, \tag{15}$$

where $\boldsymbol{f}_p$ denotes the perceptual feature learned by tactile-visual fusion, which has been introduced above (annotated in a blue box in Figure 5); v denotes the index of the vertex on the mesh; $\mathcal{N}(v)$ denotes the neighbors of v, which is available as the mesh is initially a sphere; $\boldsymbol{f}_v^l, \boldsymbol{x}_v^l$ are the feature representation and the learned coordinate of vertex v for the l-th GNN layer, respectively; $\boldsymbol{g}_v^l$ is the hidden feature given by the concatenation of $\boldsymbol{f}_v^l, \boldsymbol{x}_v$ and a multimodal feature projection on $\boldsymbol{f}_p$, denoted as $\text{proj}(\boldsymbol{f}_p)$; $\boldsymbol{W}_{f0}, \boldsymbol{W}_{f1}, \boldsymbol{W}_{x0}, \boldsymbol{W}_{x1}$ are learnable weights.

4 EXPERIMENT

We implement Algorithm 1 based on Taichi [11], and adopt Mitsuba [19] for rendering 3D models, *e.g.* in Figure 2 and Figure 7.

4.1 Effects of Coefficient Settings in EIP

In Figure 7, we provide the tactile patterns when pressing a spoon, and compare the patterns under different grid numbers (described in § 3.2). We observe that the larger the number of grids is the more fine-grained simulation we will attain. We set the grid number as 128^3 considering the trade-off between efficacy and efficiency. To illustrate how the deformation behaves during the contact, in Figure 8, we keep pressing the tactile sensor on a gear object and record results at different time steps. For each time step, we also provide its corresponding format for CNN input, and the Chamfer distance l as described in § 3.2; The deformation becomes more remarkable as the contact proceeds. Figure 9 contrasts the influence by Young's modulus E under the same press displacement. It is shown that the tactile range gets smaller with the increase of Young's modulus, which is consistent with the conclusion in elastic theory. In our simulation, we choose $E = 3$ and $v = 0.25$ in Eq. 3 by default.

4.2 Tactile Dataset

We build a tactile dataset containing 7,000 tactile images of 35 different object classes. These tactile images are collected through different contact policies including press directions and forces. The dataset summary and train/test splits are provided in Table 1. Figure 10 illustrates an example subgroup of the tactile dataset.

Once we obtain the tactile data (namely, the deformation of particles of the sensor), we can apply these data for object recognition which partially evaluates the quality of the tactile data. We suppose the tactile deformation of the contact surface as $\boldsymbol{X}^{(N)} \in \mathbb{R}^{H \times W \times d}$, where H and W denote the height and the width of the sensor, respectively, and N denotes the final time step. Then, we train a neural network $\hat{y} = f(\boldsymbol{X}^{(N)})$ to predict the object label. In practice, we prefer to try several attempts of the touch for more accurate recognition. All the deformation outcomes of different touching, denoted as $\{\boldsymbol{X}_i^{(N)}\}_{i=1}^I$ will be concatenated along the channel direction, leading to $\boldsymbol{Z} \in \mathbb{R}^{W \times H \times (Id)}$, as input of the network f.

To predict the object category given tactile patterns, we train a ResNet-18 [8] with ImageNet pretraining. We also conduct prediction experiments with more than one input tactile image. Specifically, during each training iteration, we randomly choose N tactile images which are yielded by different press directions to the same object. As introduced in § 3.3, we treat the number of images as the number of the input channel. Table 2 summarizes the accuracies of the tactile perception with the number of input images (touches) from 1 to 10. It reads that increasing the number of touches consistently improves the classification accuracy, and when the number

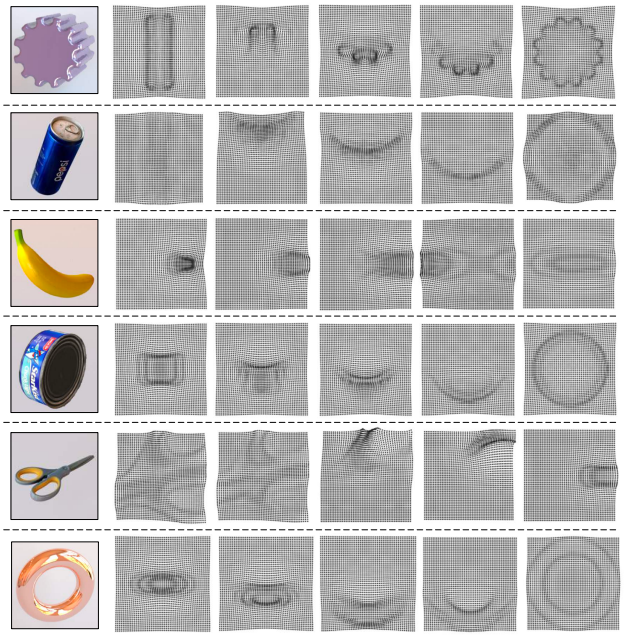

Figure 10: Sample visualization of our tactile dataset generated by EIP simulation.

Table 1: Summary of the dataset structure and dataset split for downstream tasks. We augment the dataset with random mesh deformation to improve the object diversity which also reduces object intersections in training and testing.

Task	Resolution	Training data			Testing data		
		Class	Tacile	Visual	Class	Tacile	Visual
Classification	224 × 224	35	5,500	0	35	1,500	0
3D reconstruction	224 × 224	35	6,500	650	10	500	50

Table 2: Classification accuracies (%) of the tactile perception. Results of each setting are collected over three runs.

1-touch	2-touch	4-touch	6-touch	8-touch	10-touch
36.7±2.3	47.4±1.6	59.7±1.1	81.5±0.7	90.2±0.6	92.9±0.3

Table 3: Results comparison of mesh reconstruction. Each result is the average over 10 different manipulated objects. Evaluation metric: chamfer distance ($\times 10^{-3}$), lower values indicate better performance.

Model	Visual	Tactile			Tactile & Visual		
		2-touch	5-touch	10-touch	2-touch	5-touch	10-touch
Chart-based [22]	13.65	-	25.06	17.44	-	8.10	5.83
Ours	14.07	28.70	21.91	15.65	10.27	7.22	5.32

is equal to 10, the accuracy becomes close to 93%, which implies the potential usage of our tactile simulation for real robotic perception.

4.3 Tactile-Visual Mesh Reconstruction

In this part, we assess the performance of 3D mesh prediction given a single-view image and a certain number of simulated tactile data

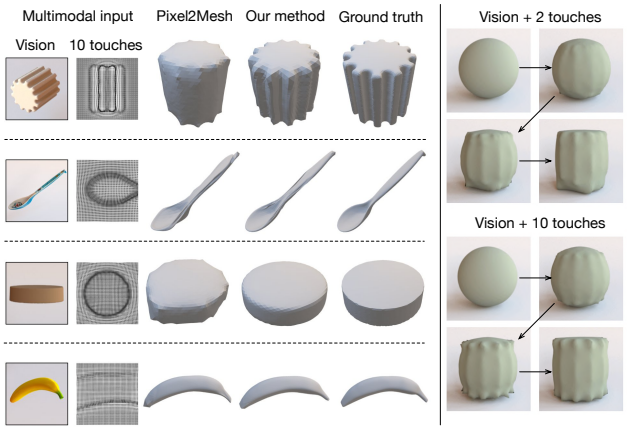

Figure 11: Left: Visualization of the mesh reconstruction. Our multimodal method achieves better performance than using the single-view image only, *i.e.* Pixel2Mesh. Note that we use 10 tactile patterns in total from different directions for reconstructing each mesh, and we depict one of them given the limited space. Right: Comparison of deformation processes of using 2 or 10 touches.

by EIP. The evaluation is accomplished on 3D mesh models of 10 classes. We collect visual images under random view and tactile data from different press directions. Per each training iteration, we randomly sample 1 visual image and 1∼10 tactile images as the network input. We perform random deformation on meshes so that there are no intersections of 3D meshes for training and testing.

We adopt ResNet-18 as our tactile-visual perception backbone and resize both visual and tactile images to the resolution 224×224. In addition, we adopt the cosine learning rate scheduler with an initial learning rate 2×10^{-5}, and we train 50 epochs in total.

The quantitative comparison is provided in Table 3, where we calculate the average Chamfer distance as the evaluation metric by sampling both 1,000 points from the predicted and target meshes. We vary the input number of tactile images from 2 to 10 in analogy to the perception task before. Even surprisingly, our method with only 2 touches is sufficient to gain a smaller reconstruction error than Pixel2Mesh (the Visual column in the table), and the error will become much smaller if we increase the number to 10. Table 3 also provides comparison results with a current SOTA the Chart-based method [22] which adopts estimated local depth and point cloud for tactile-visual fusion. We use the public codes by the authors and share their default setting of generating 5 touches at each time. Thus, the tactile input by [22] could only be the multiplier of 5 (5, 10, etc). According to the results in Table 3, our method surpasses the current the Chart-based method with a large margin, probably thanks to the more elaborate multimodal fusion in our method.

Table 4 provides detailed evaluation results for each of the 10 classes. We also perform ablation studies to verify the effectiveness of each component that we propose. The ablation variants include:

- W/o GAP: the global average pooling is removed from our model and the feature maps of tactile and visual modalities are fused pixel-by-pixel. This ablation study is to justify the validity of our global-to-local fusion by GAP on alleviating multimodal feature misalignment.

Table 4: Results comparison with SOTA method and ours on 10 manipulated objects. Ablation studies on our method are also performed, and their descriptions are provided in Sec 4.3. Evaluation metric: chamfer distance ($\times 10^{-3}$), lower values indicate better performance.

Model	Bottle	Bowl	Can	Conditioner	Cube	Fork	Gear	Pepsi	Spoon	Scissors	Average
Chart-based [22]	7.02	**6.50**	5.27	4.31	5.65	**4.79**	7.45	5.77	4.95	6.62	5.83
Ours	**6.23**	7.41	**4.62**	**3.57**	**4.71**	5.04	**7.13**	4.79	3.86	5.82	**5.32**
W/o GAP	8.75	11.24	8.08	6.30	6.29	6.75	10.74	6.16	9.95	6.47	8.07
W/o tactile direction	8.12	9.54	6.79	5.01	6.04	6.10	8.61	6.22	4.90	6.30	6.76
1×1 per-pixel fusion	7.55	8.18	6.34	4.97	6.18	4.92	7.51	5.77	**3.67**	**5.53**	6.06
5×5 per-pixel fusion	6.80	7.10	5.66	5.73	5.36	5.17	8.73	**4.65**	5.36	6.69	6.13
Single-stage fusion	6.75	8.06	5.38	4.20	5.29	4.91	8.10	5.75	4.74	6.20	5.94

- W/o tactile direction: the learned embedding of tactile direction (as illustrated in Figure 6) is no longer integrated with the Agg embedding.
- 1×1 per-pixel fusion: as illustrated in Figure 6, each aggregated tactile embedding is added to the 3×3 visual pixels. We change 3×3 to 1 single visual pixel to observe if attending the local region is necessary.
- 5×5 per-pixel fusion: conversely, we change 3×3 to 5×5 to see whether the performance is benefited from a larger visual receptive field.
- Single-stage fusion: we remove the dense multi-scale connections between the tactile and visual branches and only allow one connection between each corresponding stage.

Observed from Table 4, we have the following findings:

- GAP plays an essential role in our tactile-visual perception method, as removing GAP leads to noticeable performance drops regarding all objects especially for those elaborate ones (*e.g.* spoon). This is possibly because adopting GAP helps alleviate the misalignment issue of multimodal features as mentioned in § 3.3.
- The tactile direction is also crucial in replenishing the information in tactile embeddings, since the generation of tactile data depends on what touch direction we apply.
- For some elaborate objects, *e.g.* spoon and scissors, adopting 1×1 per-pixel fusion can achieve promising or even better performance than the default 3×3 setting; similarly, 5×5 yields the superiority accuracy regarding some uniform objects. Overall, the 3×3 setting comes as the best choice for general classes of the evaluated objects.
- Performing single-stage fusion still produces desired results, but it is inferior to the multi-scale version.

Figure 11 displays the generation process of the 3D mesh models. We compare our method with Pixel2Mesh [28] that only adopts the visual input. Qualitatively, by adding the tactile input, our approach obtains better predictions than Pixel2Mesh in Figure 11. The experimental results here well verify the power of our tactile simulation in capturing the fine-grained patterns of the touched object.

4.4 Robot Environment Integration

We integrate our tactile simulation with the robot environment, to perform the pick-and-place task for several different objects. We first fuse RGB and depth information to get the corresponding semantic segmentation based on the multimodal fusion method in [30]. With the segmentation at hand, we detect the 3D position of the can and then pick it up and finally put it down at a different place. The whole process is depicted in Figure 12, below which we plot the corresponding tactile simulation for each phase. We observe that our tactile simulation does encode the cylinder shape of the can. Besides, the last column shows that the simulated tactile sensor can return to its original state after the grasping process.

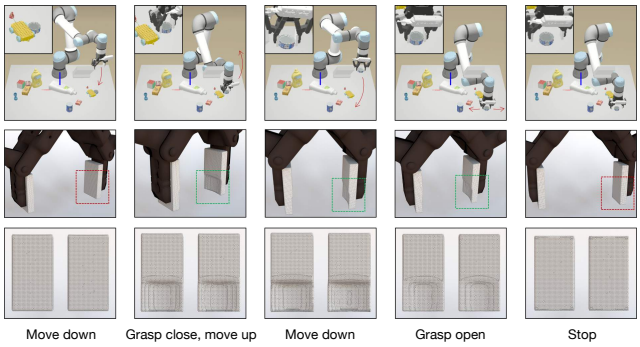

Figure 12: Illustration of a simulation scene where a robot picks up a fish can and puts it down. Red/green frames indicate without/with tactile deformation respectively. Zoom in for the best view.

5 CONCLUSION

In this work, we propose Elastic Interaction of Particles (EIP), a new method to simulate interactions between the tactile sensor and the object during robot manipulation. Different from existing tactile simulation methods, our design is based on the elastic interaction of particles, which allows much more accurate simulation with high resolution. Based on our tactile simulation, we further propose a global-to-local tactile-vision perception method for 3D geometric mesh reconstruction. Detailed experimental results verify the effectiveness of our tactile simulation and the proposed tactile-visual perception scheme.

ACKNOWLEDGEMENT

This research is jointly funded by Major Project of the New Generation of Artificial Intelligence, China (No. 2018AAA0102900), the Sino-German Collaborative Research Project Crossmodal Learning (NSFC 62061136001/DFG TRR169), and sponsored by CAAI-Huawei MindSpore Open Fund.

REFERENCES

[1] Björkman, M., Bekiroglu, Y., Hogman, V., Kragic, D.: Enhancing visual perception of shape through tactile glances. In: IROS (2013)
[2] Ding, Z., Lepora, N.F., Johns, E.: Sim-to-real transfer for optical tactile sensing. In: ICRA (2020)
[3] Edmonds, M., Gao, F., Liu, H., Xie, X., Qi, S., Rothrock, B., Zhu, Y., Wu, Y.N., Lu, H., Zhu, S.C.: A tale of two explanations: Enhancing human trust by explaining robot behavior. In: Science Robotics (2019)
[4] Fang, B., Sun, F., Yang, C., Xue, H., Chen, W., Zhang, C., Guo, D., Liu, H.: A dual-modal vision-based tactile sensor for robotic hand grasping. In: ICRA (2018)
[5] Gandler, G.Z., Ek, C.H., Björkman, M., Stolkin, R., Bekiroglu, Y.: Object shape estimation and modeling, based on sparse gaussian process implicit surfaces, combining visual data and tactile exploration. Robotics Auton. Syst. (2020)
[6] Gomes, D.F., Wilson, A., Luo, S.: Gelsight simulation for sim2real learning. In: ICRA ViTac Workshop (2019)
[7] Habib, A., Ranatunga, I., Shook, K., Popa, D.O.: Skinsim: A simulation environment for multimodal robot skin. In: CASE (2014)
[8] He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR (2016)
[9] Higham, N.J.: Computing the polar decomposition—with applications. In: SIAM Journal on Scientific and Statistical Computing (1986)
[10] Hu, Y., Fang, Y., Ge, Z., Qu, Z., Zhu, Y., Pradhana, A., Jiang, C.: A moving least squares material point method with displacement discontinuity and two-way rigid body coupling (2018)
[11] Hu, Y., Li, T.M., Anderson, L., Ragan-Kelley, J., Durand, F.: Taichi: a language for high-performance computation on spatially sparse data structures. In: TOG (2019)
[12] Huang, W., Sun, F., Cao, L., Zhao, D., Liu, H., Harandi, M.: Sparse coding and dictionary learning with linear dynamical systems. In: CVPR (2016)
[13] Ilonen, J., Bohg, J., Kyrki, V.: Three-dimensional object reconstruction of symmetric objects by fusing visual and tactile sensing. Int. J. Robotics Res. (2014)
[14] Kappassov, Z., Corrales-Ramon, J.A., Perdereau, V.: Simulation of tactile sensing arrays for physical interaction tasks. In: AIM (2020)
[15] Kazhdan, M., Hoppe, H.: Screened poisson surface reconstruction. In: TOG (2013)
[16] Kwiatkowski, J., Cockburn, D., Duchaine, V.: Grasp stability assessment through the fusion of proprioception and tactile signals using convolutional neural networks. In: IROS (2017)
[17] Liu, H., Wu, Y., Sun, F., Guo, D.: Recent progress on tactile object recognition. In: International Journal of Advanced Robotic Systems (2017)
[18] Moisio, S., León, B., Korkealaakso, P., Morales, A.: Model of tactile sensors using soft contacts and its application in robot grasping simulation. In: Robotics and Autonomous Systems (2013)
[19] Nimier-David, M., Vicini, D., Zeltner, T., Jakob, W.: Mitsuba 2: A retargetable forward and inverse renderer. In: TOG (2019)
[20] Ramachandram, D., Taylor, G.W.: Deep multimodal learning: A survey on recent advances and trends. In: IEEE Signal Processing Magazine (2017)
[21] Sferrazza, C., Bi, T., D'Andrea, R.: Learning the sense of touch in simulation: a sim-to-real strategy for vision-based tactile sensing. In: arXiv preprint arXiv:2003.02640 (2020)
[22] Smith, E.J., Calandra, R., Romero, A., Gkioxari, G., Meger, D., Malik, J., Drozdzal, M.: 3d shape reconstruction from vision and touch. In: NeurIPS (2020)
[23] Stomakhin, A., Howes, R., Schroeder, C., Teran, J.M.: Energetically consistent invertible elasticity. In: ACM SIGGRAPH/Eurographics (2012)
[24] Stomakhin, A., Schroeder, C., Chai, L., Teran, J., Selle, A.: A material point method for snow simulation. In: TOG (2013)
[25] Sundaram, S., Kellnhofer, P., Li, Y., Zhu, J.Y., Torralba, A., Matusik, W.: Learning the signatures of the human grasp using a scalable tactile glove. In: Nature (2019)
[26] Tian, S., Ebert, F., Jayaraman, D., Mudigonda, M., Finn, C., Calandra, R., Levine, S.: Manipulation by feel: Touch-based control with deep predictive models. In: ICRA (2019)
[27] Valada, A., Mohan, R., Burgard, W.: Self-supervised model adaptation for multimodal semantic segmentation. In: IJCV (2020)
[28] Wang, N., Zhang, Y., Li, Z., Fu, Y., Liu, W., Jiang, Y.G.: Pixel2mesh: Generating 3d mesh models from single rgb images. In: ECCV (2018)
[29] Wang, S., Wu, J., Sun, X., Yuan, W., Freeman, W.T., Tenenbaum, J.B., Adelson, E.H.: 3d shape perception from monocular vision, touch, and shape priors (2018)
[30] Wang, Y., Huang, W., Sun, F., Xu, T., Rong, Y., Huang, J.: Deep multimodal fusion by channel exchanging. In: NeurIPS (2020)
[31] Wang, Y., Sun, F., Lu, M., Yao, A.: Learning deep multimodal feature representation with asymmetric multi-layer fusion. In: ACM MM (2020)
[32] Watkins-Valls, D., Varley, J., Allen, P.K.: Multi-modal geometric learning for grasping and manipulation. In: ICRA. pp. 7339–7345 (2019)
[33] Yuan, W., Dong, S., Adelson, E.H.: Gelsight: High-resolution robot tactile sensors for estimating geometry and force. In: Sensors (2017)
[34] Zapata-Impata, B.S., Gil, P., Torres, F.: Tactile-driven grasp stability and slip prediction. In: Robotics (2019)
[35] Zhang, H., Chen, N.N.: Control of contact via tactile sensing. In: IEEE Trans. Robotics Autom. (2000)
[36] Zhou, Q.Y., Park, J., Koltun, V.: Open3D: A modern library for 3D data processing. In: arXiv:1801.09847 (2018)