IEEE Transactions on Robotics (T-RO) paper, presented at ICRA 2026, Vienna, Austria. Cite as T-RO paper.

TacFlex: Multi-Mode Tactile Imprints Simulation for Visuotactile Sensors with Coating Patterns

Chaofan Zhang, Shaowei Cui, Jingyi Hu, Tianyu Jiang, Tiandong Zhang, Rui Wang, and Shuo Wang

***Abstract*—Visuotactile sensors have been shown to provide rich contact information for robots. However, how to build a high-fidelity visuotactile simulator that supports multi-mode tactile imprints and various sensor configurations (such as coating patterns) remains a challenging problem. In this paper, we present TacFlex, an efficient and flexible simulator for visuotactile sensors, which physically simulates the elastomer deformation using Finite Element Methods (FEM), and focuses on linking the deformed elastomer mesh to diverse tactile imprints, including tactile images with arbitrary coating patterns and tactile 3D point clouds. We further propose a ray tracing-based rectification method to deal with multi-medium refraction effects to make the simulated tactile images more realistic. Extensive qualitative and quantitative experiments are conducted to demonstrate the effectiveness of TacFlex on several visuotactile sensors. Furthermore, we explore the Sim2Real performance of different tactile imprints provided by TacFlex in tactile perception and manipulation tasks, such as cylindrical object pose estimation and peg-in-hole. The perception/policy models trained in simulation are successfully deployed in the real world. Finally, we present the outlook on the potential of TacFlex in visuotactile manipulation learning. The TacFlex simulator is open-sourced to the community. See supplementary video, code, and results at https://sites.google.com/view/tacflex/.**

***Index Terms*—Visuotactile sensors, visuotactile simulation, ray tracing, Sim2Real, robot learning**

I. Introduction

TACTILE sensing can provide contact information, such as force, geometry, and object position, and it is significant to improve the ability of robots in contact-rich manipulation tasks [1]. Visuotactile sensors, which employ built-in cameras to capture the deformation of the soft sensor surface, are well-known for high spatial resolution and rich tactile information [2] [3], and have been applied to cable manipulation [4], peg-in-hole [5], and tool manipulation [6] tasks. Nevertheless, most of the existing robotic manipulation skills are acquired through real-world interactions, presenting researchers with challenges in data collection and policy training that are both time-consuming and damaging to robots and sensors [7] [8]. The natural idea is to use the simulation environment to improve the robot learning efficiency [9]. However, visuotactile simulation is still a challenging problem in most robot simulators.

Nowadays, non-physics-based and physics-based methods are employed to simulate the elastomer deformation under sensor-object interactions. The non-physics-based method al-

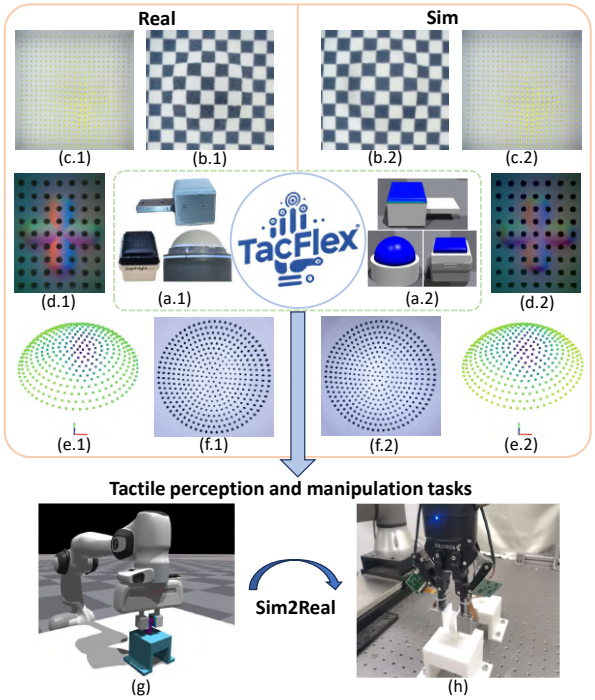

Fig. 1. This paper proposes a visuotactile simulation framework named TacFlex, supporting multi-mode tactile imprints and various visuotactile sensors. We explore the Sim2Real performance of different tactile imprints provided by TacFlex in tactile perception and manipulation tasks, in which the perception/policy models learned in simulation could be successfully deployed in the real world. (a) Visuotactile sensors. (b) Tactile image with checkerboard pattern. (c) Tactile image with marker pattern. The yellow arrows indicate marker motion fields drawn on the image. (d) Tactile image with marker pattern and multi-color lighting. (e) Tactile 3D point clouds. (f) Tactile image of a sensor with a curved surface. (g) A peg-in-hole task in simulation. (h) A peg-in-hole task in the real world.

This work was supported in part by the National Key Research and Development Program of China under Grant 2023YFB4705000, in part by the National Natural Science Foundation of China under 62303455, 62373039, U23B2038, U24A20128, 62273342, and 62122087, in part by Beijing Natural Science Foundation under Grant L233006, in part by the Youth Program and the Open Projects Program of State Key Laboratory of Multimodal Artificial Intelligence Systems. *(Corresponding author: Shaowei Cui)*

Chaofan Zhang and Tianyu Jiang are with the Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China, and also with the School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing 100049, China (e-mail: zhangchaofan2020@ia.ac.cn; jiangtianyu2023@ia.ac.cn).

Shaowei Cui, Tiandong Zhang, and Rui Wang are with the State Key Laboratory of Multimodal Artificial Intelligence Systems, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China (e-mail: shaowei.cui@ia.ac.cn; tiandong.zhang@ia.ac.cn; rwang5212@ia.ac.cn).

Jingyi Hu is with the Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China, and also with the School of Future Technology, University of Chinese Academy of Sciences, Beijing 100049, China (e-mail: hujingyi2019@ia.ac.cn).

Shuo Wang is with the State Key Laboratory of Multimodal Artificial Intelligence Systems, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China, with the School of Artificial Intelligence, University of Chinese Academy of Sciences, Beijing 100049, China, and also with the Center for Excellence in Brain Science and Intelligence Technology Chinese Academy of Sciences, Shanghai 200031, China (e-mail: shuo.wang@ia.ac.cn).

IEEE Transactions on Robotics (T-RO) paper, presented at ICRA 2026, Vienna, Austria. Cite as T-RO paper.

lows penetration between the object and the sensor surface to approximate the deformation of the sensor elastomer [10] [11] [12], which fails to simulate the contact dynamics, resulting in a severe sim-real gap. Instead, Material Point Methods (MPM) and Finite Element Methods (FEM) are physics-based approaches. The MPM encounters difficulties in simulating frictional force [13], while the FEM shows advantages in accurately simulating the elastic and contact dynamics of sensor elastomer under various contact states [14] [15]. As accurately simulating the sensor deformation is essential for contact-rich manipulation tasks, the FEM is a worthwhile choice in visuotactile simulation.

In addition to deformation simulation, several attempts are also desired to simulate diverse visuotactile imprints. The current studies focus on RGB optical rendering of tactile images for GelSight-type sensors [12] [16] [17]. These simulators cannot support coating patterns on the sensor surface, which makes the simulation not available to pattern-based sensors (e.g., GelSight sensor with markers [18], DenseTact 2.0 sensor [19]). To date, the high-fidelity visuotactile simulation supporting various sensor configurations and different tactile imprints has not been fully investigated.

In this paper, we propose a flexible visuotactile simulation framework named TacFlex, as shown in Fig. 1, which integrates *multi-mode tactile imprints*, including tactile images with arbitrary coating patterns, marker motion fields, tactile 3D point clouds, etc. The TacFlex could support *multi-type sensors*, such as GelSight [20], GelStereo [21] [22], FingerVision [23], and so on. Specifically, the Finite Element Method (FEM) is employed to physically simulate the elastomer deformation during sensor-object interactions and generate the deformed mesh. Subsequently, a mesh-marker mapping method and a texture mapping method are presented to link the elastomer mesh to tactile markers embedded in the sensor elastomer and coating patterns on the sensor surface, respectively. As the elastomer deforms, the markers shift, and the pattern warps accordingly. In this way, we could simulate the tactile 3D point clouds and render the tactile images at each simulation step.

Moreover, we find that the existing render methods of tactile images do not consider the multi-medium refraction effects, which brings significant discrepancies between simulation and reality. Some simulators describe these discrepancies as scaling errors and mitigate them by offsetting the camera position or scaling the image [13] [16] [24]. These simple methods hinder the fidelity of visuotactile simulation. Although the multi-medium refraction effect can be physically simulated in render engines supporting ray tracing, such as Blender and Unity, the rendering process is computationally expensive and time-consuming. In summary, how to efficiently simulate highly realistic tactile images remains to be explored. In this paper, we propose *a ray tracing-based rectification model* that can be generated within a few milliseconds, enabling fast and high-fidelity simulation of tactile images. To our knowledge, this is the first time that the multi-medium refraction effect has been dealt with in visuotactile simulation.

Furthermore, we explore the Sim2Real applications of TacFlex in tactile perception and manipulation tasks, including cylindrical object pose estimation and peg-in-hole task. The perception and policy models are trained using simulated tactile images and tactile 3D point clouds generated by TacFlex, respectively, and successfully achieve zero-shot transfer to the real world. Moreover, the peg-in-hole policy trained in a 2.0 mm clearance scene in simulation can be directly applied to different real-world clearances, such as a 0.6 mm clearance. These results demonstrate the effectiveness of TacFlex and its significant potential in Sim2Real robot learning.

The contributions of this paper are summarized as follows:

- A visuotactile simulation framework named TacFlex is introduced, integrating multi-mode tactile imprints and supporting multi-type visuotactile sensors. To the best of our knowledge, it is the first framework supporting tactile images with arbitrary coating patterns.
- We propose a ray tracing-based rectification method to reduce the sim-real gap of tactile images caused by multi-medium refraction effects. This method is computationally efficient and significantly improves the accuracy of tactile image simulation.
- We explore the robot learning on two tactile perception and manipulation tasks using TacFlex, in which the Sim2Real performance demonstrates the effectiveness of TacFlex on multi-mode tactile imprints simulation.

The rest of this paper is organized as follows. Section II lists the related works on visuotactile simulation. Section III introduces the TacFlex pipeline. The simulation and real-world experiments on various sensors are shown in Section IV. Additionally, Section V demonstrates the Sim2Real applications of TacFlex in tactile perception and manipulation tasks. Finally, we discuss and summarize this article in Section VI and VII.

II. Related Works

Visuotactile simulation has been emerging for diverse sensors, aiming to reduce the burden of data acquisition and promote tactile research. According to the visuotactile sensing principle, the simulation can be divided into two phases, including deformation simulation of sensor elastomer and mapping the deformation to tactile imprints. The deformation simulation is the fundamental step of tactile imprints simulation. In this section, we introduce the related works on these two phases, as listed in Table I.

A. Deformation Simulation

The elastomer deformation simulation can be divided into non-physics-based and physics-based methods. For non-physics-based methods, the works in [10] [16] [25] utilize a virtual depth camera to capture the height map of the sensor surface and penetrated objects, approximating the deformation. The works in [12] [17] [24] directly generate the depth map from the 3D model of objects. Smoothing methods such as Gaussian filters are employed to approximate the smooth contact edges. These non-physics methods generate elastomer deformation directly from object shape and pose rather than elastic dynamics. They cannot simulate the stretching of the sensor surface, especially in tangential directions, which is essential for slip detection and contact-rich manipulation.

IEEE Transactions on Robotics (T-RO) paper, presented at ICRA 2026, Vienna, Austria. Cite as T-RO paper.

TABLE I
COMPARISON OF THE PROPOSED TACFLEX WITH THE STATE-OF-THE-ART VISUOTACTILE SIMULATORS.

Simulators	Deformation Simulation	Tactile Imprints Simulation				
		Depth Map	3D Point Clouds	Optical Rendering	Coating Pattern	Refraction Effects
Tactile Gym [10]	Non-physics	Depth camera	/	/	/	/
Gomes *et al.* [16]	Non-physics	Depth camera + smoothing	/	Phong's model	/	Scale transformation
TACTO [12]	Non-physics	Depth camera	/	RGB rendering in Pyrender	/	/
Agarwal *et al.* [17]	Non-physics	Object shape + smoothing	/	Physics-based rendering	/	/
Taxim [24]	Non-physics	Object shape + smoothing	/	Polynomial table mapping	/	Manually aligned
Kim *et al.* [25]	Non-physics	Depth camera	/	GAN	GAN, only support marker pattern	/
Tacchi [13]	MPM	2D interpolation on surface particles	Represented by particles	Phong's model	/	Scale transformation
Luu *et al.* [14]	FEM	Deformed skin surface	/	Virtual camera in Gazebo	Model markers as thin cylinders	/
DIFFTACTILE [26]	FEM	Deformed surface mesh + smoothing	Interpolate on surface vertices	MLP mapping	/	/
Chen *et al.* [15]	FEM (IPC)	Deformed surface mesh	Triangular barycentric mapping	/	/	/
TacFlex (ours)	FEM	Deformed surface mesh	Tetrahedron barycentric mapping	RGB rendering in Pyrender	Supporting arbitrary patterns	Ray tracing-based rectification

Chen *et al.* [13] employ the Material Point Method (MPM) for physical deformation simulation, which applies particles and an imaginary grid to manage the contact between soft elastomer and objects. However, the MPM encounters difficulties in simulating frictional force, resulting in inaccurate simulation during sliding contact. Moreover, Si *et al.* [26] introduce a differentiable simulator named DIFFTACTILE, which models the sensors' elastomer by Finite Element Method (FEM) and simulates the contact by a penalty-based model. Chen *et al.* [15] and Du *et al.* [27] adopt Incremental Potential Contact (IPC) to enable robust FEM computing and simulate the sensor deformation under various contact states. Currently, the FEM offers substantial advantages in accurately simulating the elastic and contact dynamics of visuotactile sensors, and it can be integrated into mesh-based robotic simulators. Additionally, some robotic simulators, such as NVIDIA Isaac Sim/Gym, support GPU acceleration and parallel computing for FEM, thereby ensuring the time-efficiency of simulations. In this paper, we employ FEM to simulate the elastomer deformation and map it to multi-mode tactile imprints.

B. Tactile Imprints Simulation

The imprints of visuotactile sensors come in a wide variety of formats and mainly depend on the sensor design. Simulating diverse tactile imprints is challenging. Also, the imprint simulation is influenced by deformation simulation methods. The MPM is natural for tactile 3D point clouds simulation, in which each marker is represented by a particle sharing the same position [13]. The tactile 3D point clouds can also be simulated by interpolating between the surface vertices in [15] and [26], while it can be hardly effectively simulated based on non-physics-based deformation simulation.

Nowadays, the studies on tactile imprints simulation primarily focus on RGB optical rendering of tactile images from GelSight-type sensors. Gomes *et al.* [16] and Chen *et al.* [13] use Phong's shading model to render GelSight's internal illumination from the depth map of a sensor surface. Wang *et al.* [12] present a flexible and open-source simulator named TACTO, which directly adds objects, sensor surface, and light sources into the Pyrender simulation scene to simulate tactile images using OpenGL. Agarwal *et al.* [17] introduce a physics-based rendering technology, taking multiple bounces of the light and surface material model into account, to generate more accurate tactile images. These simulators can only support GelSight-type sensors with RGB lighting but cannot simulate coating patterns on the sensor surface. Nevertheless, a few studies simulate tactile images with markers. Luu *et al.* [14] model the markers as thin cylinders attached to the soft skin in SOFA and render the tactile image in Gazebo. This method is only applicable to simple patterns like markers. Kim *et al.* [25] employ a Generative Adversarial Network (GAN) framework to generate tactile images with markers from depth map directly. The network is trained using abundant aligned simulated depth images and real RGB images. The data collection is time-consuming, and the generalization is worrying. Therefore, how to simulate visuotactile sensors with arbitrary coating patterns needs to be explored in depth, which is significant for pattern-based sensors, such as DelTact [28] and GelStereo [29] [30].

III. METHODS

A. Pipeline

The TacFlex pipeline for visuotactile simulation is illustrated in Fig. 2. Deformation simulation is responsible for physically simulating the elastomer behaviors caused by

IEEE Transactions on Robotics (T-RO) paper, presented at ICRA 2026, Vienna, Austria. Cite as T-RO paper.

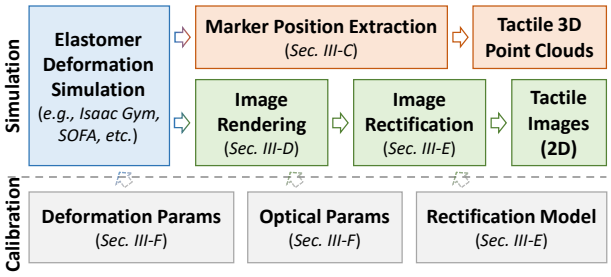

Fig. 2. The TacFlex pipeline for visuotactile simulation.

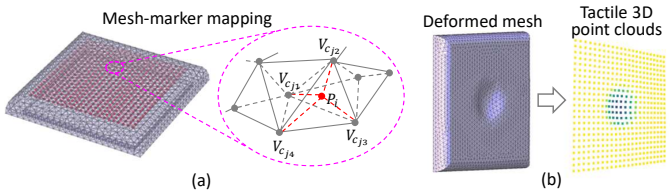

Fig. 3. Tactile 3D point clouds simulation. (a) The diagram of mesh-marker mapping in initialization phase. (b) Extracting tactile 3D point clouds from a deformed elastomer mesh in each simulation step.

sensor-object interactions and outputs the deformed mesh. We focus on linking the deformed elastomer mesh to different tactile imprints, including tactile 3D point clouds and tactile images. A marker position extraction method is created to generate tactile 3D point clouds from the elastomer mesh. We present an image rendering method to simulate tactile images with coating patterns, in which each surface vertex is mapped to a pixel coordinate on the pattern, and the pattern warps following the elastomer deformation. Subsequently, an image rectification method is employed to transform the previously rendered images into tactile images with multi-medium refraction effects. Furthermore, we calibrate the deformation and optical parameters in simulation with real visuotactile sensors.

B. Elastomer Deformation Simulation

We employ the FEM to simulate the elastomer deformation during sensor-object interactions, as FEM is commonly used in soft body simulation and is effective for elastic and contact dynamics. The visuotactile sensors are divided into soft parts (i.e., the elastomer) and rigid parts (i.e., the sensor frame). The elastomer is discretized into contiguous and nonoverlapping tetrahedral cells. The rigid part and objects are discretized into triangular cells on their surfaces and provide boundary conditions for FEM computing. The whole elastomer mesh with N_v vertices and N_c tetrahedrons can be indicated by $(\mathbf{V}, \mathbf{C})$, where $\mathbf{V} \in \mathbb{R}^{N_v \times 3}$ is the vertex positions in 3D space, and $\boldsymbol{C} = [[c_{j1}, c_{j2}, c_{j3}, c_{j4}], ...] \in \mathbb{R}^{N_c \times 4}$ is the tetrahedral cells denoted by vertex indexes.

The dynamic equation of the elastomer deformation under external forces could be formulaic as [31] [32]:

$$\mathbf{M}(\mathbf{u})\ddot{\mathbf{u}} + \mathbf{D}(\mathbf{u}, \dot{\mathbf{u}})\dot{\mathbf{u}} + \mathbf{F}_{elastic}(\mathbf{u}) + \mathbf{N}^T\lambda = \mathbf{F}_{ext} \quad (1)$$

where $\mathbf{u} = flatten(\mathbf{V} - \bar{\mathbf{V}}) \in \mathbb{R}^{3N_v}$ is the displacement vector of vertices under contact, in which $\bar{\mathbf{V}}$ is the vertex position without contact. Correspondingly, $\dot{\mathbf{u}}$ and $\ddot{\mathbf{u}}$ are the velocity vector and acceleration vector, respectively. $\mathbf{M}(\mathbf{u})$ is the mass matrix, representing the inertial properties of the soft elastomer. $\mathbf{D}(\mathbf{u}, \dot{\mathbf{u}})$ is the damping matrix, which represents the energy dissipation caused by factors such as internal friction, where Rayleigh damping model [33] is commonly employed. $\mathbf{F}_{ext}$ gathers known external forces applied on the elastomer. The λ indicates Lagrange multipliers, which introduce constraints into the dynamic system. $\mathbf{N}^T$ denotes the constraint conditions. The contact forces could be solved as constraints, in which $\mathbf{N}^T$ includes the contact constraints given by the contact detection, and λ is the desired contact force. Moreover, $\mathbf{F}_{elastic}(\mathbf{u})$ denotes the elastic force determined by vertex displacements and the properties of soft materials, which can be computed by elastic models, such as the co-rotational model [34], Neo-Hookean model, Mooney-Rivlin model, Ogden model, and so on [35].

This dynamic system can be solved using iterative methods, and updated at each simulation step. The elastomer deformation simulation could be performed in off-the-shelf FEM simulators, such as NVIDIA Isaac Gym and SOFA. We obtain the contact forces and deformed elastomer mesh in each simulation step for visuotactile simulation.

C. Tactile 3D Point Clouds Simulation

The visuotactile sensors usually have markers embedded on the sensor surface or in the elastomer. We model the mapping function between the markers and the elastomer's volumetric mesh to generate the tactile 3D point clouds, as shown in Fig. 3. Specifically, the tactile 3D point cloud perceived by the real sensor without contact is employed to initialize the simulation. We compute the barycentric coordinates of each marker with respect to the tetrahedron that contains it. Taking a marker $\mathbf{P}_i$ for example, the barycentric coordinates $(\gamma_{i1}, \gamma_{i2}, \gamma_{i3}, \gamma_{i4})$ are computed by

$$[\gamma_{i1}, \gamma_{i2}, \gamma_{i3}]^T = \mathbf{T}^{-1}(\bar{\mathbf{P}}_i - \bar{\mathbf{V}}_{c_{j4}}) \quad (2)$$

$$\mathbf{T} = [\bar{\mathbf{V}}_{c_{j1}}, \bar{\mathbf{V}}_{c_{j2}}, \bar{\mathbf{V}}_{c_{j3}}] - \bar{\mathbf{V}}_{c_{j4}} \quad (3)$$

$$\gamma_{i4} = 1 - \gamma_{i1} - \gamma_{i2} - \gamma_{i3} \quad (4)$$

where $\bar{\mathbf{P}}_i$ is the initial position of the ith marker, and the ith marker belongs to the jth tetrahedron. The vertices of the jth tetrahedron in the initial position are denoted by $\bar{\mathbf{V}}_{c_{j*}}$.

Then, given the vertices $\mathbf{V}_{c_{j*}}$ of the deformed mesh in each simulation step, the marker position $\mathbf{P}_i$ can be computed using

$$\mathbf{P}_i = \gamma_{i1}\mathbf{V}_{c_{j1}} + \gamma_{i2}\mathbf{V}_{c_{j2}} + \gamma_{i3}\mathbf{V}_{c_{j3}} + \gamma_{i4}\mathbf{V}_{c_{j4}}. \quad (5)$$

In this way, the 3D positions of all markers are generated in simulation, which we call tactile 3D point clouds.

D. Tactile Images Rendering

To simulate tactile images with coating patterns, we first map the pattern onto the sensor's surface mesh in the initialization phase, and then render the image using a virtual camera in each simulation step.

IEEE Transactions on Robotics (T-RO) paper, presented at ICRA 2026, Vienna, Austria. Cite as T-RO paper.

1) Texture Mapping: The tactile image from the real visuotactile sensor without contact is utilized as the texture pattern in simulation, as it is natural and accessible for any sensors. The texture mapping process can be defined as

$$\mathbf{p} = \mathbb{F}_m(\mathbf{V}_s) \tag{6}$$

where $\mathbf{V}_s \in \mathbb{R}^{N_s \times 3}$ denotes the vertices on the sensor surface that are extracted from the elastomer's volumetric mesh. $\mathbf{p} \in \mathbb{R}^{N_s \times 2}$ is the pixel coordinates on the texture pattern corresponding to $\mathbf{V}_s$. How to model the $\mathbb{F}_m(\cdot)$ function is the key point of texture mapping.

The straightforward idea for $\mathbb{F}_m(\cdot)$ function is to directly project the vertices onto the image plane using the pinhole camera model, as shown in Fig. 4 (a). However, we find that the texture on the sensor surface appears larger than that on the real sensor, as shown in Fig. 4 (c), which is intuitively explained in Fig. 4 (d). The multi-medium refraction effects in the visuotactile sensor's imaging system cannot be ignored for accurate texture mapping. To this end, we employ a refractive ray tracing model to back-propagate the rays from pixel coordinates into the 3D space. The texture mapping is then defined as an optimization problem.

Fig. 4 (b) shows the ray tracing process in n times of refraction with flat and parallel refracting interfaces. Specifically, given pixel coordinates (u, v) on the image, the direction of the ray $\vec{r}_0$ passing through the camera optical center O can be computed using camera intrinsic parameters. Then, the intersection points Q_j of the rays and the refracting interfaces are computed by

$$Q_j = \sum_{i=0}^{j-1} \frac{d_i}{\vec{r}_i \cdot \vec{n}} \vec{r}_i, j = 1, 2, \cdots, n \tag{7}$$

where $\vec{r}_i$ is the ray direction denoted by a unit vector in the ith medium; $\vec{n}$ is the unit normal vector of the refracting interface; d_i is the thickness of the ith medium. The refracted ray direction can be computed based on Snell's law.

$$\vec{r}_{i+1} = \alpha \vec{r}_i + (\sqrt{1 - \alpha^2[1 - (\vec{r}_i \cdot \vec{n})^2]} - \alpha \vec{r}_i \cdot \vec{n}) \cdot \vec{n} \tag{8}$$

where $\alpha = \mu_i / \mu_{i+1}$; μ_i is the refractive index of the ith medium. After that, we can map pixel coordinates (u, v) to a ray starting from point Q_n and pointing to $\vec{r}_n$ in the nth medium just under the sensor surface. The above ray tracing process can be represented as

$$Q_n, \vec{r}_n = \mathbb{F}_r(u, v; \phi) \tag{9}$$

where $\phi = \{\mu_*, d_*, \vec{n}, ...\}$ is the parameter set of the imaging system in the visuotactile sensor.

Subsequently, we create an optimization problem for each vertex in $\mathbf{V}_s$ to obtain the corresponding pixel coordinates (u, v) by minimizing the vertex-ray distance.

$$\min_{u,v} \ \mathbb{F}_{dist}(\mathbf{V}_{s_i}, Q_n, \vec{r}_n) \tag{10}$$

where $\mathbb{F}_{dist}$ is the distance function between a point and a line. In this way, we could realize accurate texture mapping. The sensor surface mesh after texture mapping is shown in the lower right corner of Fig. 4 (c). It seems much similar to the real sensors.

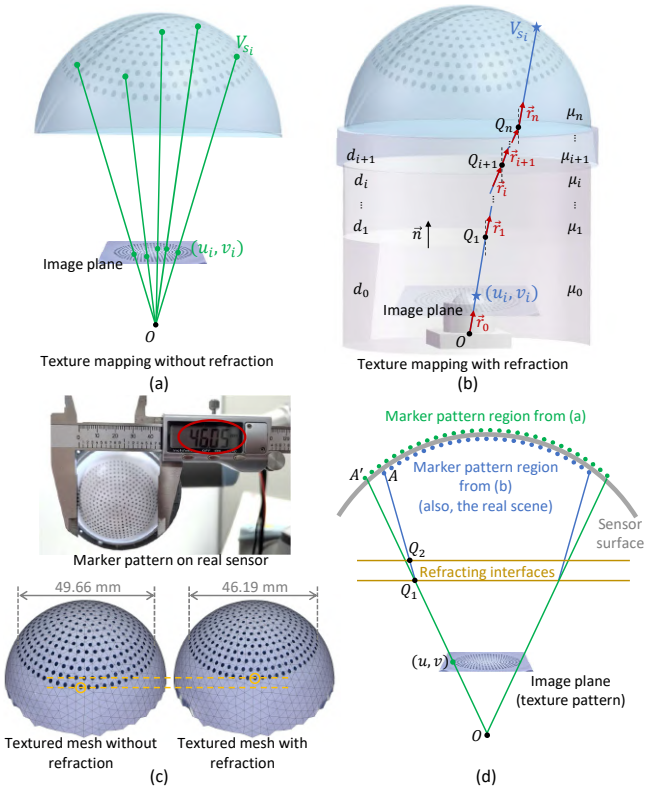

Fig. 4. The diagram of texture mapping which maps mesh vertices to pixels on a pattern image. (a) Texture mapping based on the pinhole camera model without refraction. (b) Texture mapping based on the refractive ray tracing model. The ray tracing process is illustrated in detail. (c) Comparison of the marker pattern on a real sensor and the textured surface meshes. (d) The marker pattern regions on the sensor surface obtained from texture mapping with and without refraction. The texture pattern becomes larger when refraction is ignored.

2) Image Rendering: Given the textured surface mesh, we render the tactile images using off-the-shelf render engines. In the initialization phase, we load the textured sensor surface mesh, cameras, and light sources of a visuotactile sensor into the simulation scene. The intrinsic and pose parameters of the camera are consistent with those in the real sensor; the color and position of light sources are also consistent with the real settings. The color of the light source can be flexibly configured. For example, we set white lights for GelStereo-type sensors and multiple color lights (e.g., red, green, blue, etc.) for GelSight-type sensors.

At each simulation step, we update the vertices of the textured surface mesh using the deformed elastomer mesh from the deformation simulation, as shown in the left part of Fig. 5 (a). A virtual camera is then employed to capture the RGB-D image of the textured sensor surface. Note that the virtual RGB-D images here are refractive-free.

In addition, for GelSight-type sensors, we deploy a post-processing method following TACTO [12], which uses a real tactile image to fine-tune simulated images to realize more realistic rendering. This procedure can be represented as:

$$I_{post}^{sim} = I_{w}^{sim} - I_{wo}^{sim} + I_{wo}^{real} \tag{11}$$

where I_{post}^{sim} is the simulated tactile image after post-processing. I_{w}^{sim} and I_{wo}^{sim} are the original simulated tactile

IEEE Transactions on Robotics (T-RO) paper, presented at ICRA 2026, Vienna, Austria. Cite as T-RO paper.

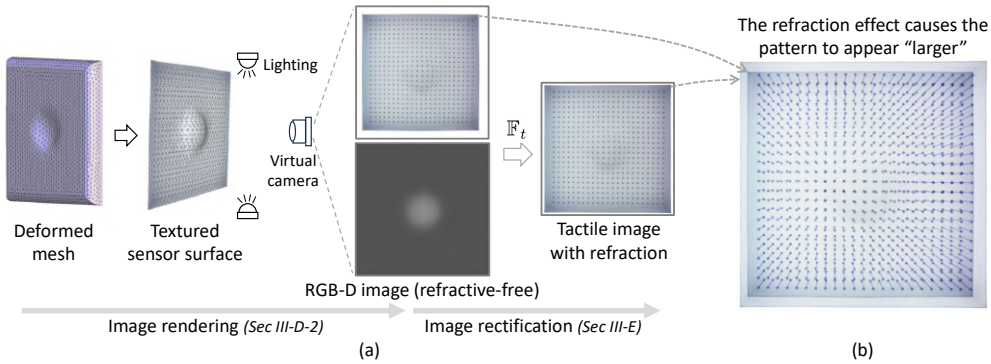

Fig. 5. The diagram of tactile images simulation. (a) TacFlex first renders the image and then applies image rectification at each simulation step. (b) We overlap the images before and after rectification and highlight their differences by arrows representing the coordinate changes of markers.

images with contact and without contact, respectively. I_{wo}^{real} is the real tactile image without contact.

E. Tactile Image Rectification

We propose a ray tracing-based rectification method to transform the refractive-free image into a tactile image with multi-medium refractive effects, as shown in the right part of Fig. 5 (a). To do this, we model the transformation function $\mathbb{F}_t$ as

$$(u, v) = \mathbb{F}_t((u', v')) \tag{12}$$

where $p' = (u', v')$ is an unrefracted image point, directly captured by a virtual camera; $p = (u, v)$ is the refracted image point after rectification.

We draw several pairs of corresponding points (p', p) and their ray paths, as shown in Fig. 6. It can be proved that the joint lines of these point pairs intersect at one point; this point is called Refractive Principal Point (RPP). Specifically, according to Snell's law, all ray segments (e.g., $O - Q_{i,1} - Q_{i,2} - A_i$), the camera optical center O, and the normal of refracting interface $\vec{n}$ lie in one common plane [36], denoted by refraction plane α_i. In imaging systems with flat and parallel refracting interfaces, the interface normal is consistent at any point. So, all refraction planes intersect at a straight line L that passes through the camera optical center and is parallel to the interface normal. Moreover, the unrefracted ray also lies in the refraction plane, e.g., $OA_i \subset \alpha_i$. So, the intersection line of the refraction plane α_i and the image plane β is the straight line $p_i'p_i$ determined by the refracted and unrefracted image point pair. As all refraction planes intersect at line L, all joint lines $p_*'p_*$ on the image plane intersect at the point p^r, which is the intersection point of the image plane and line L. $p^r = (u^r, v^r)$ is the Refractive Principal Point, which can be computed using camera intrinsic matrix $\mathbf{K} \in \mathbb{R}^{3\times3}$:

$$[\bar{u}, \bar{v}, \bar{w}]^T = \mathbf{K} \cdot \vec{n},\ p^r = (\bar{u}/\bar{w}, \bar{v}/\bar{w}). \tag{13}$$

Given a tactile image captured by the virtual camera, we can compute the distance from each pixel to the RPP. This distance is denoted using $r_i' = |p^r p_i'|$, as shown in Fig. 6 (b). We hope to model the refractive distortion $r_i^d = |p^r p_i| - |p^r p_i'|$ using r_i' to achieve the image transformation. To this end, we visualize some (r_i', r_i^d) samples on a plane, as shown in Fig. 7 (a). Obviously, the relationship between r_i^d and r_i' is related to h_i, where h_i is the depth of the points A_i in the camera coordinate system. Then, we apply a polynomial regression to approximate r_i^d as

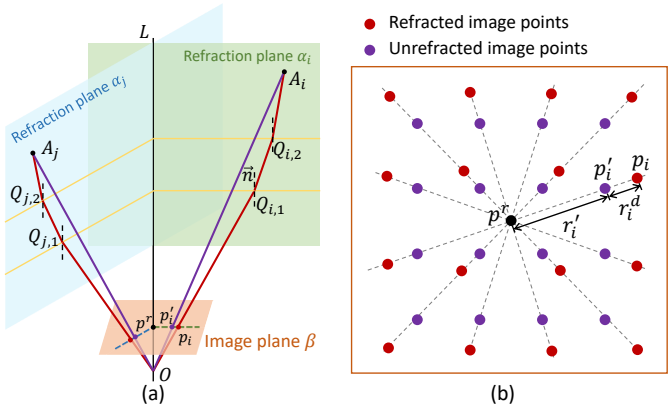

Fig. 6. (a) Diagram of the refracted and unrefracted ray paths. (b) The refracted and unrefracted point pairs on the image plane.

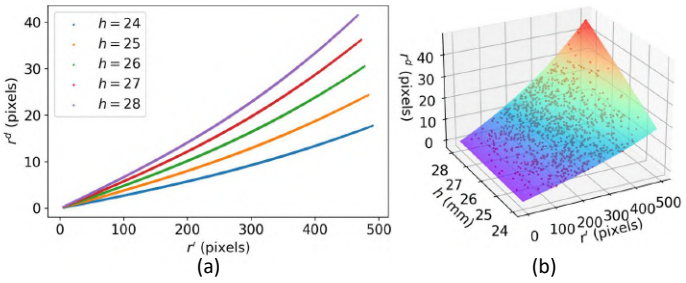

Fig. 7. (a) The curves of refractive distortion r^d with respect to distance r'. (b) The polynomial regression of r^d. The dots represent the samples, with a sample size of 1k. The surface illustrates the model after regression, achieving an R^2 score of 0.999.

$$r_i^d = \mathbb{F}_t(r_i', h_i) \tag{14}$$

where the depth h_i can be given by virtual depth camera in simulation; $\mathbb{F}_t$ is modeled by a cubic polynomial function.

Subsequently, an automated program is designed to generate a small-scale dataset containing (r_i', h_i, r_i^d) samples, which are utilized to fit the $\mathbb{F}_t$ function. Specifically, we first randomly sample some pixels on the image plane as $p_i = (u_i, v_i)$, and

IEEE Transactions on Robotics (T-RO) paper, presented at ICRA 2026, Vienna, Austria. Cite as T-RO paper.

calculate the corresponding ray $Q_{i,n}, \vec{r}_{i,n}$ using the refractive ray tracing model $\mathbb{F}_r$. Then, we sample a 3D point on this ray as A_i, and the depth h_i of A_i can be directly given by its z-axis coordinate. Finally, we project A_i onto the image plane using the pinhole camera model to obtain $p'_i = (u'_i, v'_i)$, which is the unrefracted image point. Given the camera and refraction system parameters of a visuotactile sensor, we can easily generate data and fit this model $\mathbb{F}_t$. The data generation and fitting process only takes several milliseconds. An example of $\mathbb{F}_t$ regression is shown in Fig. 7 (b). The R^2 score of the predicted refractive distortion achieves 0.999, indicating that this simple polynomial regression model represents the refractive distortion very well. For each pixel on the unrefracted image, the corresponding pixel on the refracted image can be computed by

$$(u, v) = (u', v') + \frac{(u', v') - (u^r, v^r)}{\|(u', v') - (u^r, v^r)\|} r^d. \tag{15}$$

The image differences before and after the rectification are illustrated in Fig. 5 (b).

F. Simulation Physical Parameters Calibration

We carefully determine the physical parameters in simulation to achieve highly realistic visuotactile simulation. Specifically, tactile signals from real sensors are collected under various contact conditions, which are induced by pressing and tangentially sliding objects on the sensor surface. These contact states are then consistently reproduced in the simulation using different simulation parameters. The calibration of these parameters is formulated as an optimization problem aimed at minimizing the discrepancies between simulated and real tactile signals.

In this paper, the deformation simulation and optical rendering parameters are calibrated separately. The material parameters, such as Young's modulus E, Poisson ratio ν, and friction coefficient μ, are calibrated for deformation simulation, in which we minimize the sim-real differences of the 3D geometry of the sensor surface and the contact force. For each contact sample, the difference of contact force f_{force} can be represented as

$$f_{force} = \|\mathbf{F}_{sim} - \mathbf{F}_{real}\|_2 \tag{16}$$

where $\mathbf{F}_{sim}$ is the simulated three-axis contact force on the sensor surface; $\mathbf{F}_{real}$ is the real contact force measured by a force sensor. In various visuotactile sensors, the contact 3D geometry is represented differently. The primary forms include tactile 3D point clouds (e.g., GelStereo-type sensor with markers) and depth maps (e.g., GelSight-type sensors). Correspondingly, the sim-real differences are computed by

$$\text{3D points: } f_{geom} = \frac{1}{N_p} \sum_{i=0}^{N_p-1} \|\mathbf{P}_i - \hat{\mathbf{P}}_i\|_2 \tag{17}$$

$$\text{Depth map: } f_{geom} = \frac{1}{N_d} \sum_{i=0}^{N_d-1} |b_i - \hat{b}_i| \tag{18}$$

where $\mathbf{P}$ and $\hat{\mathbf{P}}$ indicate the simulated and real tactile markers in 3D space, respectively. N_p is the number of markers. b and $\hat{b}$ denote the simulated and real depths of the sensor surface, respectively. N_d is the number of elements in the depth map. Further, the differences of 3D geometry and contact force are normalized to establish the objective function, which is defined as

$$\min_{\phi_d} \frac{1}{N_m} \sum_{i=0}^{N_m-1} (\bar{f}_{geom,i} + \bar{f}_{force,i}) \tag{19}$$

where ϕ_d is the deformation parameter set. N_m is the number of aligned sim-real samples. $\bar{f}_{geom,i}$ is the normalized $f_{geom,i}$; $\bar{f}_{force,i}$ is the normalized $f_{force,i}$. In practice, min-max normalization is applied, wherein the minimum and maximum values are determined by the sim-real differences within a predefined parameter range.

Subsequently, based on the accurate deformation simulation using the parameters from the above procedure, we render tactile images using different optical parameters and compute the similarity of simulated and real tactile images. The Mean Square Error (MSE) is employed to set up the objective function, as MSE is sensitive to luminance changes.

$$\min_{\phi_o} \frac{1}{N_m} \sum_{i=0}^{N_m-1} \mathbb{F}_{mse}(I_i^{sim}, I_i^{real}) \tag{20}$$

where ϕ_o denotes the optical parameter set, such as light intensities, beam angle, and diffuse reflectivity. I_i^{sim} and I_i^{real} indicate the simulated and real tactile images, respectively. $\mathbb{F}_{mse}$ is the MSE function.

To address these optimization problems, we begin by simulating tactile signals using a set of parameters derived from a coarse parameter grid. Subsequently, we assess the differences between the simulated and real signals within this parameter grid to progressively refine the parameter ranges. Finally, a heuristic optimization algorithm, such as differential evolution, is employed to finely optimize the parameter set. In this way, these parameters are calibrated from coarse to fine.

IV. Experiments

A. Design and Setup

The primary goal of our experiments is to verify the effectiveness of the proposed multi-mode tactile imprints simulation pipeline for visuotactile sensors. Particularly, we should answer the following questions:

- Why do we employ FEM for elastomer deformation simulation?
- How realistic are the simulated multi-mode tactile imprints (e.g., tactile 3D point clouds, tactile images, marker motion fields) according to real sensor outputs?
- How flexible is the proposed TacFlex? Could TacFlex support sensors with various coating patterns? Could it be applied to diverse visuotactile sensors and integrated into different FEM simulators?
- Could TacFlex apply to robot learning in tactile perception and manipulation tasks? What is the Sim2Real performance?

To answer these questions, we design a set of evaluation experiments.

IEEE Transactions on Robotics (T-RO) paper, presented at ICRA 2026, Vienna, Austria. Cite as T-RO paper.

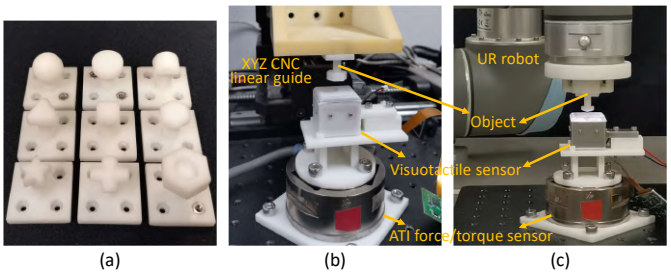

Fig. 8. Data collection setup. (a) The 3D printed objects. (b) The XYZ CNC linear guide platform for real data collection. (c) The UR robot platform for real data collection.

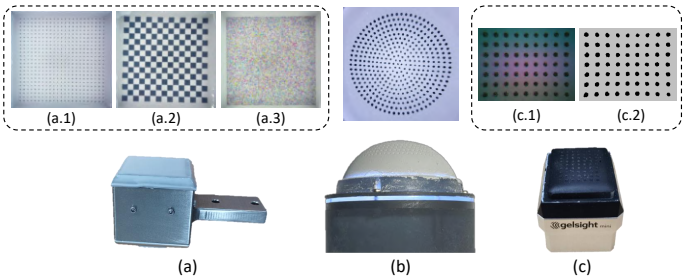

Fig. 9. Visuotactile sensors and their coating patterns. (a) GelStereo 2.0 sensor. (b) GelStereo Palm2.0 sensor. (c) GelSight Mini sensor.

1) Experimental Setup: In order to verify the simulation performance of TacFlex, we design a platform to collect tactile signals from real visuotactile sensors, as shown in Fig. 8. The visuotactile sensor is mounted on an ATI Gamma force/torque sensor fixed on the workbench. An XYZ Computer Numerical Control (CNC) linear guide (3-DOFs) and a UR robot (6-DOFs) are employed to drive the objects to contact with the visuotactile sensor. Specifically, the object is controlled to perform pressing, tangential sliding, and rotational sliding actions on the sensor surface to generate various contact states. During each object-sensor contact process, we record the object's trajectory, visuotactile signals, and force readings. The force readings are utilized to calibrate simulation parameters. Subsequently, the object's motion trajectory can be replicated in simulation to produce contact states consistent with the real world. In this way, we can obtain multiple sim-real tactile samples with various contact conditions.

2) Visuotactile Sensors: As shown in Fig. 9, we simulate several sensors with different coating patterns, surface shapes, and lighting conditions using TacFlex and carry out the evaluation experiments. Specifically, we design different coating patterns for GelStereo 2.0 sensor [21] to approximate different visuotactile sensors, including a marker pattern in Fig. 9 (a.1) (original GelStereo 2.0), a checkerboard pattern in Fig. 9 (a.2) (similar to [37]), and a random color pattern in Fig. 9 (a.3) (similar to DelTact [28]). The GelStereo Palm2.0 sensor [22] is employed to verify the effectiveness of the proposed simulation method for sensors with curved surfaces. Moreover, the GelSight Mini sensor[1] belonging to the GelSight family is simulated using TacFlex. Note that the raw tactile image (Fig. 9 (c.1)) cannot be used as pattern texture, as RGB lighting exists. The marker pattern (Fig. 9 (c.2)) is extracted from the raw tactile image by replacing the background.

3) Experimental Design: The tactile signals of these visuotactile sensors are simulated in TacFlex using the calibrated parameters. We design the following experiments to evaluate the simulation performance.

Exp. 1: Comparison of deformation simulation methods. We perform different deformation simulation methods to evaluate their effectiveness in simulating contact and elastic dynamics. These experiments are performed on a GelStereo 2.0 sensor with marker pattern under several contact conditions generated by pressing, tangential sliding, and rotational sliding. The Mean Euclidean Distance (MED) between the simulated and real markers is employed to evaluate the deformation simulation error, as the marker positions in 3D space are mainly sensitive to deformation simulation. Specifically, we carry out the non-physics-based, MPM-based, and FEM-based methods as follows:

- *Non-physics*: Given the object pose, we generate the deformed sensor surface by combining the undeformed sensor surface and the object surface. Then, we project the initial markers on the deformed sensor surface to obtain the markers in contact.
- *MPM*: Based on Tacchi [13], we use particles to represent markers, in which the markers share positions with corresponding particles.
- *FEM*: The simulation method of tactile 3D point clouds introduced in this paper is integrated with NVIDIA Isaac Gym, which provides FEM computing.

Exp. 2: Multi-mode tactile imprints simulation experiments. We comprehensively evaluate the simulation accuracy of tactile 3D point clouds, tactile images, and 2D marker motion fields to verify the performance of the proposed TacFlex. These experiments are carried out on a GelStereo 2.0 sensor with marker pattern under pressing contact conditions. The evaluation metrics for each mode are listed as follows:

- *Tactile 3D point clouds*: The MED between the simulated and real markers in 3D space is employed to evaluate the simulation error of tactile 3D point clouds.
- *Tactile images*: We compare the simulated and real tactile images at the pixel level using three commonly used metrics: Structural Similarity Index Measure (SSIM), Peak Signal-to-Noise Ratio (PSNR), and Mean Squared Error (MSE).
- *2D marker motion fields*: We detect the markers in both simulated and real tactile images and calculate the mean pixel distances between corresponding markers, a metric we refer to as Marker Pixel Distance Error (MPDE).

Exp. 3: Different coating patterns experiments. We replace the marker pattern in the GelStereo 2.0 sensor with a checkerboard pattern and a random color pattern, as shown in Fig. 9 (a.2) and (a.3). We simulate the tactile images of these sensors using TacFlex and evaluate the image similarity to verify the performance of TacFlex on different coating patterns.

Exp. 4: Different sensors experiments. We validate the simulation performance of TacFlex on sensors with different

[1]https://www.gelsight.com/gelsightmini/

IEEE Transactions on Robotics (T-RO) paper, presented at ICRA 2026, Vienna, Austria. Cite as T-RO paper.

configurations. In addition to the GelStereo 2.0 sensor, the GelStereo Palm2.0 with curved surface and GelSight Mini sensor with RGB lighting and marker pattern, as shown in Fig. 9 (b) and (c), are simulated using the proposed TacFlex.

Exp. 5: Method comparison of refraction simulation. To evaluate the simulation performance of refraction effects in tactile images, we conduct a series of method comparison experiments listed below. These experiments are conducted on all sensors in Fig. 9, and the image similarities are computed.

- *Without Refraction (WoR)*: The multi-medium refraction effect is not considered. Specifically, the tactile images are rendered in Pyrender without rectification.
- *Scale Transformation (ST)*: Similar to [16], the tactile images rendered in Pyrender are post-processed using scale transformation.
- *Ray tracing-based rectification (ours)*: The image transformation function is built based on the refractive ray tracing model and sensor parameters.

Exp. 6: Complex contact states experiments. We evaluate the simulation performance under complex contact states in tangential sliding, rotational sliding, and combinations, which are widespread in robotic applications. We visualize the tactile images and the marker motion fields of the GelStereo 2.0 and GelSight Mini sensors obtained from TacFlex simulation and the real world.

Exp. 7: Different FEM simulators experiments. The elastomer deformation simulation in this paper is mainly conducted in NVIDIA Isaac Gym [38]. We also integrate the proposed TacFlex with SOFA [32] and polyFEM supporting IPC [39] to verify the flexibility of the proposed framework. Furthermore, we evaluate the performance of these FEM simulators in terms of simulation accuracy and speed for elastomer deformation simulation.

Exp. 8: Simulation speed experiments. We use several objects to test the deformation simulation speed and accuracy with different elastomer mesh densities (i.e., mesh sizes). These experiments are conducted on a GelStereo 2.0 sensor with a marker pattern and in several FEM simulators.

Moreover, we compare TacFlex's tactile image rendering speed (frames per second, FPS) with that of Blender, a rendering engine that supports ray tracing. Different image resolutions are tested, including 1280×720, 640×480, and 320×240.

- *TacFlex*: We test the rendering speeds of unrefracted and refracted tactile image, respectively.
- *Blender*: We create the imaging scene of the visuotactile sensor in Blender, including the camera, light sources, and different media (i.e., air, acrylic plate, and gel). At each simulation step, the deformed textured sensor surface in Fig. 5 is loaded into the imaging scene in Blender, and the tactile image is generated after ray tracing rendering. We test the tactile image rendering speed in this case.

B. Implementation Details

1) Data Collection in Real World: We collect extensive real tactile signals to evaluate the fidelity of TacFlex simulation. Specifically, using the GelStereo 2.0 sensor with various coating patterns, the object is pressed at 3×3 grid positions on the sensor surface, with pressing depths ranging from 0.2 mm to 1.8 mm in 0.2 mm intervals. For the GelSight Mini sensor, the object is pressed at three positions on the sensor surface with depths ranging from 0.2 mm to 1.2 mm. Instead of using the platform in Fig. 8, the GelStereo Palm2.0 sensor is mounted on the end of a UR3 robot, which is then controlled to touch multiple positions on a large-scale object. Additionally, we collect real tactile signals under sliding contact on both the GelStereo 2.0 sensor with marker pattern and GelSight Mini sensor. In practice, the object is controlled to slide $0.3 \sim 5.0$ mm along the sensor surface using the CNC linear guide or rotate $5 \sim 60$ degrees around its central axis using the UR robot. The visuotactile sensor's program is running on a laptop (Intel Core i7-11800H @ 2.30GHz $\times$ 16), and the codes are implemented by Python using OpenCV, Open3D [40], and ROS package.

2) Simulation: In this paper, we employ the Flex physics engine in NVIDIA Isaac Gym to simulate elastomer deformation except for Exp. 7 and 8, and employ Pyrender as the render engine of tactile image. The Neo-Hookean model is utilized as the elastic model, and the detailed parameters in Exp. 1 - 6 are described as follows. For the GelStereo 2.0 sensor, $E = 2.5 \times 10^5 Pa$, $\nu = 0.47$, and $\mu = 0.3$. For the GelStereo Palm2.0 sensor, $E = 1.0 \times 10^5 Pa$, $\nu = 0.48$, and $\mu = 0.5$. For the GelSight Mini sensor, $E = 3.0 \times 10^5 Pa$, $\nu = 0.47$, and $\mu = 0.2$. We generate the volumetric mesh of the elastomer using Gmsh. Unless otherwise specified, the elastomer mesh of GelStereo 2.0 sensor contains 5252 vertices; the elastomer mesh of GelStereo Palm2.0 contains 3055 vertices; the elastomer mesh of GelSight Mini contains 7993 vertices. In Exp. 7 - 8, we also carefully calibrated the simulation parameters for each FEM simulator and each elastomer mesh. The cameras in visuotactile sensors are calibrated using Zhang's method. The refraction system parameters of GelStereo 2.0 and GelStereo Palm2.0 sensor are calibrated using the method in [22]. For GelSight Mini sensor, the structure parameters are manually measured, and the refractive indices are calibrated by minimizing the Euclidean distance errors between markers on the free sensor surface. The simulation is running on a desktop (Intel Core i9-13900KF $\times$ 32 CPU with NVIDIA RTX 4070Ti GPU).

C. Results and Analysis

Exp. 1: Table II presents the simulation errors of different deformation simulation methods under various contact states. The results indicate that the FEM-based method significantly outperforms the non-physics and MPM-based methods across all contact states. As the sliding distance or angle increases, the simulation errors of non-physics and MPM-based methods also increase, whereas those of the FEM-based method remain relatively stable. Moreover, we project the marker displacements in 3D space onto a 2D plane for easy observation, as illustrated in Fig. 10. The visualization results demonstrate that the marker displacements simulated using FEM are most similar to the real ones. The non-physics method fails to model

IEEE Transactions on Robotics (T-RO) paper, presented at ICRA 2026, Vienna, Austria. Cite as T-RO paper.

TABLE II
THE MED (MM) BETWEEN THE REAL MARKERS AND SIMULATED MARKERS FROM DIFFERENT DEFORMATION SIMULATION METHODS UNDER SEVERAL CONTACT STATES. THE SECOND ROW SHOWS THE PRESSING DEPTHS, SLIDING DISTANCES, AND ROTATION ANGLES.

Objects	Methods	Pressing			Tangential sliding			Rotational sliding		
		0.6 mm	1.2 mm	1.8 mm	0.4 mm	0.8 mm	1.8 mm	10°	20°	60°
	Non-physics	0.050	0.091	0.155	0.078	0.085	0.085	0.086	0.093	0.110
	MPM [13]	0.065	0.121	0.212	0.065	0.082	0.188	0.101	0.113	0.216
	FEM	**0.043**	**0.060**	**0.091**	**0.055**	**0.053**	**0.055**	**0.057**	**0.057**	**0.057**
	Non-physics	0.085	0.167	0.257	0.142	0.191	0.197	0.185	0.221	0.247
	MPM [13]	0.116	0.206	0.322	0.105	0.119	0.236	0.156	0.185	0.590
	FEM	**0.053**	**0.081**	**0.129**	**0.071**	**0.068**	**0.079**	**0.080**	**0.086**	**0.086**

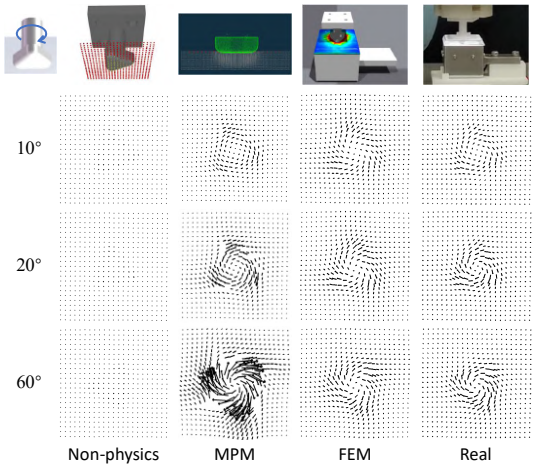

Fig. 10. The marker displacements from different deformation simulation methods.

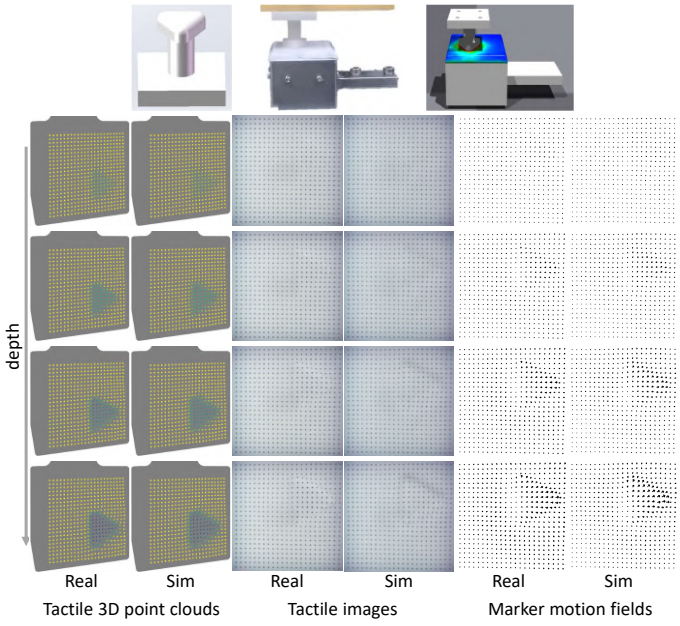

Fig. 11. Visualization of the simulated and real tactile imprints of GelStereo 2.0 sensor with marker pattern. The first row shows the object, the real scene, and the simulation scene. The second to fifth rows display the tactile 3D point clouds, tactile images, and marker motion fields as the contact depth increases. The tactile 3D point clouds are shown using a depth-dependent color map. The marker motion fields are denoted by arrows.

TABLE III
THE IMAGE SIMILARITY METRICS BETWEEN THE SIMULATED AND REAL TACTILE IMAGES.

Sensors	Methods	SSIM↑	PSNR↑	MSE↓	MPDE↓
GelStereo 2.0 (marker)	WoR	0.550	18.406	940.951	17.981
	ST	0.874	25.635	183.497	2.106
	ours	**0.936**	**29.349**	**101.583**	**1.102**
GelStereo 2.0 (checkerboard)	WoR	0.380	11.581	4518.385	/
	ST	0.811	20.470	591.240	/
	ours	**0.924**	**27.171**	**133.578**	/
GelStereo 2.0 (random color)	WoR	0.283	18.422	935.219	/
	ST	0.587	25.630	179.202	/
	ours	**0.865**	**31.470**	**50.381**	/
GelStereo Palm2.0	WoR	0.461	11.403	4708.390	24.661
	ST	0.515	12.163	3968.475	7.434
	ours	**0.660**	**15.492**	**1908.208**	**3.734**
GelSight Mini	WoR	0.494	19.081	804.309	14.403
	ST	0.760	25.577	183.779	1.070
	ours	**0.790**	**26.528**	**147.398**	**0.805**

the elastomer stretching caused by object sliding on the sensor surface, as this method relies on the object pose and does not account for its movement trajectory. Although the MPM-based method can simulate elastomer deformation effects under object rotation, the simulated marker displacements significantly deviate from the real ones as the rotation angle increases. This discrepancy may be attributed to sliding between the object and the sensor surface at larger rotation angles, which the MPM-based method struggles to handle accurately due to its limitations in addressing friction. The FEM-based method performs well in simulating elastomer deformation under various contact states, which implies the necessity of FEM-based deformation simulation in the proposed TacFlex.

Exp. 2: Fig. 11 displays the simulated and real multi-mode tactile imprints of the GelStereo 2.0 sensor in a pressing process. The simulated tactile 3D point clouds could describe the object shape well, and the MED error on diverse objects is 0.075 mm. Quantitatively, the image similarity metrics are listed in Table III. The SSIM achieves 0.936; the PSNR is 29.349; and the MSE is 101.583. Additionally, the marker motion fields intuitively illustrate the stretching of the marker pattern under contact, where the MPDE is 1.102 pixels.

Exp. 3: The simulated and real tactile images of GelStereo 2.0 sensors with checkerboard pattern and random color pat-

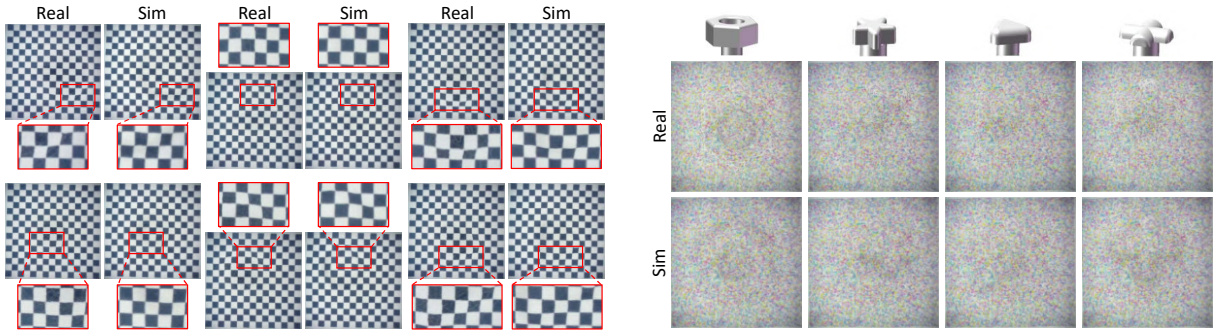

Fig. 12. Visualization of the simulated and real tactile images of GelStereo 2.0 sensor with checkerboard pattern (left) and random color pattern (right).

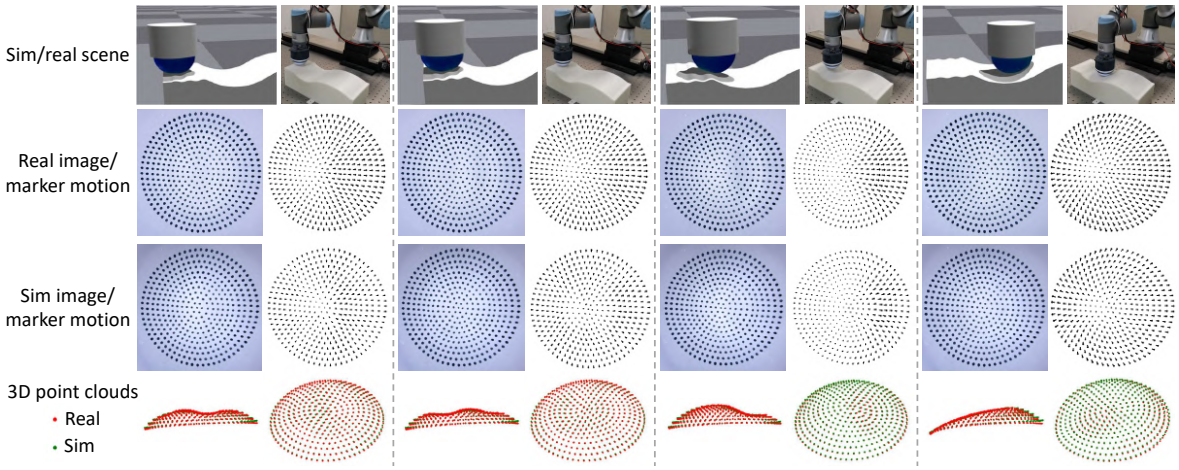

Fig. 13. Visualization of the simulated and real tactile imprints of GelStereo Palm2.0 sensor. The first row is the simulation and real scene. The second row shows the real tactile images and marker motion fields. The third row displays the simulated ones. The fourth row shows the simulated and real tactile 3D point clouds from two views.

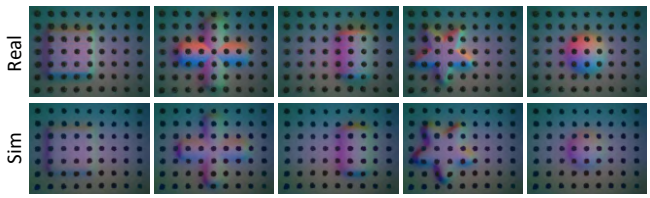

Fig. 14. The simulated and real tactile images of GelSight Mini sensor.

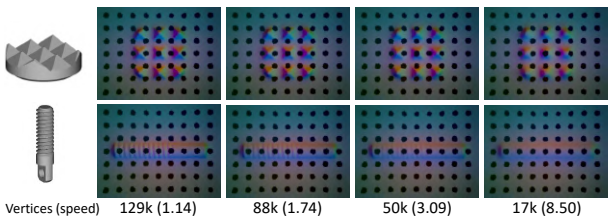

Fig. 15. The simulated tactile images of GelSight Mini sensor in contact with textured objects. The bottom row shows vertices of the elastomer mesh and the FEM simulation speed (steps/s) in NVIDIA Isaac Gym.

tern are displayed in Fig. 12. In the simulation, we observe that the lines of the checkerboard bend due to contact, closely mirroring real-world elastomer behavior. The SSIM of tactile images with checkerboard pattern and random color pattern is 0.924 and 0.865, respectively, as shown in Table III. These results indicate that TacFlex has the capability to accurately simulate various coating patterns, highlighting its potential for advancing research in visuotactile sensing.

Exp. 4: The proposed TacFlex is also employed to simulate different types of visuotactile sensors. Fig. 13 illustrates the simulation outputs of the GelStereo Palm2.0 sensor as it touches a large-scale object. The MED error of tactile 3D point clouds is 0.459 mm. The simulation performs well even with large elastomer deformation, as illustrated at the bottom of Fig. 13, where the hemispherical elastomer deforms dramatically into a saddle shape in the first scene. Additionally, Fig. 14 displays the simulated and real tactile images of the GelSight Mini sensor. The SSIM of tactile images is 0.790, and the MPDE is 0.805 pixels. Note that the markers on tactile images are directly rendered in TacFlex rather than synthesized by RGB optical simulation and marker motion as in [24]. Fig. 15 illustrates the simulation performance of the GelSight Mini on textured objects such as a screw. Increasing the mesh density enhances the texture detail of objects in the simulated images, but also raises the computational cost of FEM simulation.

Exp. 5: Table III demonstrates the similarity metrics of simulated and real tactile images using different refraction simulation methods. If the refraction effect is not considered in simulation, significant discrepancies exist between the sim-

IEEE Transactions on Robotics (T-RO) paper, presented at ICRA 2026, Vienna, Austria. Cite as T-RO paper.

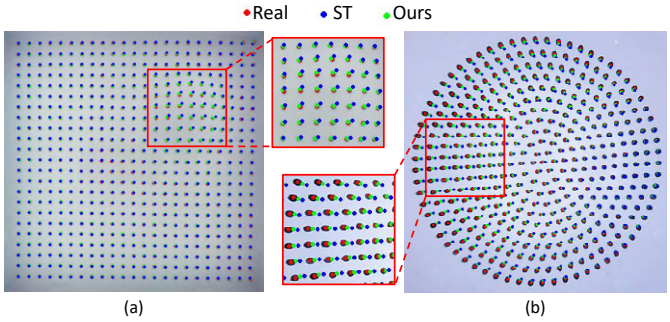

Fig. 16. The markers on tactile images generated using Scale Transformation (ST) and our proposed rectification method. We draw the simulated and real marker positions on a single image to show the similarity. (a) The GelStereo 2.0 sensor with marker pattern contacting with a sphere. (b) The GelStereo Palm2.0 sensor touching an object.

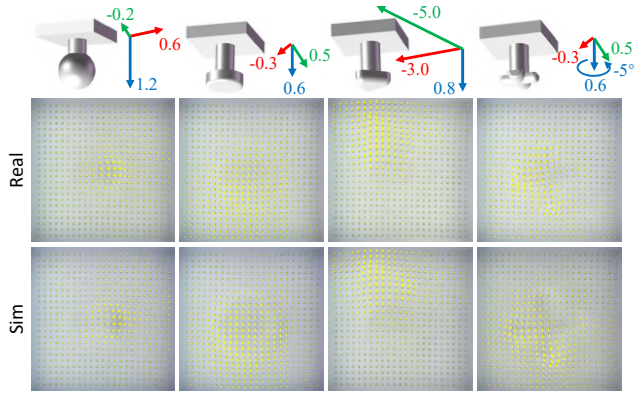

Fig. 17. Visualization of simulated and real tactile images and marker motion fields under various contact states on GelStereo 2.0 sensor. The first row shows the object's movement on the sensor surface, in which the unit is mm. In the second and third rows, the yellow arrows on tactile images, indicating the marker motion fields, are drawn for visualization, and not included in simulation outputs.

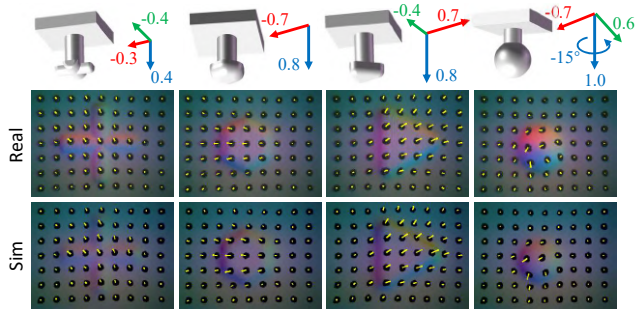

Fig. 18. Visualization of simulated and real tactile images and marker motion fields under various contact states on GelSight Mini sensor. The first row shows the object's movement on the sensor surface, in which the unit is mm. In the second and third rows, the yellow arrows on tactile images, indicating the marker motion fields, are drawn for visualization, and not included in simulation outputs.

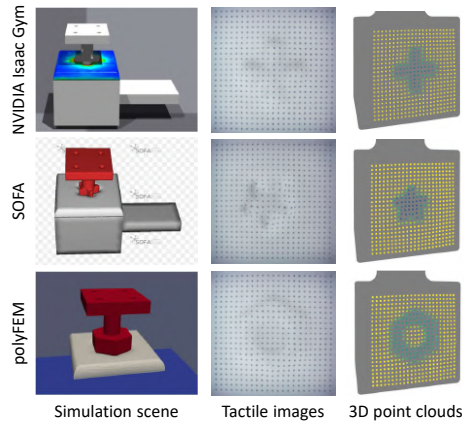

Fig. 19. TacFlex connected with different FEM simulators. The deformation simulation is conducted using NVIDIA Isaac Gym, SOFA, and polyFEM. TacFlex simulates the corresponding tactile imprints.

ulated and real tactile images, where the markers' imaging positions are seriously misplaced and the SSIM is even lower than 0.5. Our ray tracing-based rectification method achieves higher image similarities than the scale transformation (ST) method. For example, the SSIM of the proposed method achieves 0.865 on the GelStereo 2.0 sensor with a random color pattern, while the SSIM of ST is only 0.587.

As shown in Fig. 16, we draw the simulated markers on the real tactile images to visualize the performances of the ST method and our proposed rectification method. On the GelStereo 2.0 sensor, the pixel coordinates of markers in non-contact regions from both ST and the proposed method are almost consistent with the real markers. However, in contact regions, the markers obtained by the proposed method are much closer to the real positions than those obtained by ST. We also notice that the larger the contact depth, the greater the simulation error of the ST method. On the GelStereo Palm2.0 sensor, the marker positions from ST method are quite different from the real ones. Probably because the sensor surface is curved, where the depths are seriously inconsistent. In these scenes, applying ray tracing-based rectification seems more significant. Additionally, these results suggest that introducing depth into refraction rectification is necessary and effective.

Exp. 6: Fig. 17 and Fig. 18 display the simulation results of GelStereo 2.0 sensor and GelSight Mini sensor under complex contact states, respectively. We observe that the simulated and real tactile images and marker motion fields seem much similar. For example, column 3 of Fig. 17 illustrates the stretching of the sensor surface when a triangular object slides over a long distance. During the object's movement, the contact between the object and the sensor surface switches from stability to slippage. These results demonstrate that the proposed TacFlex performs well in simulating the visuotactile imprints under various complex contact states and is effective in simulating contact and elastic dynamics, which is important for contact-rich robot manipulation tasks.

Exp. 7: In addition to NVIDIA Isaac Gym, we connect the proposed TacFlex with other FEM simulators, including SOFA and polyFEM. Fig. 19 displays the deformation simulation scene and the simulated tactile imprints. Moreover, Table IV presents the deformation simulation errors and speeds of different FEM simulators under various contact conditions. Among the three simulators, Isaac Gym demonstrates the highest simulation accuracy for over half of the contact samples. The polyFEM is slightly less accurate than Isaac Gym, but the difference remains marginal. Also, it exhibits an accuracy

IEEE Transactions on Robotics (T-RO) paper, presented at ICRA 2026, Vienna, Austria. Cite as T-RO paper.

TABLE IV
THE DEFORMATION SIMULATION ERRORS INDICATED BY MED (MM) BETWEEN THE REAL MARKERS AND SIMULATED MARKERS FROM DIFFERENT FEM SIMULATORS UNDER SEVERAL CONTACT STATES. THE SECOND ROW SHOWS THE PRESSING DEPTHS, SLIDING DISTANCES, AND ROTATION ANGLES. THE LAST COLUMN GIVES THE SIMULATION SPEED. THE ELASTOMER MESH DENSITY (ABOUT 2.0K VERTICES) IS CONSISTENT ACROSS SIMULATORS.

Objects	FEM simulators	Pressing			Tangential sliding			Rotational sliding			Sim speed (steps/s)
		0.6 mm	1.2 mm	1.8 mm	0.4 mm	0.8 mm	1.8 mm	10°	20°	60°	
	Isaac Gym [38]	0.043	0.065	0.105	**0.049**	0.052	0.056	**0.057**	**0.057**	**0.059**	**36.987**
	SOFA [32]	**0.042**	0.062	0.101	0.072	0.075	0.067	0.067	0.071	0.084	6.422
	polyFEM [39]	0.043	**0.057**	**0.080**	0.062	**0.050**	**0.051**	**0.057**	0.058	0.069	0.448
	Isaac Gym [38]	**0.056**	**0.098**	0.159	**0.063**	**0.070**	**0.068**	**0.084**	**0.094**	**0.103**	**37.188**
	SOFA [32]	0.059	0.101	0.153	0.130	0.174	0.176	0.137	0.170	0.179	6.865
	polyFEM [39]	0.074	**0.098**	**0.127**	0.101	0.099	0.084	0.096	0.101	0.120	0.256

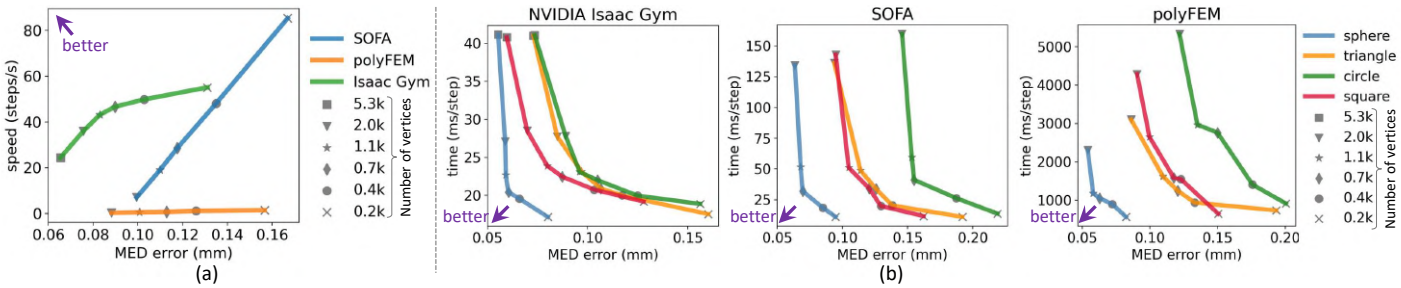

Fig. 20. Effect of elastomer mesh density on speed/time and error of FEM-based deformation simulation. (a) The simulation speed-error curve of different FEM simulators. (b) The detailed simulation time-error curves of several objects for each FEM simulator. The speed testing is carried out on a desktop with Intel Core i9-13900KF CPU and NVIDIA RTX 4070Ti GPU.

advantage at larger pressing depths (e.g., 0.080 mm compared to 0.105 mm at a depth of 1.8 mm). Furthermore, SOFA achieves comparable accuracy to Isaac Gym and polyFEM in simulations involving pressing contact. However, its performance is less accurate in sliding contact scenarios (e.g., 0.176 mm compared to 0.068 mm and 0.084 mm).

Furthermore, we plot the speed-error curves of these FEM simulators with different elastomer mesh densities, as illustrated in Fig. 20 (a). The results indicate that as the mesh density decreases, both the simulation error and speed increase. The Isaac Gym could achieve real-time simulation even with dense meshes, simulating about 25 steps per second using 5.3k elastomer vertices. The simulation speed of SOFA is heavily influenced by mesh density. With a sparse mesh containing 0.2k vertices, SOFA achieves a simulation speed exceeding 80 steps per second, surpassing the other two simulators. In contrast, the polyFEM is very slow, even with a sparse mesh. Overall, these experiments suggest that Isaac Gym achieves a good trade-off between accuracy and speed, making it well-suited for deformation simulation within the TacFlex pipeline. It is important to emphasize that the deformation simulation is not restricted to the FEM simulators discussed. Any simulator capable of outputting deformed elastomer meshes could be integrated into TacFlex.

Exp. 8: We present an in-depth analysis of the effect of mesh density on simulation time and accuracy, facilitating the selection of appropriate finite element mesh. Fig. 20 (b) shows simulation time-error curves in each FEM simulator. These results demonstrate that increasing mesh density can enhance simulation accuracy, albeit at the cost of increased computational time. Reducing the mesh density appropriately can greatly shorten the computation time without significant loss of accuracy. For instance, in SOFA, reducing the number of mesh vertices from 2.0k to 1.1k decreases the simulation time by approximately 60%, with an accuracy loss of less than 0.01 mm. However, when the number of vertices falls below a certain threshold (0.4k in this case), the simulation error increases substantially. A favorable trade-off between simulation time and accuracy is achieved when the mesh density is within the range of the two aforementioned cases. The mesh density could be selected according to specific requirements by referring to the time-error curves.

TABLE V
THE SPEED (FPS) OF TACTILE IMAGES RENDERING WITH DIFFERENT IMAGE RESOLUTIONS.

	Resolution	Unrefracted image	Refracted image
TacFlex	1280×720	56.40	**18.76**
	640×480	83.59	**32.44**
	320×240	172.97	**67.20**
Blender	1280×720	/	0.89
	640×480	/	1.89
	320×240	/	3.47

The unrefracted image rendering in TacFlex uses the Pyglet backend in Pyrender. The speed testing is carried out on a desktop with Intel Core i9-13900KF CPU and NVIDIA RTX 4070Ti GPU. The time spent on deformation simulation is not included.

Furthermore, the rendering speed of tactile images is shown in Table V. TacFlex can simulate tactile images with refraction effects at 67.2 FPS with an output resolution of 320×240 pixels. This rendering speed exceeds the image reading speed in real sensors. However, the rendering speed using Blender

IEEE Transactions on Robotics (T-RO) paper, presented at ICRA 2026, Vienna, Austria. Cite as T-RO paper.

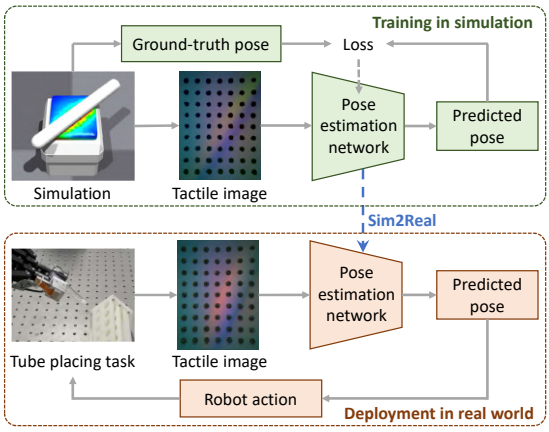

Fig. 21. The cylindrical object pose estimation pipeline using Sim2Real (from TacFlex to reality) framework.

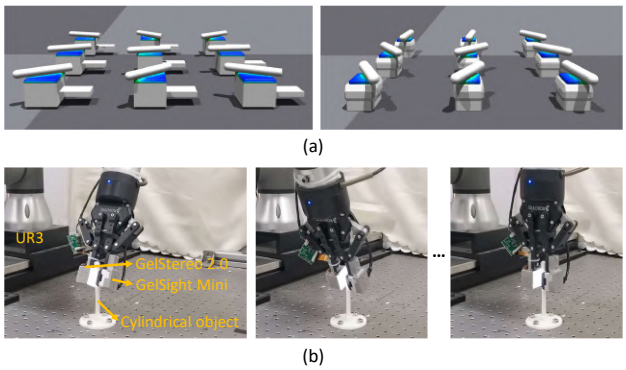

Fig. 22. Collecting tactile images and ground-truth poses of cylindrical objects using GelStereo 2.0 sensor and GelSight Mini Sensor. (a) The simulation scene. (b) The real world scene.

is lower than 5 FPS, which is much slower than TacFlex pipeline. These results demonstrate that TacFlex could realize time-efficient and accurate simulation of tactile images.

V. Applications of TacFlex in Tactile Perception and Manipulation Tasks

This section explores two Sim2Real applications based on TacFlex, including cylindrical object pose estimation and peg-in-hole task. In these applications, the perception and policy neural networks are trained in TacFlex simulation and subsequently deployed in the real world.

A. Cylindrical Object Pose Estimation

We apply a cylindrical object pose estimation task to show how TacFlex can be applied to learning-based tactile perception, as shown in Fig. 21. We first train a pose estimation network using a training dataset generated in TacFlex simulation. The dataset contains tactile image and object pose pairs. Then, we deploy this network in the real world, in which we evaluate the pose estimation errors on real sensors and carry out a tube placing task by estimating the tube pose once.

1) Setups: The goal of this task is to estimate the position of the contact center and the angle of a cylindrical object with respect to the visuotactile sensor. For simplicity, the axis of the cylindrical object is restricted to be parallel to the sensor surface. This pose estimation task is conducted on both GelStereo 2.0 sensor and GelSight Mini sensor.

We design several cylinders with different diameters ranging from 6 mm to 25 mm. The cylinders are loaded into the simulation with random poses under the parallel restriction and are controlled to press on the sensor surface to various depths, as shown in Fig. 22 (a). The simulated tactile images and cylinder poses in the sensor coordinate system are recorded in the pressing process. We use 9 parallel environments to collect the training dataset, containing about 30k samples for each sensor.

Since it is difficult to directly obtain the ground-truth pose of the objects in the real world, we design a specific data collection platform, as shown in Fig. 22 (b). The cylindrical object is fixed on the workbench. A GelStereo 2.0 sensor and a GelSight Mini sensor are mounted on the fingertips of a Robotiq-85 gripper. We program the UR3 robot to grasp the cylindrical object with various poses, and then convert the gripper poses to object poses with respect to the sensor coordinate system. In this way, we collect 200 samples for each sensor and each object in the real world, where cylindrical objects with 8 mm, 10 mm, 12 mm, and 15 mm diameters are employed.

To verify the Sim2Real performance, we compute the Mean Absolute Error (MAE) of the estimated position and rotation under several training and testing configurations. 1) **Sim**: The network is trained and tested on simulation data. The testing set is not included in the training set. 2) **Sim2Real**: The network is trained on simulation data and directly evaluated using real data. All the following Sim2Real configurations are tested on the same real dataset. 3) **LR+Sim2Real**: The network is trained using simulation images, in which the light densities are randomized during image rendering. 4) **CJ+Sim2Real**: Color jittering (including adjustment of brightness, contrast, saturation, and hue) is applied to the simulated tactile images in the training process. 5) **GN+Sim2Real**: We add Gaussian noise to the simulated tactile images. 6) **CJ+GN+Sim2Real**: Both color jittering and Gaussian noise are applied to the simulated tactile images.

Moreover, we utilize the pose estimation model to complete a *tube placing task* to further verify the Sim2Real performance, in which the robot action is generated based on the estimated tube pose, as shown in Fig. 23 (a). A double-layered and tilted tube rack with a known pose is fixed on the workbench. The success rate of this task mainly depends on the accuracy of object pose estimation.

2) Implementation Details: We train a ResNet-50 [41] neural network using the simulated dataset to learn the pose estimation. The mean squared error of position and rotation is employed as the loss function.

The tube placing process is described as follows. We hand a tube to the robot with a random pose. The gripper closes to grasp the tube. Given the estimated tube pose ${}^{s}\mathbf{T}_{t} \in \mathbb{R}^{4\times4}$ in sensor coordinate system, the target pose of gripper ${}^{r}\mathbf{T}_{g}^{target} \in \mathbb{R}^{4\times4}$ in robot coordinate system (i.e., the robot action) can

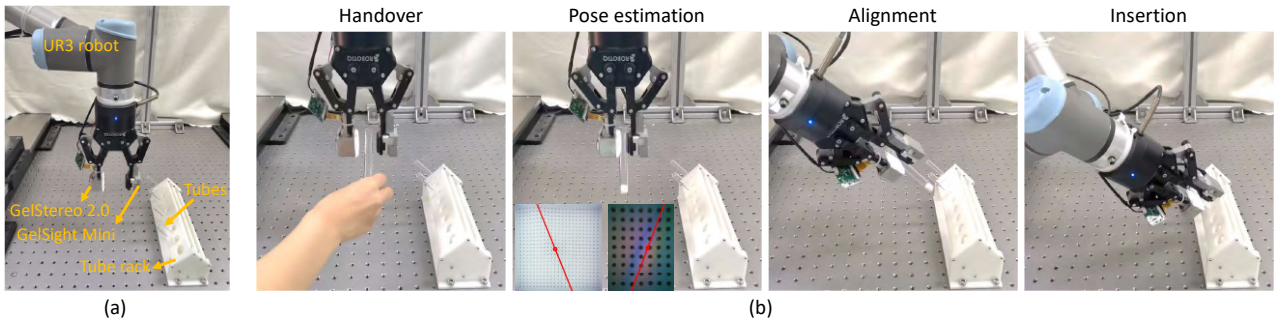

Fig. 23. The tube placing task in the real world. (a) The experiment platform. (b) The whole process of the tube placing task which contains four phases, including handover, pose estimation, robot pose adjustment to align the tube and rack, and insertion.

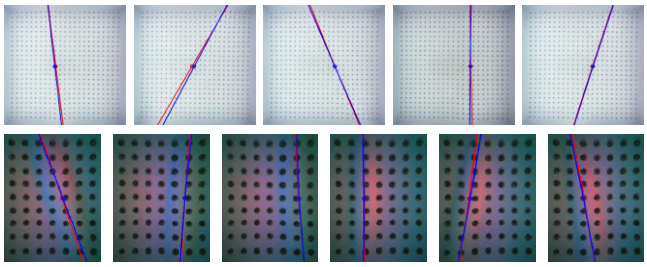

Fig. 24. Visualization of the ground-truth (blue) and predicted (red) object poses on real tactile images. The first and second rows show the results of GelStereo 2.0 and GelSight Mini sensor, respectively. The lines indicate the rotation, and the small circles denote the contact center.

TABLE VI
THE POSE ESTIMATION ERRORS OF CYLINDRICAL OBJECTS. THE UNIT FOR POSITION ERROR IS MM; THE UNIT FOR ROTATION ERROR IS DEGREES.

Configurations	GelStereo 2.0		GelSight Mini	
	position	rotation	position	rotation
Sim	0.055	0.463	0.013	0.185
Sim2Real	1.227	2.977	1.281	8.257
LR+Sim2Real	**0.600**	**1.406**	1.715	6.338
CJ+Sim2Real	0.888	2.587	0.351	3.057
GN+Sim2Real	0.992	2.286	0.400	2.751
CJ+GN+Sim2Real	0.970	2.004	**0.336**	**1.741**

be computed using

$$ {}^{r}\mathbf{T}_{g}^{target} = {}^{r}\mathbf{T}_{t}^{target} \times ({}^{g}\mathbf{T}_{s} \times {}^{s}\mathbf{T}_{t})^{-1} \tag{21} $$

where ${}^{r}\mathbf{T}_{t}^{target} \in \mathbb{R}^{4\times4}$ denotes the known target tube pose in robot frame, and ${}^{g}\mathbf{T}_{s} \in \mathbb{R}^{4\times4}$ is the transformation matrix between the sensor frame and gripper frame.

3) Results: Fig. 24 visualizes the ground-truth and predicted poses on real tactile images, suggesting the estimated poses closely align with the ground truth on the GelStereo 2.0 sensor and GelSight Mini sensor. Table VI shows the pose estimation errors. The pose errors in the simulation are minor, indicating that the neural network is well-trained. The model trained on simulated data without any augmentation (Sim2Real) exhibits a satisfactory performance on the real GelStereo 2.0 sensor, with mean position and rotation errors of 1.227 mm and 2.977 degrees, respectively. In contrast, the rotation error observed on the real GelSight Mini sensor is higher, at 8.257 degrees. These results suggest that the sim-real gap on GelStereo 2.0 is slighter than that on the GelSight Mini sensor. The multi-color light sources increase the difficulty of simulating highly realistic tactile images for GelSight Mini. Applying advanced rendering techniques might further narrow this sim-real gap. Moreover, we find that data augmentation can enhance the Sim2Real performance. Specifically, randomizing light densities (LR+Sim2Real) could reduce the pose estimation error by 50% on the GelStereo 2.0 sensor. For the GelSight Mini sensor, color jittering and Gaussian noise (CJ+GN+Sim2Real) achieve the best performance on the real data, where the mean position and rotation errors are 0.336 mm and 1.741 degrees, respectively. The performance differences of data augmentation methods on these two sensors may result from the varying characteristics of tactile images, where the images of GelSight Mini have rich color information, while the color of each pixel on GelStereo 2.0's image is nearly uniform.

TABLE VII
THE SUCCESS RATES OF THE TUBE PLACING TASK.

Tubes	GelStereo 2.0	GelSight Mini	GelStereo 2.0 & GelSight Mini
8 mm	26/30 (86.7%)	27/30 (90.0%)	29/30 (96.7%)
10 mm	26/30 (86.7%)	27/30 (90.0%)	28/30 (93.3%)
12 mm	27/30 (90.0%)	27/30 (90.0%)	29/30 (96.7%)
15 mm	26/30 (86.7%)	25/30 (83.3%)	29/30 (96.7%)
all	105/120 (87.5%)	106/120 (88.3%)	115/120 (95.8%)

In the tube placing task, we utilize the tube pose estimated from three different setups: using only the GelStereo 2.0 sensor, using only the GelSight Mini sensor, and using a fusion of both the GelStereo 2.0 and GelSight Mini sensors. We conduct 30 trials for each setup and each tube. Fig. 23 (b) displays the whole process of a tube placing task. The success rates are listed in Table VII. The success rates are 87.5% and 88.3% when using the GelStereo 2.0 sensor and the GelSight Mini sensor, respectively. Fusing the poses estimated by these two sensors increases the success rate to 95.8%.

Note that the simulated and real tactile images are identically processed before feeding into the network in the testing phase. No additional processing, such as scaling, is applied to the simulated images. The performances of the cylindrical object pose estimation task in this section have illustrated that

IEEE Transactions on Robotics (T-RO) paper, presented at ICRA 2026, Vienna, Austria. Cite as T-RO paper.

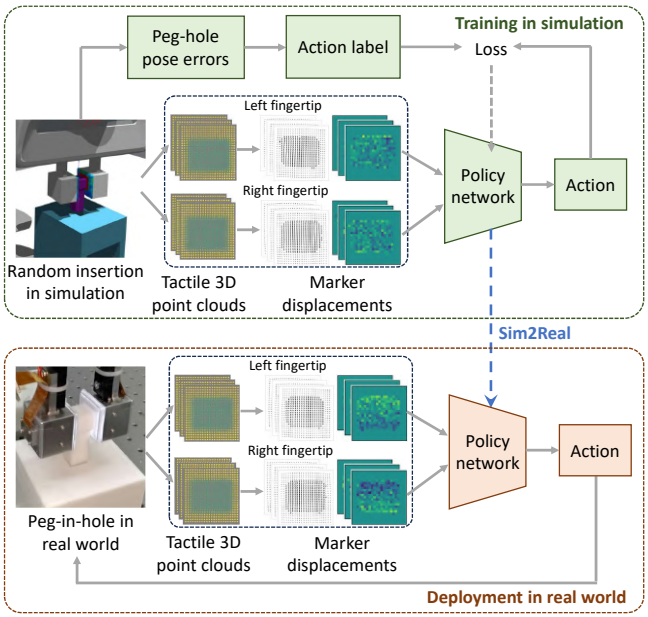

Fig. 25. The peg-in-hole pipeline using Sim2Real (from TacFlex to reality) framework.

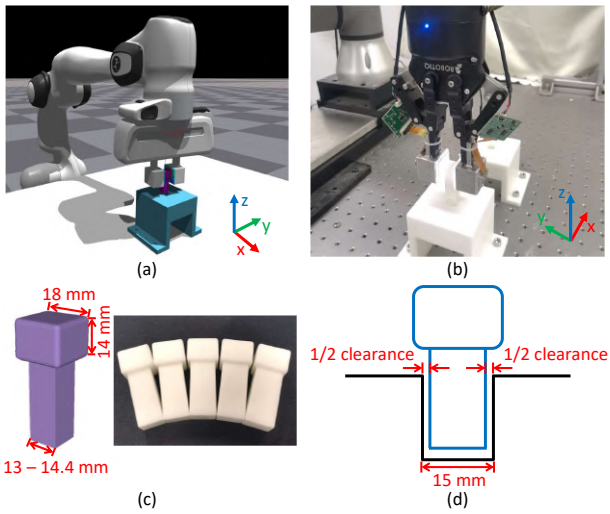

Fig. 26. Peg-in-hole task setups. (a) The task scene in simulation. (b) The task scene in the real world. (c) The pegs. (d) The definition of clearance in this paper.

the proposed tactile image simulation method in this paper is effective and could support zero-shot transfer of tactile perception models from simulation to reality.

B. Peg-in-Hole Task

Here, we carry out a peg-in-hole task to show how TacFlex can be applied to skill learning for robot manipulation tasks, as shown in Fig. 25. We employ the tactile 3D point clouds of GelStereo 2.0 sensors as the observation and feed the sequences of marker displacements into the policy network. The insertion policy is trained in the simulation with the principle of behavior cloning, and then deployed on a real robot.

1) Setups: In this peg-in-hole task, a gripper with tactile sensing is controlled to insert a peg into a squared hole through several attempts, as shown in Fig. 26. Compared to [7] [42], the task setting in this paper is more challenging. The length and width of the peg are much shorter than that of the visuotactile sensor, resulting in a weaker tactile response during peg-hole contact. Also, we try to insert several pegs of different sizes, the assembly clearance of which ranges from 2.0 mm to 0.6 mm.

For each episode in this task, the gripper first grasps the peg and moves to the top of a 15 mm $\times$ 15 mm hole with a random 3-DOFs misalignment error. Specifically, the translation error is in the x-axis and y-axis, each ranging from -2.5 mm to 2.5 mm; the rotation error is around the z-axis (indicated by θ), ranging from -5° to 5°. Then, the gripper moves downward in the z-axis to attempt insertion, while monitoring peg-hole collisions using the total displacements of tactile markers. If the displacements exceed a predefined threshold, we infer that a collision occurs and this attempt fails. Once a collision occurs, the gripper keeps moving down a further 0.4 mm and then lifts up. The tactile 3D point clouds of these two GelStereo 2.0 sensors during collision are pre-processed and fed into the policy model to generate the robot action $(\Delta x, \Delta y, \Delta\theta)$ for peg pose adjustment, and then the next attempt starts. If the collision is not detected while the gripper moving down to a predetermined position, that indicates a successful insertion. The maximum number of attempts is 15. Otherwise, the task fails.

We develop a simulation environment of this peg-in-hole task in NVIDIA Isaac Gym integrated with TacFlex, as shown in Fig. 26 (a). A random policy is conducted to attempt peg insertion in simulation. In each attempt, the tactile 3D point clouds during collision and the peg-hole pose errors before collision are recorded to create a simulated dataset. Similarly, we collect 150 samples in the real world and create a small real dataset. The peg pose cannot be directly accessed in the real world. Instead, we use the gripper pose for approximation.

To verify the Sim2Real performance, we test the success rates of the peg-in-hole task conducted in four cases. The training data is collected *only using the peg with 2.0 mm clearance*, and the policy is tested using pegs with different clearances. 1) **Real**: The policy network is trained only using the real dataset containing 150 samples and tested in the real world. 2) **Sim**: The policy network is trained only using simulated data and tested in the simulation environment. 3) **Sim2Real**: The policy network is trained only using simulated data and tested in the real world. 4) **Sim2Real+Finetune**: The policy network is firstly trained using simulated data, and then the output layer is fine-tuned using the small real dataset. The policy is tested in the real world. This task is tested 100 times in the simulation environment and 30 times in the real world for each case and each peg.

2) Domain Randomization for Sim2Real: To enhance Sim2Real performance, we apply the domain randomization technique to the physical and task-related parameters in the simulation, and add random noises to the simulated tactile observations. The details are shown in Table VIII. The physical parameters, including Young's modulus, Poisson ratio, and

IEEE Transactions on Robotics (T-RO) paper, presented at ICRA 2026, Vienna, Austria. Cite as T-RO paper.

TABLE VIII
THE SETTINGS OF DOMAIN RANDOMIZATION PARAMETERS.

	Parameter names	Distribution
Physical	Young's modulus (Pa)	U(1.0e5, 5.0e5)
	Poisson ratio	U(0.3, 0.48)
	Friction coefficient	U(0.2, 0.7)
Task-related	Peg offset x in gripper (mm)	U(-1.0, 1.0)
	Peg offset z in gripper (mm)	U(-1.0, 1.0)
	Contact depth (mm)	U(0.5, 1.2)
Tactile noises	Scaling of displacements	U(0.5, 2.0)
	Each value in displacements (mm)	N(0, 0.01)

U(low, high) indicates a uniform distribution.
N(mean, std) indicates a Gaussian distribution.

friction coefficient, are randomized to deal with elastomer aging and environmental changes. We randomize the peg positions in the gripper and the gripper widths to improve the adaptability of the policy in this task. Moreover, we add Gaussian noises to each value in the marker displacements to make the policy insensitive to noises in real sensors, and randomly scale the displacements in each sample to make the training process focus on the distribution of marker displacements but not the magnitude.

3) Details of Policy Training: We process the tactile 3D point clouds and peg-hole pose errors to create the training dataset for behavioral cloning. We generate the action labels from the peg-hole pose errors as follows:

$$\hat{a}_x = \mathbb{F}_{clip}(-e_x + \mathbb{F}_{sgn}(e_x) \cdot c/2, 0, -\delta \cdot \mathbb{F}_{sgn}(e_x)) \quad (22)$$

$$\hat{a}_y = \mathbb{F}_{clip}(-e_y + \mathbb{F}_{sgn}(e_y) \cdot c/2, 0, -\delta \cdot \mathbb{F}_{sgn}(e_y)) \quad (23)$$

$$\hat{a}_\theta = \mathbb{F}_{clip}(-e_\theta, -1.5°, 1.5°) \quad (24)$$

where $(\hat{a}_x, \hat{a}_y, \hat{a}_\theta)$ is the action label of translation in x-axis and y-axis, and rotation around z-axis. (e_x, e_y, e_θ) is the peg-hole pose error which can be directly accessed in simulation. $\mathbb{F}_{clip}(x, a, b) = \max(\min(a, b), \min(x, \max(a, b)))$ is the function that constrains the output within a given range a and b. $\mathbb{F}_{sgn}(x)$ is the sign function. The parameter δ is introduced to constrain the action label, which may assist in policy learning, as we find that tactile observations have difficulties in distinguishing large peg-hole pose errors. Based on our observation, δ is set to 1 mm. c is the assembly clearance. Introducing the clearance into the above process aims to generate the minimum action required to insert the peg into the hole. In this way, the training process could focus on learning the contact states of the peg and hole, regardless of the assembly clearance. In addition, we compute the marker displacements caused by peg-hole collision as the network input in this task. Specifically, we sample three frames of tactile 3D point clouds starting from the collision to the end of the insertion. For each sensor, the marker displacements $\Delta\mathbf{P}^{(i)}$ can be obtained by

$$\Delta\mathbf{P}^{(i)} = \mathbf{P}^{(i)} - \mathbf{P}^{(0)} \in \mathbb{R}^{h \times w \times 3}, i = 1, 2, 3 \quad (25)$$

where $\mathbf{P}^{(i)}$ indicates the ith tactile 3D point clouds in the sequence. $\mathbf{P}^{(0)}$ denotes the 3D point clouds just before peg-hole collision. The marker displacements are organized in the form of matrices, similar to the marker distribution on the GelStereo 2.0 sensor with $h = w = 23$. Moreover, since the contact states of the peg and hole are primarily indicated by the markers in contact regions, we apply a contact mask $\mathbf{M}$ to the marker displacements $\Delta\mathbf{P}$ using:

$$\Delta\mathbf{P}_m^{(i)} = \Delta\mathbf{P}^{(i)} \odot \mathbf{M}^{(i)}, i = 1, 2, 3 \quad (26)$$

where the $\mathbf{M}^{(i)} \in \mathbb{R}^{h \times w}$ is a 0-1 array, in which 0 indicates non-contact, and 1 indicates contact. $\odot$ denotes element-by-element multiplication. After the processing, about 70k simulated samples, each containing a tactile observation and an action label, are generated for policy training.

Further, we employ Convolutional Neural Networks (CNN) followed by Multi-Layer Perceptrons (MLP) to extract tactile features from the marker displacements in each direction. After that, the features from the three directions and two sensors are concatenated and fed into an MLP to infer the robot action $(\Delta x, \Delta y, \Delta\theta)$. The mean squared error is employed as the loss function.

4) Results: The process of a successful experiment in the real world is shown in Fig. 27. We can observe that the peg-hole pose error gradually decreases after several pose adjustments. The marker displacements are also visualized for each insertion attempt, implying different peg-hole contact states. With a small y-axis peg-hole position error and a larger +x-axis error in attempt ② and ③, the peg undergoes a slight rotation around the -y-axis, as indicated by the marker displacements on the sensor surface plane. The policy network responds by outputting actions that significantly reduce the error in the x direction, although the actions in the y direction are somewhat uncertain. In attempt ④, the pose error only exists in the +y direction, causing the peg to rotate slightly around the +x-axis, which results in noticeable marker displacements in the vertical direction of the sensor surface. Although a mistaken action may occur, it can be corrected in subsequent attempts.

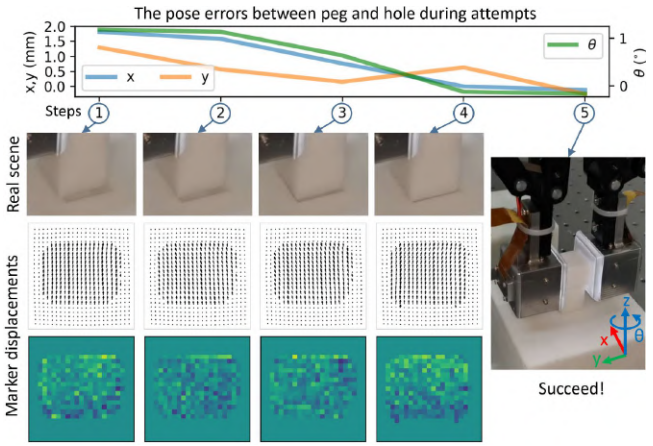

Fig. 27. The snapshots of a peg-in-hole task in the real world showing the insertion scenes and tactile marker displacements. The pose errors between peg and hole gradually decrease after several insertion attempts. The third and fourth rows show the marker displacements in the left sensor, in which the black arrow fields show the marker displacements on the sensor surface plane, and the heat maps show the marker displacements in the vertical direction of the sensor surface. Only the last frame of a tactile sequence is displayed in this figure. The assembly clearance in this scene is 0.6 mm.

IEEE Transactions on Robotics (T-RO) paper, presented at ICRA 2026, Vienna, Austria. Cite as T-RO paper.

TABLE IX
THE SUCCESS RATES OF THE PEG-IN-HOLE TASK WITH DIFFERENT ASSEMBLY CLEARANCES.

Clearance	2.0 mm		1.6 mm		1.2 mm		0.8 mm		0.6 mm	
	Success	Attempts	Success	Attempts	Success	Attempts	Success	Attempts	Success	Attempts
Real	80.0%	2.21	60.0%	2.44	46.7%	2.50	46.7%	2.43	33.3%	2.90
Sim	100.0%	1.62	99.0%	1.97	100.0%	2.56	99.0%	2.84	98.0%	3.21
Sim2Real	90.0%	2.30	80.0%	4.54	73.3%	4.86	76.7%	6.09	63.3%	6.26
Sim2Real+Finetune	100.0%	1.90	96.7%	4.03	93.3%	4.14	100.0%	3.90	86.7%	3.54

The policy network is only trained on data collected in a 2.0 mm clearance scene.

Table IX lists the success rates and numbers of attempts. Training on a small real dataset results in an unsatisfactory performance. The success rate on 0.6 mm clearance is only 33.3%. The small amount of real data might lead to overfitting of the neural network. We find that the Sim policy achieves nearly 100% success rate on assembly clearances ranging from 0.6 mm to 2.0 mm, even though the training data is only collected in the scene of 2.0 mm clearance. This suggests that the policy network is well-trained in simulation and could generalize across different clearances.

Then, the Sim2Real policy achieves a success rate of 90.0% in a 2.0 mm clearance scene, indicating that the policy trained in simulation could be directly deployed in the real world. Furthermore, the Sim2Real policy could transfer to 0.6 mm clearance scene, achieving 63.3% success rate. These results demonstrate that the simulated tactile imprints in TacFlex are highly realistic and helpful for zero-shot Sim2Real transfer. Additionally, the Sim2Real+Finetune policy reaches 100.0% success rate on 2.0 mm clearance and still achieves 86.7% on 0.6 mm clearance, suggesting that fine-tuning the output layer of the policy model using a small amount of real data could further improve the Sim2Real performance.

Although real data is helpful in robot learning, a well-designed simulation environment is indeed significant to reduce the burden of data collection. The performance of this peg-in-hole experiment demonstrates that the proposed visuotactile simulation in this paper is competent in robot learning of contact-rich manipulation tasks.

VI. DISCUSSION

In this paper, the primary contribution of TacFlex lies in offering a suite of simulation methods for efficiently generating realistic multi-mode tactile imprints for various types of visuotactile sensors. The TacFlex is built upon FEM-based deformation simulation and can theoretically be integrated with any FEM simulator that supports volumetric mesh output. In this section, we will first discuss different FEM simulators within the TacFlex pipeline, as it has attracted much attention and forms the foundation for tactile imprints simulation. Subsequently, we will discuss the limitations of TacFlex and provide our outlooks on several promising applications.

Discussion on FEM simulators. In this paper, we connect TacFlex with NVIDIA Isaac Gym, SOFA, and polyFEM, and compare their simulation performances. Based on the authors' experience, NVIDIA Isaac Gym achieves an ideal trade-off between simulation accuracy and speed for the mentioned visuotactile sensors. Additionally, it is native for robot kinematics simulation and visual rendering, providing distinct advantages for robot learning in tactile perception and manipulation tasks. A concern is that the FEM in Isaac Gym does not support custom boundary conditions, which raises uncertainty in simulation accuracy when dealing with complex elastomer geometries. On the other hand, SOFA and polyFEM are more specialized in FEM calculations. The polyFEM could achieve accurate deformation simulation, but runs very slow. They might be beneficial for developing and validating tactile sensing algorithms, especially for sensors with complex deformations, as demonstrated in [14]. Users can choose an FEM simulator based on their specific requirements and connect it with TacFlex to achieve multi-mode tactile imprints simulation.

Limitations. Despite the promising performance of TacFlex, a few limitations remain to be addressed. As the designs of visuotactile sensors are too diverse in the community, some sensors are temporarily unsupported in TacFlex. 1) The visuotactile sensor that generates image texture by pins or particles with physical volumes, such as TacTip [43], is not supported by TacFlex pipeline. The texture mapping method in TacFlex only supports patterns pasted or painted on the sensor surface. 2) Rendering tactile images of the visuotactile and proximity sensor, such as StereoTac [44], presents challenges for TacFlex. The translucent coating layer makes the tactile image rendering process more complex. In addition, the image rectification method based on the refractive principal point used in TacFlex is not well-suited for visuotactile sensors with curved refracting interfaces. We plan to develop a more general approach in future work.

Outlooks. We provide our outlooks on several intriguing topics that are worth exploring using TacFlex simulation, which also constitute our future work. 1) The high-fidelity tactile images generated in simulation provide an opportunity for research on visuotactile sensing, such as dense depth reconstruction in [19] and [30], in which TacFlex could provide the ground truth with high quality. 2) A neural network architecture designed for visuotactile data (especially marker displacement fields) is desired for effective tactile representation. The TacFlex simulation offers an efficient approach for extensive data collection to support the development and training of such networks. 3) Integrating TacFlex into robot simulators has the potential to facilitate research on vision-tactile fusion methods and policy learning techniques for addressing more complex and dexterous robotic manipulation

IEEE Transactions on Robotics (T-RO) paper, presented at ICRA 2026, Vienna, Austria. Cite as T-RO paper.

tasks. Additionally, it may provide an opportunity for training of vision-tactile-language foundation models.

VII. Conclusion

In this paper, we introduce TacFlex, an efficient, flexible, and FEM-driven visuotactile simulation pipeline. TacFlex links the deformed elastomer mesh from the FEM simulator to multi-mode tactile imprints, such as tactile images with coating patterns, marker motion fields, and tactile 3D point clouds. We also propose a ray tracing-based rectification method that transforms virtual tactile images without refraction into those with refraction effects, thereby reducing the sim-real gap without requiring complex ray tracing rendering. After carefully calibrating the simulation parameters, extensive experiments demonstrate that TacFlex produces highly realistic tactile imprints simulation across various contact states. On the GelStereo 2.0 sensor, the MED error of simulated tactile 3D point clouds is 0.075 mm; the SSIM between simulated and real tactile images achieves 0.936; the marker pixel distance error is 1.102 pixels. The SSIM on GelStereo 2.0 sensor with checkerboard pattern and random color pattern is 0.924 and 0.865, respectively. Additionally, the TacFlex simulation performs well on diverse visuotactile sensors, such as the GelStereo Palm2.0 sensor with curved sensor surface and the GelSight Mini sensor with RGB lighting and marker pattern.

Furthermore, we train a cylindrical object pose estimation network and a peg-in-hole policy network within the TacFlex simulation. The pose estimation network could realize zero-shot transfer to real sensors. The position and rotation errors on the real GelStereo 2.0 sensor are 0.6 mm and 1.4 degrees, respectively. The peg-in-hole policy network, only trained in a 2.0 mm clearance scene in simulation, could be directly deployed in the real world, achieving a 90% success rate on 2.0 mm clearance and 63.3% on 0.6 mm clearance. We also find that the policy trained in simulation could achieve higher success rates of 100% and 86.7% when fine-tuned with a small amount of real data. These results demonstrate the effectiveness of Sim2Real framework based on TacFlex and highlight its potential in advancing robot tactile perception and manipulation.

In the future, we will continue to improve the proposed TacFlex pipeline from the following aspects. 1) TacFlex will incorporate more deformation simulation engines, including those based on MPM. 2) The pipeline will be extended to support more visuotactile sensors, such as the Soft-bubble tactile sensor. 3) We will further make efforts to improve the fidelity of simulated tactile imprints by exploring advanced rendering techniques and style transfer methods. We hope that TacFlex could advance the visuotactile research in the robot community.

References

[1] A. Billard and D. Kragic, "Trends and challenges in robot manipulation," *Science*, vol. 364, no. 6446, p. eaat8414, 2019.

[2] Q. Li, O. Kroemer, Z. Su, F. F. Veiga, M. Kaboli, and H. J. Ritter, "A review of tactile information: Perception and action through touch," *IEEE Transactions on Robotics*, vol. 36, no. 6, pp. 1619–1634, 2020.

[3] S. Li, Z. Wang, C. Wu, X. Li, S. Luo, B. Fang, F. Sun, X.-P. Zhang, and W. Ding, "When vision meets touch: A contemporary review for visuotactile sensors from the signal processing perspective," *arXiv preprint arXiv:2406.12226*, 2024.

[4] Y. She, S. Wang, S. Dong, N. Sunil, A. Rodriguez, and E. Adelson, "Cable manipulation with a tactile-reactive gripper," *The International Journal of Robotics Research*, vol. 40, no. 12-14, pp. 1385–1401, 2021.

[5] S. Kim and A. Rodriguez, "Active extrinsic contact sensing: Application to general peg-in-hole insertion," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 10 241–10 247.

[6] Y. Shirai, D. K. Jha, A. U. Raghunathan, and D. Hong, "Tactile tool manipulation," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 12 597–12 603.

[7] S. Dong, D. K. Jha, D. Romeres, S. Kim, D. Nikovski, and A. Rodriguez, "Tactile-rl for insertion: Generalization to objects of unknown geometry," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 6437–6443.

[8] C. Wang, S. Wang, B. Romero, F. Veiga, and E. Adelson, "Swingbot: Learning physical features from in-hand tactile exploration for dynamic swing-up manipulation," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 5633–5640.

[9] B. Tang, M. A. Lin, I. Akinola, A. Handa, G. S. Sukhatme, F. Ramos, D. Fox, and Y. Narang, "Industreal: Transferring contact-rich assembly tasks from simulation to reality," *arXiv preprint arXiv:2305.17110*, 2023.

[10] A. Church, J. Lloyd, N. F. Lepora *et al.*, "Tactile sim-to-real policy transfer via real-to-sim image translation," in *Conference on Robot Learning*. PMLR, 2022, pp. 1645–1654.

[11] Y. Lin, J. Lloyd, A. Church, and N. F. Lepora, "Tactile gym 2.0: Sim-to-real deep reinforcement learning for comparing low-cost high-resolution robot touch," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 10 754–10 761, 2022.

[12] S. Wang, M. Lambeta, P.-W. Chou, and R. Calandra, "Tacto: A fast, flexible, and open-source simulator for high-resolution vision-based tactile sensors," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 3930–3937, 2022.

[13] Z. Chen, S. Zhang, S. Luo, F. Sun, and B. Fang, "Tacchi: A pluggable and low computational cost elastomer deformation simulator for optical tactile sensors," *IEEE Robotics and Automation Letters*, vol. 8, no. 3, pp. 1239–1246, 2023.

[14] Q. K. Luu, N. H. Nguyen *et al.*, "Simulation, learning, and application of vision-based tactile sensing at large scale," *IEEE Transactions on Robotics*, 2023.

[15] W. Chen, J. Xu, F. Xiang, X. Yuan, H. Su, and R. Chen, "General-purpose sim2real protocol for learning contact-rich manipulation with marker-based visuotactile sensors," *IEEE Transactions on Robotics*, 2024.

[16] D. F. Gomes, P. Paoletti, and S. Luo, "Generation of gelsight tactile images for sim2real learning," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 4177–4184, 2021.

[17] A. Agarwal, T. Man, and W. Yuan, "Simulation of vision-based tactile sensors using physics based rendering," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 1–7.

[18] S. Dong, W. Yuan, and E. H. Adelson, "Improved gelsight tactile sensor for measuring geometry and slip. in 2017 ieee," in *RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 137–144.

[19] W. K. Do, B. Jurewicz, and M. Kennedy, "Densetact 2.0: Optical tactile sensor for shape and force reconstruction," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 12 549–12 555.

[20] W. Yuan, S. Dong, and E. H. Adelson, "Gelsight: High-resolution robot tactile sensors for estimating geometry and force," *Sensors*, vol. 17, no. 12, p. 2762, 2017.

[21] C. Zhang, S. Cui, S. Wang, J. Hu, Y. Cai, R. Wang, and Y. Wang, "Gelstereo 2.0: An improved gelstereo sensor with multimedium refractive stereo calibration," *IEEE Transactions on Industrial Electronics*, 2023.

[22] C. Zhang, S. Cui, S. Wang, J. Hu, Y. Huangfu, and B. Zhang, "High-precision 3d reconstruction study with emphasis on refractive calibration of gelstereo-type sensors," *Sensors*, vol. 23, no. 5, p. 2675, 2023.

[23] Y. Du, G. Zhang, Y. Zhang, and M. Y. Wang, "High-resolution 3-dimensional contact deformation tracking for fingervision sensor with dense random color pattern," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 2147–2154, 2021.

[24] Z. Si and W. Yuan, "Taxim: An example-based simulation model for gelsight tactile sensors," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 2361–2368, 2022.

[25] W. D. Kim, S. Yang, W. Kim, J.-J. Kim, C.-H. Kim, and J. Kim, "Marker-embedded tactile image generation via generative adversarial networks," *IEEE Robotics and Automation Letters*, 2023.

[26] Z. Si, G. Zhang, Q. Ben, B. Romero, Z. Xian, C. Liu, and C. Gan, "Difftactile: A physics-based differentiable tactile simulator for contact-rich robotic manipulation," *arXiv preprint arXiv:2403.08716*, 2024.

[27] W. Du, W. Xu, J. Ren, Z. Yu, and C. Lu, "Tacipc: Intersection- and inversion-free fem-based elastomer simulation for optical tactile sensors," *IEEE Robotics and Automation Letters*, 2024.

[28] G. Zhang, Y. Du, H. Yu, and M. Y. Wang, "Deltact: A vision-based tactile sensor using a dense color pattern," *IEEE Robotics and Automation Letters*, vol. 7, no. 4, pp. 10 778–10 785, 2022.

[29] S. Cui, R. Wang, J. Hu, J. Wei, S. Wang, and Z. Lou, "In-hand object localization using a novel high-resolution visuotactile sensor," *IEEE Transactions on Industrial Electronics*, vol. 69, no. 6, pp. 6015–6025, 2021.

[30] S. Cui, R. Wang, J. Hu, C. Zhang, L. Chen, and S. Wang, "Self-supervised contact geometry learning by gelstereo visuotactile sensing," *IEEE Transactions on Instrumentation and Measurement*, vol. 71, pp. 1–9, 2021.

[31] C. A. Felippa, "Introduction to finite element methods," *University of Colorado*, vol. 885, 2004.

[32] E. Coevoet, T. Morales-Bieze, F. Largilliere, Z. Zhang, M. Thieffry, M. Sanz-Lopez, B. Carrez, D. Marchal, O. Goury, J. Dequidt *et al.*, "Software toolkit for modeling, simulation, and control of soft robots," *Advanced Robotics*, vol. 31, no. 22, pp. 1208–1224, 2017.

[33] M. Liu and D. G. Gorman, "Formulation of rayleigh damping and its extensions," *Computers & structures*, vol. 57, no. 2, pp. 277–285, 1995.

[34] C. A. Felippa, "A systematic approach to the element-independent corotational dynamics of finite elements," 2000.

[35] R. W. Ogden, *Non-linear elastic deformations*. Courier Corporation, 1997.

[36] A. Agrawal, S. Ramalingam, Y. Taguchi, and V. Chari, "A theory of multi-layer flat refractive geometry," in *2012 IEEE conference on computer vision and pattern recognition*. IEEE, 2012, pp. 3346–3353.

[37] M. Li, L. Zhang, T. Li, and Y. Jiang, "Continuous marker patterns for representing contact information in vision-based tactile sensor: Principle, algorithm, and verification," *IEEE Transactions on Instrumentation and Measurement*, vol. 71, pp. 1–12, 2022.

[38] V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin, A. Allshire, A. Handa *et al.*, "Isaac gym: High performance gpu-based physics simulation for robot learning," *arXiv preprint arXiv:2108.10470*, 2021.

[39] T. Schneider, J. Dumas, X. Gao, M. Botsch, D. Panozzo, and D. Zorin, "Poly-spline finite-element method," *ACM Transactions on Graphics (TOG)*, vol. 38, no. 3, pp. 1–16, 2019.

[40] Q.-Y. Zhou, J. Park, and V. Koltun, "Open3D: A modern library for 3D data processing," *arXiv:1801.09847*, 2018.

[41] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[42] J. Xu, S. Kim, T. Chen, A. R. Garcia, P. Agrawal, W. Matusik, and S. Sueda, "Efficient tactile simulation with differentiability for robotic manipulation," in *Conference on Robot Learning*. PMLR, 2023, pp. 1488–1498.

[43] B. Ward-Cherrier, N. Pestell, L. Cramphorn, B. Winstone, M. E. Giannaccini, J. Rossiter, and N. F. Lepora, "The tactip family: Soft optical tactile sensors with 3d-printed biomimetic morphologies," *Soft robotics*, vol. 5, no. 2, pp. 216–227, 2018.

[44] E. Roberge, G. Fornes, and J.-P. Roberge, "Stereotac: A novel visuotactile sensor that combines tactile sensing with 3d vision," *IEEE Robotics and Automation Letters*, 2023.

Chaofan Zhang received the B.E. degree in automation from the Central South University in 2020. She is currently completing a Ph.D. degree in control theory and control engineering at the Institute of Automation, Chinese Academy of Sciences, focusing on visuo-tactile perception and robotic dexterous manipulation.

Shaowei Cui (Member, IEEE) received the B.E. degree in process equipment and control engineering from the South China University of Technology in 2017 and the Ph.D. degree in pattern recognition and intelligent system from the Institute of Automation, Chinese Academy of Sciences in 2022.

He is currently an Assistant Professor with the State Key Laboratory of Multimodal Artificial Intelligence Systems, Institute of Automation, Chinese Academy of Sciences. His research interests include visuo-tactile perception and robot manipulation.

Jingyi Hu received the B.E. degree in automation from the Beijing University of Posts and Telecommunications in 2019. She is currently completing a Ph.D. degree in pattern recognition and intelligent system at the Institute of Automation, Chinese Academy of Sciences, focusing on vision-based tactile sensing and robotic dexterous manipulation.

Tianyu Jiang received the B.E. degree in automation from the University of Science and Technology Beijing, Beijing, China, in 2023. He is currently working toward the Ph.D. degree in control theory and control engineering with the Institute of Automation, Chinese Academy of Sciences, Beijing, China. His current research interests include planning and control of intelligent vehicles.

Tiandong Zhang received the B.E. degree in electronic science and technology from the Beijing University of Posts and Telecommunications in 2018 and the Ph.D. degree in control theory and control engineering from the Institute of Automation, Chinese Academy of Sciences in 2023.

He is currently an Assistant Professor in the State Key Laboratory of Multimodal Artificial Intelligence Systems, Institute of Automation, Chinese Academy of Sciences. His research interests include intelligent control and biomimetic robots.

Rui Wang received the B.E. degree in automation from the Beijing Institute of Technology in 2013 and the Ph.D. degree in control theory and control engineering from the Institute of Automation, Chinese Academy of Sciences in 2018.

He is currently an Associate Professor with the State Key Laboratory of Multimodal Artificial Intelligence Systems, Institute of Automation, Chinese Academy of Sciences. His current research interests include robot perception, control, and learning.

Shuo Wang (Member, IEEE) received the B.E. degree in electrical engineering from the Shenyang Architecture and Civil Engineering Institute, the M.E. degree in industrial automation from the Northeastern University, and the Ph.D. degree in control theory and control engineering from the Institute of Automation, Chinese Academy of Sciences in 1995, 1998, and 2001, respectively.

He is currently a Professor with the State Key Laboratory of Multimodal Artificial Intelligence Systems, Institute of Automation, Chinese Academy of Sciences, with the School of Artificial Intelligence, University of Chinese Academy of Sciences, and with the Center for Excellence in Brain Science and Intelligence Technology, Chinese Academy of Sciences. His research interests include biomimetic robot, robot skill learning, and multi-robot systems.