# A Modularized Approach to Vision-based Tactile Sensor Design Using Physics-based Rendering

Arpit Agarwal

CMU-RI-TR-24-37
July, 2024

School of Computer Science
The Robotics Institute
Carnegie Mellon University
Pittsburgh, PA

**Thesis Committee:**

| | |
|---|---|
| Dr. Wenzhen Yuan (Co-Chair) | University of Illinois Urbana-Champaign |
| Dr. Ioannis Gkioulekas (Co-Chair) | |
| Dr. Nancy Pollard | |
| Dr. Edward Adelson | Massachusetts Institute of Technology |

*Submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Robotics.*

# Abstract

Touch is an essential sensing modality for making autonomous robots more dexterous and allowing them to work collaboratively with humans. In particular, the advent of vision-based tactile sensors has resulted in efforts to design such sensors that can be easily incorporated into various robot structures for different robotic manipulation tasks, to increase robustness, precision, and reliability. However, this design task remains a challenging problem. This is for two reasons: first, the design of the sensor itself requires the compact integration of multiple optical elements to improve optical signal fidelity during interaction with the environment; second, the successful integration of vision-based tactile sensors into robotic manipulation tasks requires the codesign of both the sensors and the robot structure itself.

This thesis aims to alleviate these two challenges by creating a general design framework that allows a roboticist to quickly iterate on the design and evaluation of vision-based tactile sensors for designated robotic manipulation tasks. The framework comprises three core elements.

First, our framework uses an optical simulator that can accurately and efficiently generate the images captured by arbitrary sensor designs. Our simulator leverages physics-based rendering techniques from computer graphics and enables the generation of realistic tactile images for any given sensor design. To create this simulator, we performed detailed real-to-sim experiments to calibrate our simulation models. We show that the resulting simulator can qualitatively and quantitatively match real-world measurements for GelSight-like sensors with flat and curved sensing surfaces.

Second, our framework proposes computational techniques for procedural sensor generation and automatic sensor design evaluation techniques. In the context of curved tactile sensors, our generator takes as input a 2D curve and uses CAD primitives to generate from it the full sensor shape. The procedural sensor generation allows for the automatic placement of different optical components, given their corresponding reference geometry. We introduce three objective functions: *RGB2Norm*, *NormDiff*, and *As-orthographic-as-possible*. These objective functions quantify sensor design's tactile signal perception and enable automatic parameter selection.

Third, our framework introduces an interactive design toolbox, *OptiSense Studio*, that functionalizes our design pipeline into a useful tool for novice users. We introduce general design modules for the rapid prototyping of GelSight-like tactile sensors. The toolbox allows interactive feedback through optical simulation while designing the sensor without deep expertise. The obtained design is automatically parameterized through our toolbox and can be optimized using our proposed objective functions.

We have successfully applied this framework for the design of vision-based tactile sensors that used curved surfaces to emulate human fingertips. We have also applied our interactive framework for rapidly creating optimized variants of existing tactile sensors, GelSight Mini, GelSight360, and GelSight Svelte. Finally, we are able to create a new sensor, GelBelt, for a different robotic application completely virtually and optimize its illumination settings using our toolbox.

Through this thesis, we demonstrate the utility of our design framework for the co-design of vision-based tactile sensors and soft robots. More broadly, we hope to create a new point of convergence between disparate communities such as computer graphics (physics-based rendering and simulation), optics (optical lens and material design), and robotics, and foster new research directions within and across these communities.

# Acknowledgments

I would like to thank my advisor, Wenzhen Yuan, for her many teachings on conducting research: picking challenging problems and presenting ideas. Wenzhen always ensured that I had the right means that would help me grow into a successful researcher. Most of all, I thank her for encouraging me to dare and persevere in tackling big challenges. This was possible because she supported me and made it safe for me to try.

I have been fortunate to have Ioannis Gkioulekas act as my second advisor during large part of my doctoral studies. He introduced me to new research communities, computational photography and computer graphics. I learned a great deal from his courses on computational photography and physics-based rendering. I would highly encourage everyone to take these courses while at CMU. His courses and research inspired me to the first part of my thesis. I also had the privilege to collaborate with Edward Adelson during my studies. His invaluable insights and encouragement were vital to my success. I am always inspired and amazed by his unique ideas. I am grateful to Nancy Pollard for her teaching and mentoring. It was her class on "Design and Control for Dexterous Manipulation" that inspired me to learn and work on design problems.

This work is the product of engaging and fruitful collaborations with Sandra Liu, Yuxiang Ma, Mohammad Amin Mirzaee, Timothy Man, and Achu Wilson. This dissertation would not have been possible without them. I would like to especially thank Timothy and Achu for accommodating my rough manufacturing ideas and making them concrete actionable items. I would like to thank members of the Perceptual Science Group at MIT CSAIL, especially Jialiang Zhao, Megha Tippur, Branden Romero, and Shaoxiong Wang for the fruitful and inspiring discussions. I would especially like to thank Jialiang, Megha, and Branden for sharing the sensor designs from their amazing works, without which this thesis would have been impossible. I want to acknowledge my fellow members of RoboTouch group, for the amazing work and social environment they created: Zilin Si, Uksang Yoo, Akhil Padmanabha, Ruihan Gao, Helen Jiang, Xioafeng Guo, Yufan Zhang, Ruohan Zhang, and Dakarai Crowder. Special thanks Brian Hutchison, for managing all of my administrative requests. I want to acknowledge the CMU communities that I was part of: Tech4Society, and Social Wellness committee. These organizations provided me with new friends and joy in life during

difficult times.

Finally, I am deeply obliged to my family and friends; for their willingness to provide me with the means necessary to pursue my doctoral studies; for their unconditional care for my well-being. The dedication and fight that my parents have shown has always inspired me. I would especially like to thank my long-time roommate and friend Ashwin Khadke for putting up with me. His positive attitude towards life and willingness to participate in my random discussions have made this journey very enjoyable. I also would like to thank my friends Muhammad Suhail, Anahita Mohseni Kabir, Kartik Iyer, Fengying Yang, and Arkadeep Narayan Chaudhury for involving me in their celebrations. Finally, I thank my sister Palak Agarwal, for supporting the whole family during my studies, especially during the Covid pandemic. Her zeal and fun nature always inspire me to see the bigger picture and feel happy.

# Contents

*When this dissertation is viewed as a PDF, the page header is a link to this Table of Contents.*

# List of Figures

# List of Tables

# Chapter 1

# Introduction

## 1.1 Motivation

Currently, robots have limited ability to perform dexterous manipulation and work collaboratively with humans. Tactile sensing provides critical information, such as perception of object properties and rich contact information for robotic manipulation. Due to this, we have seen the development of tactile sensors with various transduction methods [35, 70, 108, 111], sensor shapes [19, 73, 101, 107], and output modalities [23, 50, 104]. Vision-based tactile sensors (VBTS) have specifically become popular in the robotics community because of their high resolution, low cost, and ability to measure multiple contact information. GelSight [107] is a specific VBTS that uses color information to encode tactile signals. Figure 1.1 shows the various contact information that is obtained using GelSight-like tactile sensors.

Due to high-resolution sensing, VBTS is also useful for perception tasks with fine features, such as detecting skin diseases and detecting fine defects on aerospace parts. Figure 1.2 shows the perception of skin dryness using tactile images. The visual pattern allows quantification of the dryness of the skin and therefore can be used to automatically monitor eczema, which is a chronic health condition. The perception of depth continuities on metallic surfaces is another big challenge in aerospace parts. The commonly used vision-based methods for defect detection are not robust for metallic parts [5]. As shown in Figure 1.3 the inspection of aircraft requires perception of a large surface with orders of magnitude small

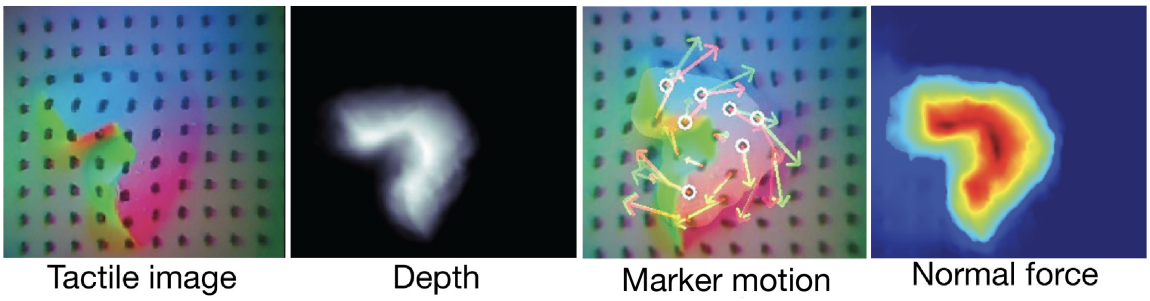



Figure 1.1: **Vision-based tactile sensing modalities**: As shown in [91], GelSight-like tactile sensors measure high-resolution tactile images that can, then be post-processed to output rich contact information - surface depth, marker motion and normal force.

defects with depth of $50\,\mu\mathrm{m}$. The use of GelSight can enable quantification and automatic detection of these defects.

Unlike other sensing modalities, such as cameras, tactile sensors offer unique challenges as they have to be integrated into the robot morphology for perception. The design of vision-based tactile sensors is a challenging problem, as it requires the complex interaction of various optical elements (lights, material, and camera) to obtain contact information. There has been tremendous progress in the design of VBTS hardware, especially GelSight-like tactile sensors, in the last decade. However, to build increasingly complex, functional and integrated robotic structures with tactile sensing, we also need to create *design tools*, that allow users to efficiently explore a design space that incorporates shape, material, and illumination. Furthermore, to empower more people to create sensors for their own personalized use cases, we also need to develop tools that are more accessible to users with little design experience.

The central question driving this thesis is:
*Can we design GelSight-like tactile sensors for arbitrary use cases in simulation and optimize various optical parameters?*

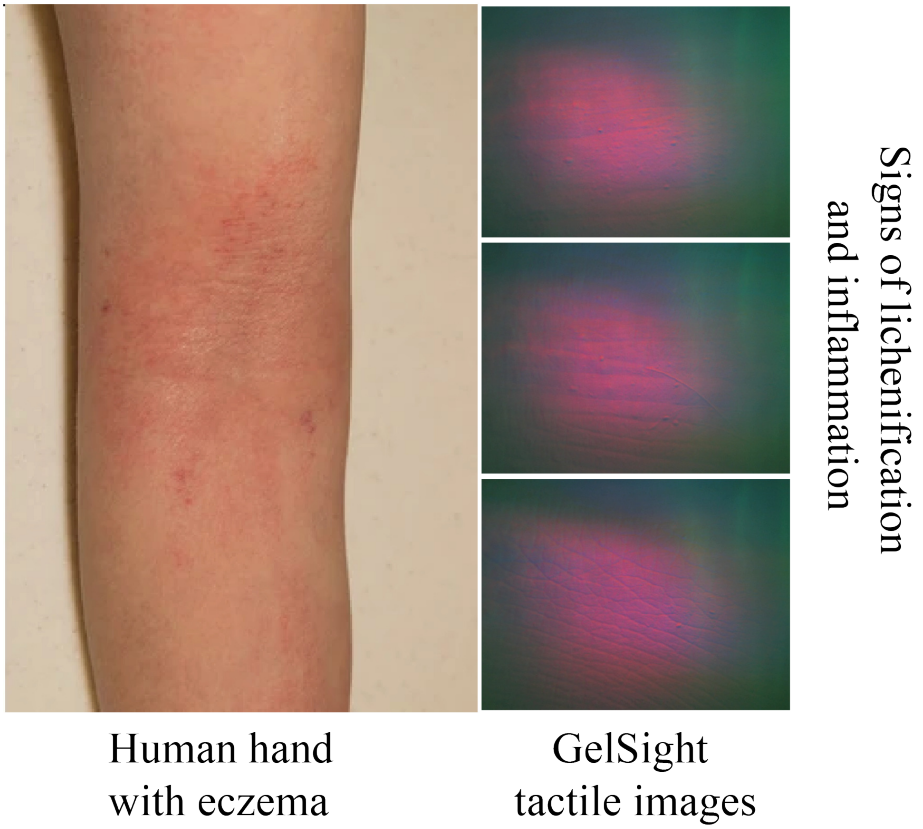



Figure 1.2: **GelSight applications in biomedical sensing**: The left half shows a human hand with various dry skin patches due to eczema. The skin pattern changes due to increased dryness level. This change in pattern can be clearly seen and quantified in GelSight tactile images shown on the right.

## 1.2 Approach and Contributions

The goal of this thesis is to democratize the process of GelSight-like tactile sensor design and make tactile sensors ubiquitous in robotics. We specifically focus on design iteration using simulation and computational techniques. The methods developed in this thesis allow for the automatic evaluation of a novel sensor design. We aim to make the design process for new end-effectors or robots with integrated vision-based tactile sensing semiautomatic and reduce the time taken to optimize the sensor parameters.

In this thesis, we introduce a computational GelSight-like vision-based tactile

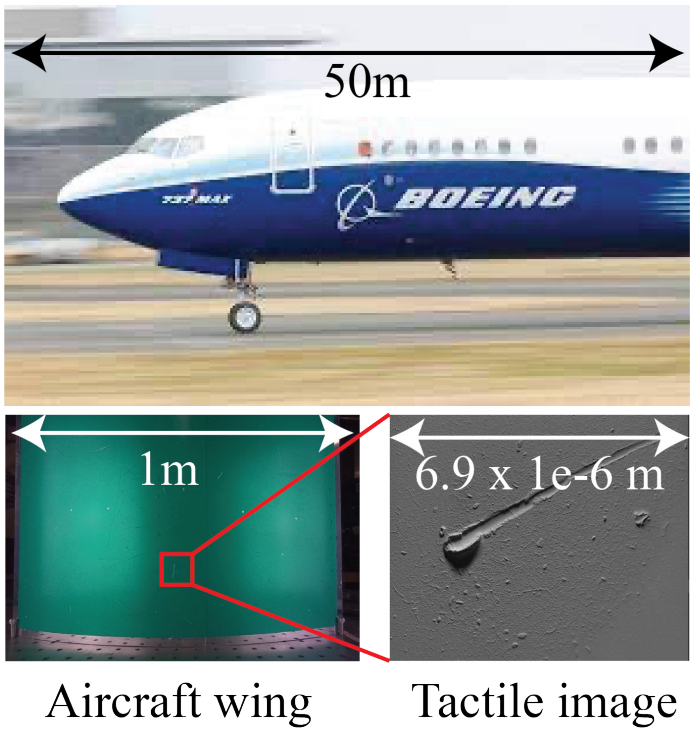



Figure 1.3: **Aerospace defect detection**: An aircraft ($50\,\mathrm{m}$ long ) requires inspection of surfaces after each flight. It may contain defects that are orders of magnitude ($1 \times 10^{-7}$) smaller than the aircraft. The bottom left shows a visual image containing multiple defects that are almost invisible in an RGB image. However, the defect is detectable in the GelSight tactile image on the bottom right.

sensor design framework with a physically accurate light transport simulation. Our framework consists of three main parts—optical simulation, computational techniques including sensor evaluation objective functions, and an interactive design toolbox. Our contributions are as follows:

**Optical tactile simulation.** We develop an optical tactile simulation framework using physically based rendering (PBR) techniques for accurately simulating vision-based tactile sensors. We perform real-to-sim experiments to calibrate key simulation models for accurately generating tactile images of novel tactile sensor designs. Our study shows that our framework can generate simulated tactile images that match the tactile images produced by a range of vision-based tactile sensors, for example, GelSight with an almost flat sensing surface; GelSight Mini with refractive

layers; a human-like fingertip curved tactile sensor; GelSight FinRay which uses fluorescent lighting and mirrors; as shown in Figure 2.1. (Chapter 3, Agarwal et al. [3, 4])

**Computational design framework.** We develop a design framework for GelSight-like vision-based tactile sensors, with two key novel elements—a low-dimensional sensor shape parameterization, and a design evaluation procedure. Our low-dimensional parameterization allows the procedural generation of varied sensor shapes and makes the design optimization tractable. We systematically characterize the effect of various design spaces using our design objective function, *RGB2Normal*, to identify the best designs. Our computational framework enables, for the first time, the design of a curved vision-based tactile sensor completely in simulation and optimization of shape parameters for better 3D reconstruction than human-expert design. (Chapter 4, Agarwal et al. [3])

**Interactive design toolbox.** We introduce a modular design framework for novice users to democratize the sensor design and provide an interactive toolbox, *OptiSense Studio* that functionalizes the techniques for rapid design iteration. We also introduce two new objective functions, *NormDiff* and *As-orthographic-as-possible* (AOAP), that optimize for 3D reconstruction and reduce sensing area distortion. We investigate faster but approximate rendering algorithms for interactive optical simulation. The interactive design interface with simulation feedback enables novice users to modify existing sensors and mechanical engineers to design new VBTS sensor form factors completely in the virtual domain. (Chapter 6, Agarwal et al. [2])

### 1.2.1 Thesis outline

In Chapter 3, we introduce the optical simulation framework that allows the generation of physically accurate tactile images. In Section 3.2 and Section 3.3 we describe tactile sensors and their real-to-sim experiments to calibrate the simulation models. We further evaluate the accuracy of the simulation results by comparing the simulated tactile images with the tactile images captured from real-world sensor

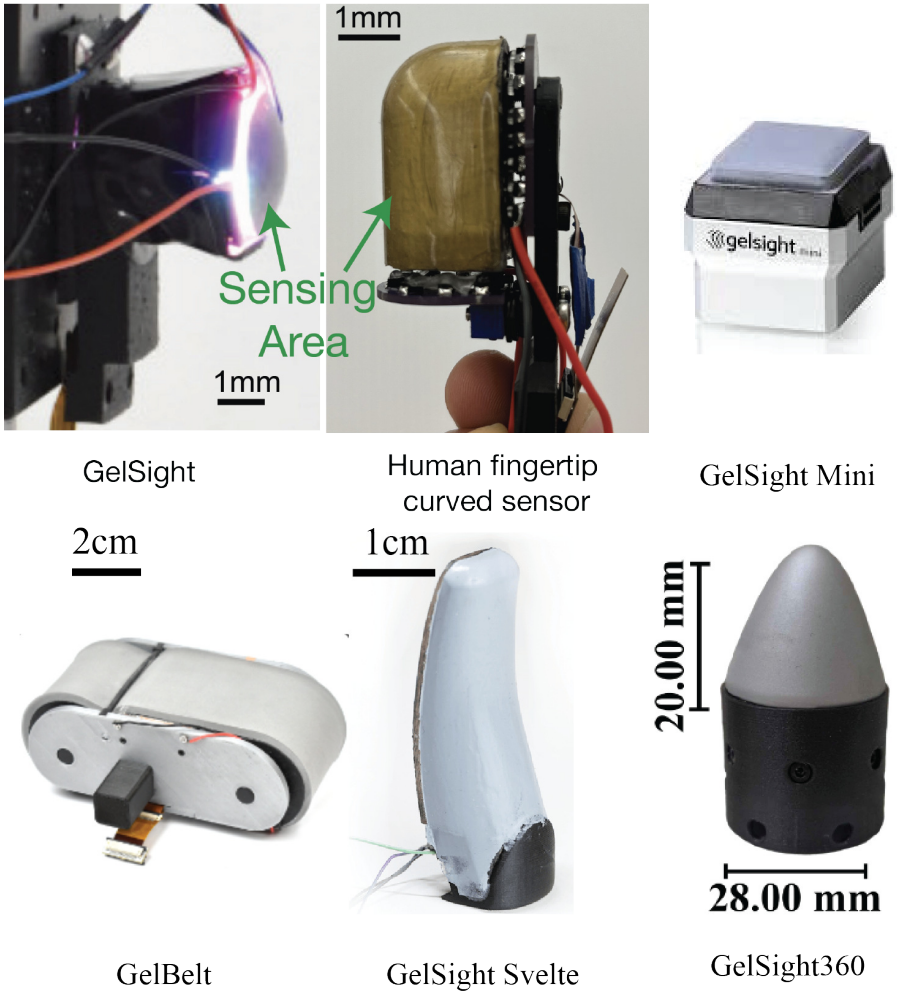



Figure 1.4: Tactile sensors discussed in this thesis are GelSight hexagon sensor, human fingertip-like sensor, commercial GelSight Mini, new GelBelt, GelSight Svelte [115], and GelSight360 [94]. The sensors considered are useful for robotic manipulation and contact perception.

prototypes.

In Chapter 4, we introduce our *sensor design framework* and apply it to design a curved vision-based tactile sensor. We further show the performance of the optimized sensor design in simulation and by manufacturing a real-world prototype. Our optimized sensor design performs approximately 5x better at robotic surface inspection compared to a human expert design.

In Chapter 6, we introduce our modular and interactive design pipeline. We introduce various sensor modeling modules and parameterization of optical components to enable optimization. We also propose two new objective functions: *NormDiff* and *As-orthographic-as-possible* for design improvement. Using the proposed framework, we show multiple case studies to optimize the shape of optical components, material properties, and light type. We also design a new tactile sensor, GelBelt, from concept to fully optimized design.

# Chapter 2

# Background and Related Work

## 2.1 Vision-based tactile sensor designs

In this section, we discuss various vision-based tactile sensor designs and probable reasoning behind those choices to highlight the trade-off between perception, function, and compactness.

Yuan et al. [107] choose the elastomer material based on transparency and mechanical considerations. The sensing surface coating layer is chosen to have matte or semi-specular reflectance. Johnson et al. [44] placed six LED light sources along the periphery of the elastomeric surface. Johnson and Adelson [43] used three colored LED arrays at an elevation angle of 30 degrees for best illumination. Li et al. [51] used four colored LED arrays and used acrylic guiding plates to illuminate the sensing surface. Dong et al. [21] proposed a new design with three colored LED arrays with collimating lens and are placed at an angle of $71^\circ$ with respect to the sensing surface. They added a translucent surface in front of the LEDs to make the outgoing illumination more diffuse. Donlon et al. [23] used an ideal mirror to indirectly view the sensing surface using the camera. They derived a trigonometric relationship between the mirror angle and the sensing surface coverage. They designed the acrylic guiding plate surface to be a parabolic to allow light paths from LEDs placed near camera to be directed towards the sensing surface. Note that all the above approaches were based on human intuition and trial-and-error for generating new designs and estimating their parameters.

We review the main GelSight-like tactile designs in Table 2.1. The designs are characterized on the basis of illumination design, camera viewport, and application features. The key illumination features are *light piping*, coating material is semi-specular or diffuse, and the number of light groups (collection of light of the same color). The key camera viewport features are if the design uses mirrors to change the positioning of the camera and the number of cameras used to cover the sensing surface. The key features that are added to aid target application (dexterous manipulation) are if the sensing surface is curved and contains markers. We consider sensors with focus on parallel-jaw grippers, omnidirectional sensing, functional moving parts, highly compliant sensing surface and full human-like finger sensor shapes.

Table 2.1: **Review of GelSight-like tactile sensor**: A comparison between state-of-the-art GelSight-like tactile sensors.

| Name | Illumination | | | Viewport | | Application features |
|---|---|---|---|---|---|---|
| | Light piping | Semi-specular | Num. light groups | Mirrors | Num. cameras | Curved |
| GelSight Hexagon [107] | no | no | 6 | no | 1 | no |
| GelSight RoundTip [77] | yes | yes | 3 | no | 1 | yes |
| GelSight Mini | yes | no | 3 | no | 1 | no |
| DIGIT [47] | no | no | 3 | no | 1 | no |
| GelSlim 1.0 [23] | yes | no | 1 | yes | 1 | no |
| GelSlim 3.0 [90] | yes | no | 3 | no | 1 | no |
| **Omnidirectional** | | | | | | |
| GelSight360 [94] | yes | yes | 3 | no | 1 | yes |
| RainbowSight [95] | yes | yes | — | no | 1 | yes |
| DenseTact 2.0 [20] | yes | yes | 3 | no | 1 | yes |
| Omnitact [73] | no | no | 11 | no | 5 | yes |
| GelTip [27] | yes | yes | 3 | no | 1 | yes |
| MinSight [8] | yes | no | 6 | no | 1 | yes |
| **Functional Designs** | | | | | | |
| RoTip [42] | yes | no | 3 | no | 1 | yes |
| Roller Grasper [105] | no | no | 3 | yes | 1 | yes |
| GelLink [65] | yes | no | 6 | yes | 1 | no |
| **Highly compliant** | | | | | | |
| GelSight Fin Ray [55] | no | yes | 3 | no | 1 | no |
| GelSight Baby Fin Ray [58] | no | no | 3 | yes | 1 | no |
| **Fingers** | | | | | | |
| Exoskeleton Soft Finger [82] | no | yes | 1 | no | 1 | yes |
| GelSight EndoFlex [59] | no | no | 3 | no | 2 | yes |
| GelFinger [53] | yes | yes | 6 | no | 1 | yes |
| GelSight Svelte [115] | no | no | 2 | yes | 1 | yes |

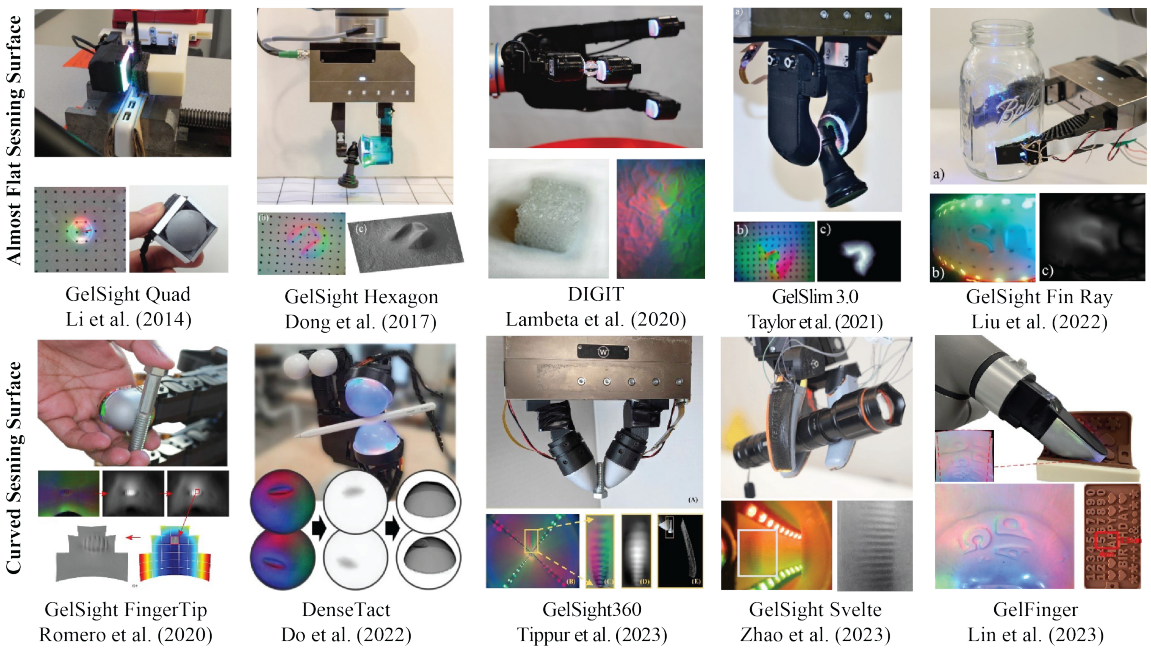



Figure 2.1: **Various types of GelSight sensor designs** (Pictures adapted from [19, 22, 47, 51, 53, 55, 77, 91, 94, 115]): Researchers have designed GelSight sensors with varied optical systems to fit robot fingers with either flat or curved surfaces. The performance of the sensors also varies and is affected by a number of the optical system parameters. Our work aims to bridge the knowledge gap between expert sensor designers and novice users, thereby simplifying and expediting the sensor design process.

## 2.2 Contact simulation for tactile sensors

Previous approaches for modeling and simulating contact between the sensor surface and the object can be categorized into a) modeling the deformation of the tactile sensor surface and b) modeling low-dimensional features used in a particular sensing technology. [71] simulated BioTac [26], which is a finger-shaped sensor with a fluid-coupled electrode array and measures impedances. In their simulation, BioTac is modeled using the Finite Element Method (FEM), which outputs the quasistatic nodal displacement under applied force over a known contact area. Vision-based tactile sensors such as TacTip [101] and [80], which track the motion of dots, either on the sensor surface or embedded in fluid, have

also been simulated. [18] modeled TacTip as a set of *pin* positions(similar to those found in the real sensor) which deform using an elastic push-pull force based on contact with objects in the scene. [81] simulated their sensor [80] using the neural network, trained using a data set generated by Finite Element(FEM) simulation. Their simulation is able to predict the position and force distribution. They decouple camera parameters from Neural Network training to allow adaptation to various cameras. This system could be used to simulate surface deformations and low-dimensional tactile features. However, the above methods do not work well for GelSight because of the complex light system. [112] simulated the tactile sensor by finding the intersection surface between the object mesh and the robot hand; and sampled points along that surface to simulate the tactile sensation for force closure.

*Tactile optical simulation*. There have been some recent works that simulate the image formation process for GelSight-like vision-based tactile sensors. [88] and [28] use directional lights, with phong material and diffuse material respectively, to simulate images formed by the tactile sensors. The assumption of directional light breaks down if the physical lights are very close to the scene(sensor surface in this case)[54]. Our work uses a general image formation process that takes multiple bounces of light into account, together with a physically accurate light model and material surface. This allows us to capture the spatial variation in color and intensity distribution in the simulated image.

## 2.3 Virtual robotic design and Sim2Real

In this section, we cover related work that leverages simulation for sensor design and other Sim2Real works that focus on robotic manipulation.

[91] is the most similar to our work. In this paper, the authors redesigned GelSlim [23] to optimally recover the surface geometry, similar to GelSight. They used raytracing simulation software to design a shaping lens and LED light position. Their simulation-driven approach was useful for coming up with a nontrivial lens shape. Although the authors used simulation, the output modality (radiant flux) was different than the tactile image (camera image). The raytracing software used by the authors is more focused on professional optical designers and requires detailed models of optical components, which can be overwhelming for roboticists.

Moreover, their design tool did not provide any guidance on how to generate tactile sensors or provide any objectives for automatic parameter selection. Therefore, it is unclear whether their approach can be extended to an end-to-end approach for tactile sensor design.

In [113], the authors analyze the common design pipelines of camera-based sensors and propose a dictionary-based process flow design approach. Their approach is useful for mixing and matching various workflows for sensor manufacturing. Although useful, their work does not provide any feedback on the validity or sensing ability of the design. Our work is the first to provide simulation-driven interactive feedback on the validity and perception capabilities of sensor design. Moreover, their work does not consider any optimization-based parameter selection techniques.

In [87], authors use efficient tactile simulation to train a grasp stability model completely in simulation and show zero-shot transfer to real robots. This approach depends on Taxim [86] which requires data from the real-sensor prototype for simulation. Since our focus is on creating a new sensor completely in simulation, this simulation approach is not applicable to our problem.

## 2.4 Robot design optimization

In this section, we cover related work that leverages digital design to optimize the design of complex robotic structures. This topic is broad and we cover only a few papers that served as inspiration or guidance for our work.

In [31], the authors introduced a design pipeline for truss-based structures, best known for structural stability and shape complexity. Their pipeline allows the creation of complex shape-changing truss structures with reconfigurable constraints. They also created a design tool that provides interactive preview and truss design modules and output control code. Their human study found that their tool "empowers users to design and build truss structures with a wide range of shapes and various functional motions. In [79], the authors introduced a system that allows the interactive exploration and optimization of parametric CAD data. They used precomputation and a new interpolation scheme on the CAD parameters. In [118], authors proposed an integrated design pipeline for robotic gripper generation with

integrated tactile sensing. They used knitted tactile sensors to put on the designed robotic grippers using grammar rules. It is unclear how to extend their work to incorporate the GelSight sensor family. Also, their work can only be tested after manufacturing the generated prototype, as they do not have any simulation step or optimization to select parameters.

# Chapter 3

# Optical Simulation Framework

Simulation is a critical tool in the development of robotic systems. It is widely used for hardware design, control, and planning. Simulations are useful not only at the start of the development process but also for debugging and rapid iteration of the design when a new design objective emerges. Due to the above advantages, we have seen the development of a number of rigid body simulators like ODE [89], SimBody [85], MuJoCo [96], Dart [48] and, particle-based simulators like Nvidia FleX [67] and SOFA[7]. Tactile sensing is a cornerstone for complex robotic manipulation together with advanced control algorithms and hardware design. However, most modern simulators have limited tactile sensing simulation.

In this thesis, we are specifically interested in simulating vision-based tactile sensors due to their high resolution. Vision-based tactile sensor simulation has two major components, namely *optical simulation* and *dynamics simulation*. In this work, we focus on an optical simulation system using physics-based rendering (PBR) [76] techniques. PBR focuses on accurately modeling the physics of light scattering. PBR allows generating physically accurate images after specifying the physical location of optical elements like cameras and lights; the deformable surface geometry; the material properties of the sensor surface. Since our system is based on accurate simulation of light, given the sensor setup, it can be used as a tool for generating accurate tactile images for novel sensor designs, without the need for ever manufacturing that design.

## 3.1 Optical simulation framework

Simulation algorithms and simulation models of various tactile sensor components are the two key ingredients to generate realistic tactile images using PBR techniques. In this section, we first motivate the need to do PBR, followed by the basics of PBR, and then talk about the specific method that we use for simulation in our work. We introduce the real-to-sim techniques for calibrating simulation models for individual tactile sensors in Section 3.2 and Section 3.3.

### 3.1.1 Simulation challenges

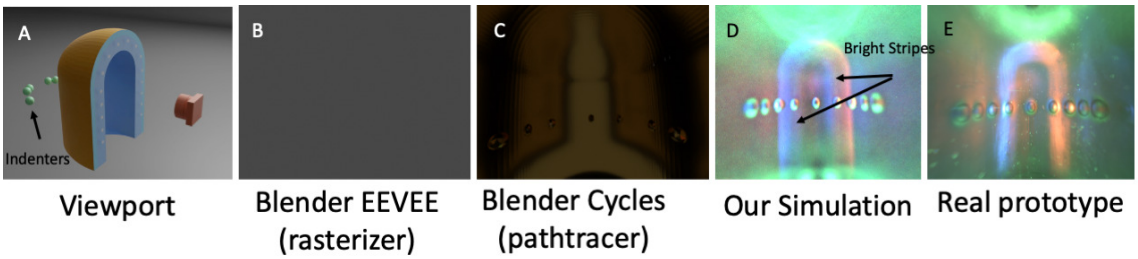



Figure 3.1: **Full vision-based tactile sensor comparison:** (**A**) Simulation scene: the camera, vision-based tactile sensor, and indenter, which we used to analyze our designs. (**B**) and (**C**) show the images simulated using Blender EEVEE and Blender Cycles renderer respectively. (**D**) HDR image simulated by our framework using calibrated simulation models. Our simulated results are a close match to the physical prototype as we are able to reproduce a)Bright light stripes due to focused LEDs b) Light piping of red color from the right to illuminate the spheres on the left and similarly for blue color. (E) HDR image captured with our real-world tactile sensor prototype, when the sensor is indented with a set of spheres.

The key motivation of our work is to do tactile sensor design by exploring design spaces. Traditional simulation techniques such as rasterization (Blender EEVEE) fail to generate any image (Figure 3.1B) and unidirectional path tracing (Blender Cycles) ( Figure 3.1C) fail to match the real prototype image. Therefore, we build a simulator of light that accounts for realistic light models, complex material properties such as Bidirectional Scattering Distribution Function (BSDF), and multiple bounces of light by using physics-based rendering techniques (PBRT) [76]. PBRT

allows us to generate unbiased RGB images efficiently.

Figure 3.16B shows an example path diagram of light as it travels from light source to camera, in our sensor setup. Our sensor setup has 3 key challenges

A useful light path is generated after refraction through multiple rough surfaces and reflecting of a highly glossy sensing surface to give information about the outermost sensing surface. This length of such successful light path is greater than 5. Light transport that requires multiple bounces on refractive and highly glossy (specular) surfaces is known to be a challenging problem in computer graphics [39, 109]. In our tactile sensor model, we have 3 key surfaces - 2 of which are refractive and the outermost surface can be highly specular (experimentally found to work better).

Another key challenge in our tactile sensor is that it is composed of curved surfaces, as it is supposedly human fingertip-like. Having curved surfaces is beneficial for various robotic applications[77]. However, this poses a challenge for sampling paths in the rendering algorithm[62].

A key component of various rendering(simulation) algorithms is a technique called *"Next Event Estimation"*(NEE)[97]. This technique tries to find the light received by each intermediate point, in the full light path, directly from the light source. However, the effect of this technique in our sensor setting is limited due to the light source not being directly visible through any point on the sensing surface.

## 3.2 Optical simulation for GelSight sensor

In this section, we use the optical simulation framework for simulation vision-based tactile sensors like Gelsight [106], which have almost flat sensing surfaces. We give a brief introduction of the sensor, real2sim experiments to calibrate simulation models, and propose a cheap tactile sensor deformation simulation based on surface convolution for almost flat sensing surface tactile sensors. We then evaluate our simulation framework by comparing the tactile images generated by our simulation framework with those of real sensor prototypes.

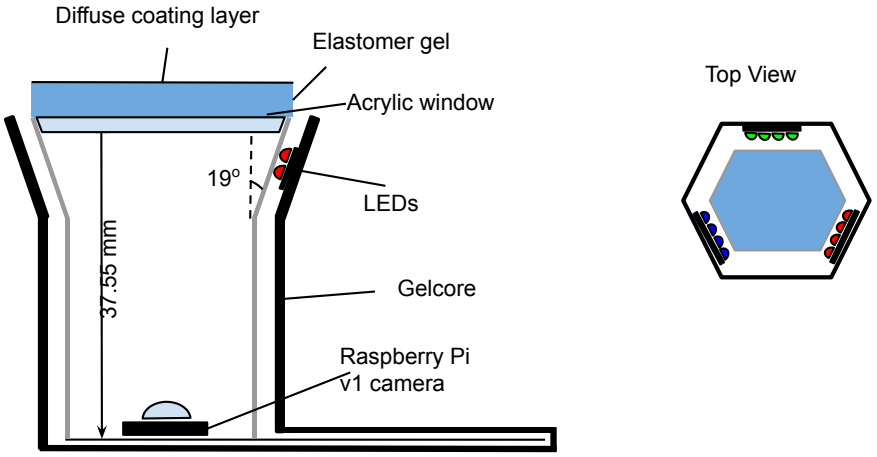



Figure 3.2: **GelSight sensor illustration**: The key components which we model in our work are gelcore, elastomer surface and LEDs.

### 3.2.1 GelSight description

These tactile sensors have multiple colored lights, soft deformable skin, and an RGB camera. When an object interacts with the soft sensor skin, the deformed skin shape interacts with light to form an image in the camera. The sensor uses photometric stereo [9] to invert RGB color information to shape information. We can obtain high-resolution shapes, multiaxis force, and friction information using Gelsight. Figure 3.2 shows the illustration of the GelSight prototype used in our study. The prototype is based on the sensor proposed in [106].

### 3.2.2 Real2sim simulation model calibration

This section describes the specific models of light, the material of the translucent supporting structure (we denote it as *gelcore*), and the elastomer used to create the GelSight sensor in simulation.

#### Light models

We use *AreaLight* model for our simulation system. The *AreaLight* is a good approximation of diffuse illumination received on the deformable surface of our sensor. This light model is fast to simulate and is a common choice for simulating natural

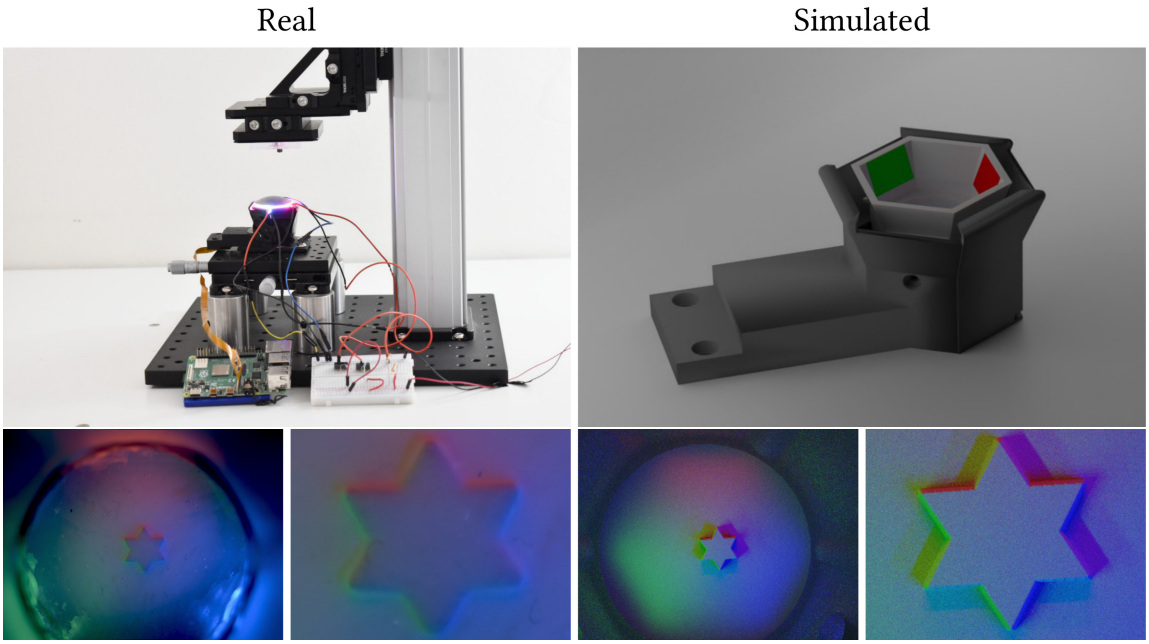



Figure 3.3: The comparison of a real GelSight sensor and a simulated GelSight sensor, when a star-shaped object is contacting the sensor. Our model well simulated the optical system in the sensor and therefore can generate a realistic tactile image that indicates the object's shape.

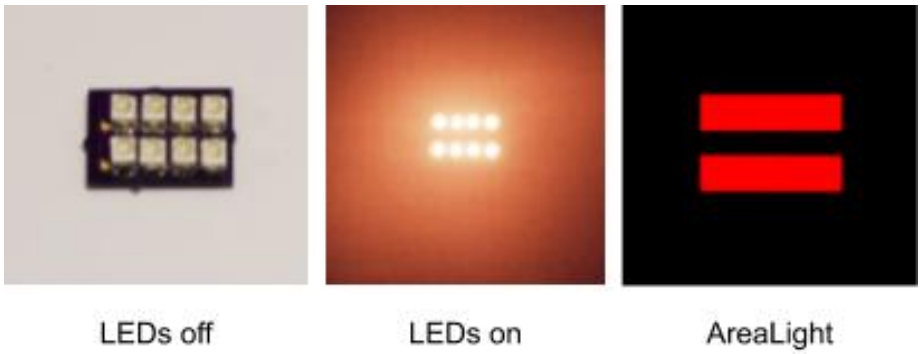
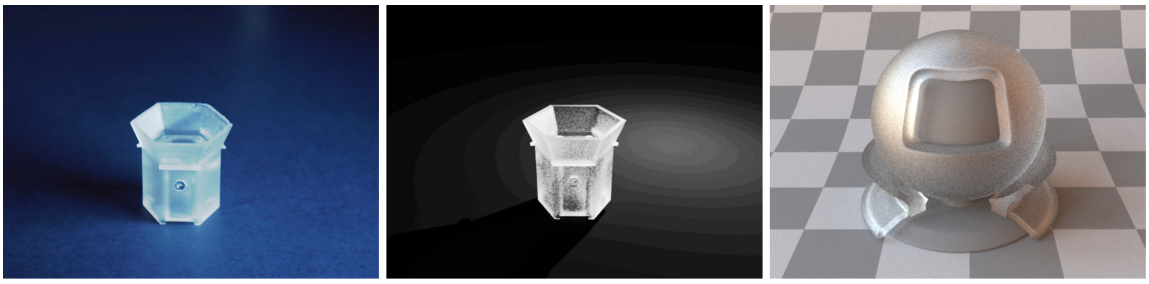



Figure 3.4: **Light model comparison**: The mesh model of the AreaLight model was chosen to match the real LEDs array set as shown on the left. Our simulation model matches closely in terms of spatially varying illumination obtained on the sensor surface.

lighting [60] in computer graphics. The key parameters in *AreaLight* are mesh, defining the geometry of the light and the three-dimensional intensity of each color. Figure 3.4 shows the comparison of the physical LEDs used in our prototype sensor and *AreaLight* model. We use the differentiable rendering ability available in Mitsuba2 [72] to obtain the color intensity for each LEDs set used in our simulation. The final optimized values were [5.23, 0.00, 0.00], [0.17,6.73,0.00], and [0.00,0.00,6.83] for red, green, and blue LEDs, respectively.

### Gelcore model

Gelcore refers to the translucent supporting structure inside the sensor, as visualized in Figure 3.6. The geometric model of the gelcore is exported from SolidWorks, a 3D geometry modeling tool. The gelcore material is modeled as a dielectric with roughness. The dielectric material model uses microfacet theory[99] with normals chosen using GGX distribution. The model uses physically accurate fresnel diffraction terms, which is essential for modeling scattering losses and roughness. Figure 3.6 shows the comparison between real gelcore, rendered gelcore, and the material visualization using a spherical ball. For more complex sensor design materials, one can model the material as a linear combination of different microfacet BRDFs using the isotropic GGX parametric model and its parameter could be optimized using differentiable rendering, as shown in [84].



Reference image Simulated gelsight material Material Preview

Figure 3.5: **Gelcore material**: The real translucent gelcore in GelSight(left), the simulated Gelcore(middle) with rough dielectric material model and preview of a sphere(right).

### Elastomer surface

In this section, we define the geometric and material model of the soft deformable elastomer surface. We use a diffuse material model for the elastomer surface to match the nature of the material's reflectance. This material model is parameterized by a 3-dimensional vector, which describes the ratio of light reflected to that of light received at the surface. We used [0.50,0.39,0.45] and [0.26,0.23,0.38] for flat gel surfaces and dome-shaped gel surfaces. Similar to the light intensities, we used differentiable rendering for optimizing material parameters.

The 3D geometry of the deformable layer is modeled as a heightfield[92]. A heightfield is a 2D matrix with each value representing height. This representation allows modifying the geometry by performing image processing operations on the matrix.

To obtain the deformed elastomer surface when an object is pressed against the sensor, we subtract the height of the object from the height of the undeformed elastomer surface. Note that because of the continuity of the elastomer material, the deformation of the elastomer is a 'smoothed-out' shape of the object in contact. We propose a simplified model of this 'smoothing out' effect by convolving the object's geometry with a kernel to generate the heightfield of the elastomer surface. The kernel is defined as

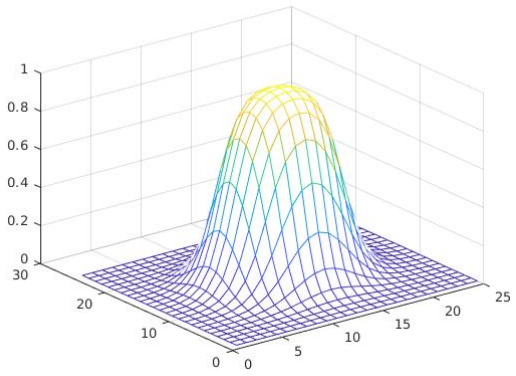



Figure 3.6: **Geometric smoothing for approximate surface deformation**: Deformation kernel with p=1 and m=200 used to smooth the heightfield

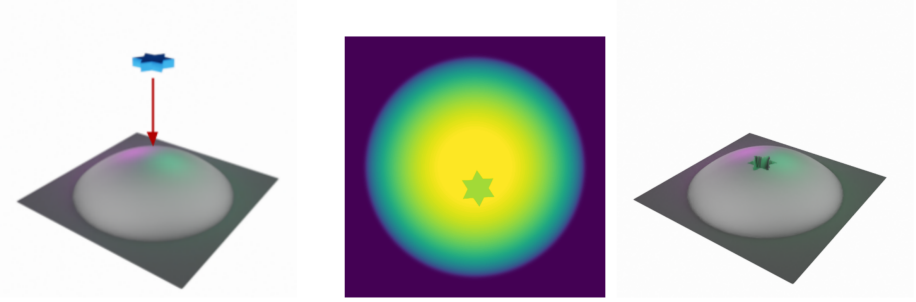
$$k(x,y) = \frac{m+1}{m + \exp\left(r \times p\right)} \tag{3.1}$$

$$\text{where } r = \sqrt{x^2+y^2}, x, y \in [-\left\lceil \frac{6}{p} \right\rceil, \left\lceil \frac{6}{p} \right\rceil] \tag{3.2}$$

This is a simplified method to get material deformation around the edges. However, it is not exact and depends on the depth of the press against the sensor. For our datasets, we found $p = 1$ and $m = 200$ work well by looking at the size of edges in sharp objects. Figure 3.7 visualizes the 3D view of the undeformed heightfield and deformed heightfield after convolution.

### 3.2.3 Results and Discussion

In the following section, we describe the data collection process and the experiments to validate models of light. We then show the sensor simulation when objects of various shapes contact the sensor at various locations. We used Mitsuba [40], which is an open-source forward renderer with a rich library of material models and light transport integrator methods for generating images with models proposed in the paper. The key time-consuming raytracing components are implemented in C++ with GPU acceleration in the Mitsuba renderer. We used Mitsuba's Python



3D view of elastomer surface and shape | Heightfield | 3D view of heightfield

Figure 3.7: **Heightfield visualization of sensing surface with an indenter**: The interaction of a star pressed against a sensor (left) and the deformed elastomer surface can be represented using a heightfield image (middle) and the corresponding 3D view is shown on the right.

API for rendering all the images. The code is available at https://github.com/CMURoboTouch/tactile_optical_simulation.

### Data Collection

We constructed an optical benchtop setup using Thorlab parts. We mounted the prototype GelSight sensor on a XY movable stage using custom-designed 3D printed parts. We mounted the objects to be pressed against the sensor on a vertically movable stage to control the depth of the press. Our experimental setup is shown in Figure 3.3. The bench-top setup allows for precise control of the depth of press against the sensor surface and makes static indentations. In the GelSight prototype, we used a Raspberry Pi V1 camera, as it is compact and allows access to raw images and jpeg images. We plugged the LEDs into a breadboard which allowed us to control the individual color LEDs.

We collected 2 datasets of real images using our sensor setup. The first dataset contains variation along elastomer surface geometry and indentation depth. The second dataset has a variation on the location of contact on the elastomer surface.

We used a $4\,\mathrm{mm}$ diameter metal ball and two 3D-printed shapes as shown in Figure 3.8 for pressing against the sensor. The first dataset contains 36 images in total with 24 images using flat elastomer surfaces (3 shapes, 2 pressing heights, 4

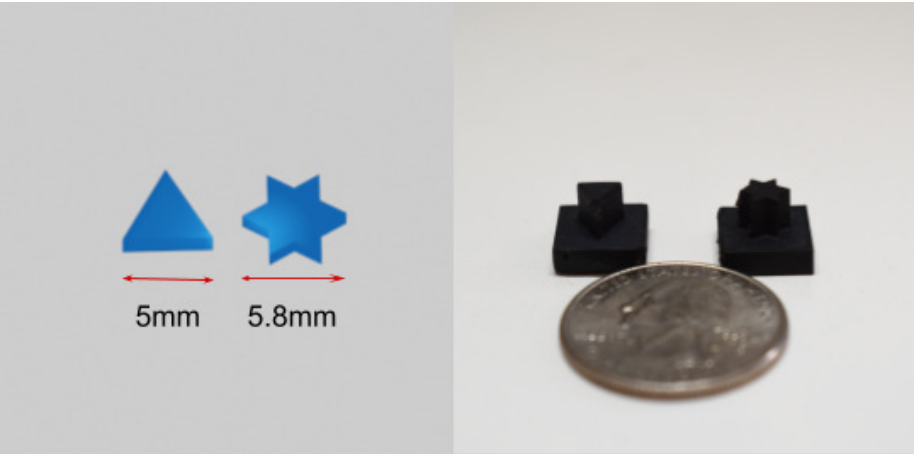



Figure 3.8: **3D printed shapes for dataset collection**: The image on the left shows the shapes visualized in a blender and the image on the right shows the real shapes placed beside a US quarter. These shapes are pressed against our sensor for data collection.

elastomer surface locations) and 12 dome elastomer surfaces (3 shapes, 1 pressing height, 4 elastomer surface locations). Dome-shaped elastomer surface was found to have better light distribution and contact for tactile sensing[106]. This dataset contains challenging simulation scenarios due to interreflection in a star shape, sharp edges in a triangle and star, and unknown boundary in a metal ball. The second dataset contains 16 images in total, and the flat elastomer surface contains 10 locations with the ball and 6 locations with a triangle. This dataset is used to evaluate if the simulation is able to model the variation in light intensity and color at different locations on the elastomer surface.

The collected datasets were hand-annotated for finding the object location w.r.t to the sensor surface. We used the camera parameters to obtain the world coordinates of the objects. The world coordinates of the objects were used to generate heightfields and place the generated geometry into the simulation environment.

### Lighting model

In this section, we evaluate the proposed light model and compare its intensity and color at different locations by capturing light probe images[17]. The *light probe* refers to a polished metal ball placed at the location where the image has to be

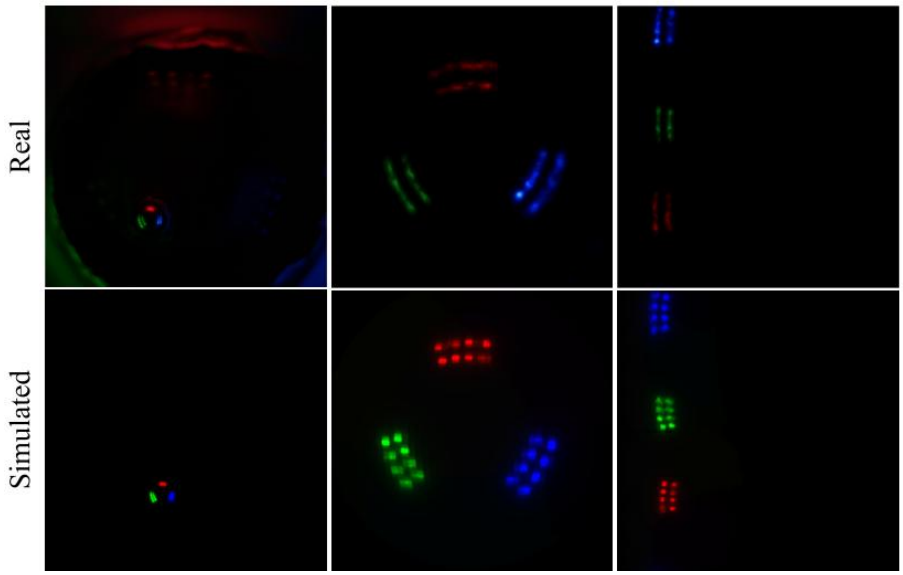



Figure 3.9: **Light probe comparisons to visualize illumination inside the sensor**: The first column shows the image from the camera viewpoint. The second column compares cropped probe images seen from the camera in real and simulated cases. The last column compares the environment map[17] of the corresponding image. This image uses our light model without gelcore with optimized scene parameters. The image shows a close match of simulated and real-world light patterns in terms of shape and color.

simulated. The light probe image essentially means to capture an image of the scene with only scene lights(sensor LEDs in our case) and the light probe placed at the location where the model has to be tested. To capture light probe images from the real sensor, we removed the elastomer surface and then inserted a metal ball at the same height as the elastomer surface. Figure 3.9 shows a comparison between real and simulated images for full camera image, cropped light probe image, and the corresponding environment map(which represents the light received by the metal ball in polar coordinates). The light probe images show a close match in the shape of real and simulated light models.

#### Evaluation of Sensor simulation

In this section, we bring together the light model, the gelcore material model, and the elastomer surface model to simulate the GelSight sensor when different objects

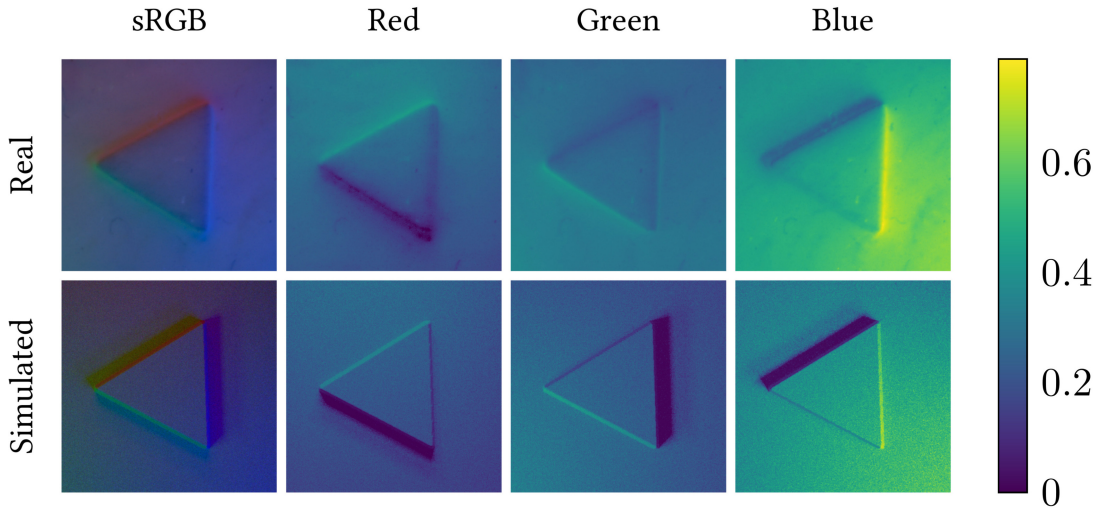



Figure 3.10: **Simulation-Real comparison for GelSight hexagon sensor**: Comparison between real and simulated image along different color channels

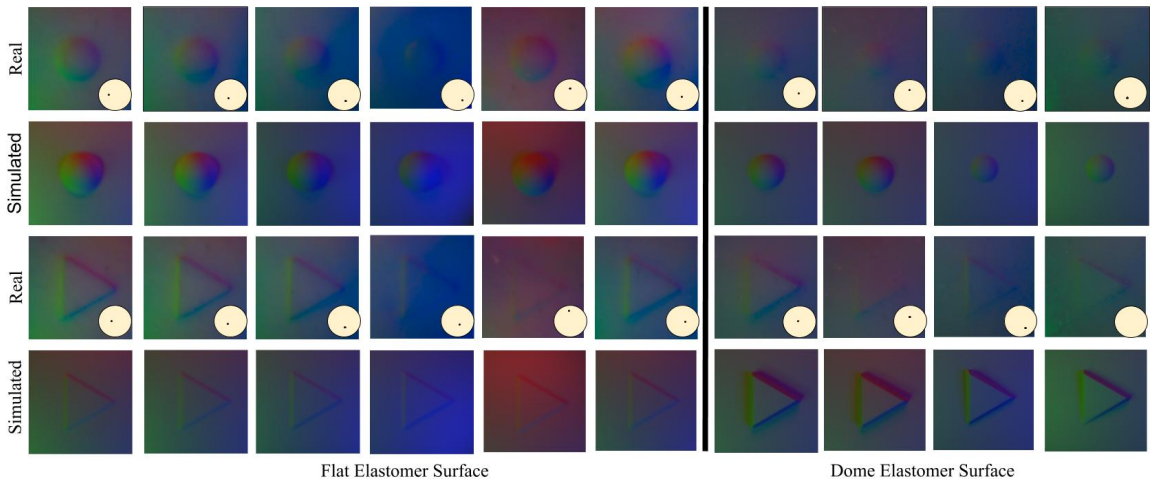



Figure 3.11: **Comparison of tactile images with indenter at multiple spatial locations on the sensing surface**: The images in the odd row show zoomed-in real sensor images and a small inset in the bottom right corner shows where the indentation was made on the original sensor. In all cases, the indentation depth was $1\,\mathrm{mm}$. The even row shows images rendered using our system. The comparison shows a close match in terms of color and intensity of the lighting variation at different parts of the elastomer surface using our simulation system.

are pressed against the sensor.

We compare against 2 previous approaches, [88] and [28], which used directional light and Phong material for elastomer surface to simulate GelSight; [106] used directional light assumption and diffuse material for elastomer surface to reconstruct the shape. We use Monte Carlo simulation for both methods as it gives physically accurate results for material models used in the above approaches.

To find the light intensity of our simulation and comparative methods, we took images with single-color LEDs switched on in GelSight and used the average intensity in the middle of the camera image to scale the corresponding light intensity in the simulation. For the elastomer reflectance color parameter, we used a single image with a metal ball pressed in the middle of the sensor and manually tuned the 3-dim RGB reflectance vector. We used the mean squared error between the RGB image to estimate the parameters. *Note*: In all our final comparisons, we used sRGB images for visualization and linear images for quantitative evaluations. Linear images represent the true radiance received by the sensor for each color channel. sRGB images represent images that are post-processed for human visualization.

**Per channel comparison**: We considered a case when a triangle is pressed against a flat elastomer surface at a depth of $1\ \mathrm{mm}$. Figure 3.10 shows the comparison of RGB channels between the real and the simulated sensor images. The figure shows a close match in terms of light intensity in all the channels especially high values on the right side in the Blue channel. Though, we note that the simulated images have large shadows which are missing in real images. However, the edges of shapes match closely in real and simulated images.

**Spatial variation**: For this experiment, we used dataset 2, which consists of shapes pressed against sensors at multiple locations. As can be in Figure 3.11, our simulation closely matches the colors and intensities for the smooth ball and sharp triangles at various locations shown in the inset of the reference images.

**Comparison against other methods**: For this experiment, we used dataset 1, which consists of variations along shapes, indentation depth, and elastomer surface geometry. For qualitative comparison refer to Figure 3.12, we show balls pressed at different locations in the first 3 columns. Our method closely matches the strong red color in the 2nd column and the strong green color in the 3rd column. The directional light model assumes that the light intensity remains constant across

the elastomer surface. This effect is seen in columns 1-3, where the image remains the same irrespective of where the ball was pressed on the sensor surface. In the 4th column, only our method shows correct colors at the edges of the star which has strong interreflections between its edges. This case is only possible to simulate using ray tracing which takes multiple bounces of light into account while the image formation process. Columns 7-10 show the comparison of shapes pressed against a dome-shaped elastomer surface. The dome shape is particularly challenging due to the concave shape of the elastomer surface, which can have strong interreflections. As can be seen, our method has correct colors at the edges and shows variation in color when shapes are pressed at different locations at the sensor. We notice that previous methods have large grayish ambient color. In a real sensor the intensity of light increases if the location of contact is closer to a light source. However, the previous method assumes constant lights and has failed to accurately capture the color at locations close to light sources.

For quantitative comparison, we used traditional signal processing metrics like Mean Squared Error (MSE), Signal-To-Noise Ratio (SNR), Symmetric mean absolute percentage error (SMAPE), and a metric from image similarity literature SSIM[37]. Since SSIM is insensitive to luminance change, contrast change, and small geometric distortions. It produces a single number per pixel by finding the mean, variance, and correlation per channel. To compare images, one can take the mean over all the pixels to obtain a single number for the image known as Mean SSIM, which we use in this paper. We used a cropped patch of $600 \times 600$ around the indented shape for calculating metrics. Table 3.1 shows the average metric values calculated using images from both datasets. We consistently outperform previous methods in a range of image similarity metrics.

Figure 3.13 shows the qualitative comparison of larger and more complex 3D shapes simulated using our system.

#### Timing and image quality trade-off

We use Monte Carlo process in our simulation which uses multiple samples per pixel to obtain the color of that pixel. Therefore the total time taken to render each image and its quality is a function of sample-per-pixel(spp), length of light path(l)

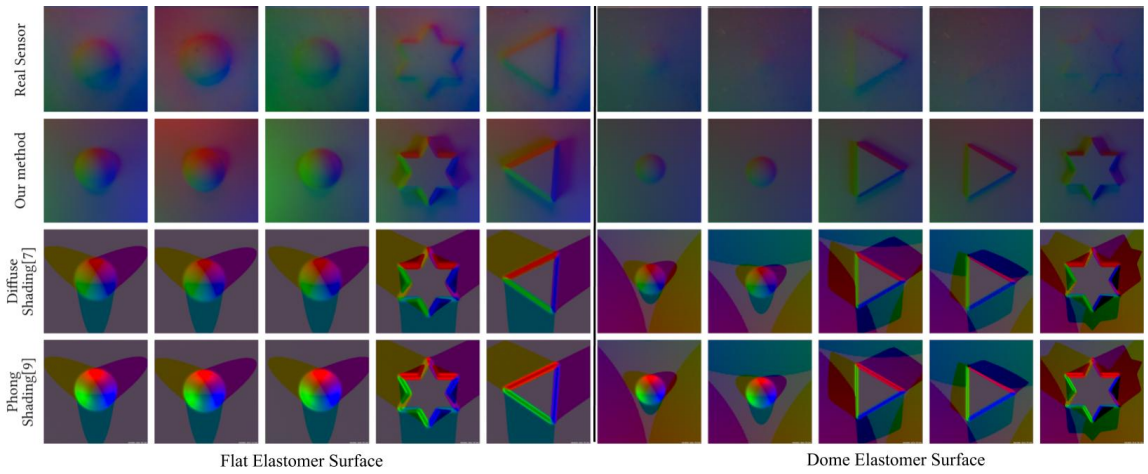



Figure 3.12: **Qualitative comparison between different simulation methods**: The baseline methods are able to capture color and intensity around the center region. However, only our method is able to capture the spatial variation and matches well with the real sensor images for multiple object geometry and elastomer surface geometry.

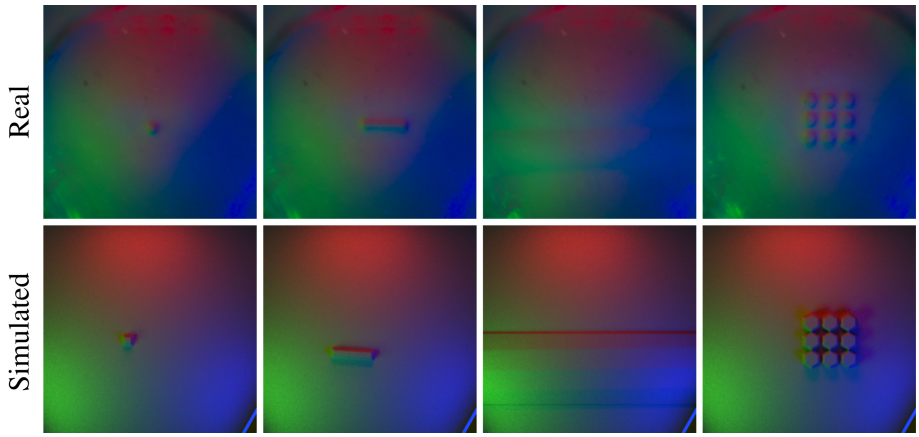
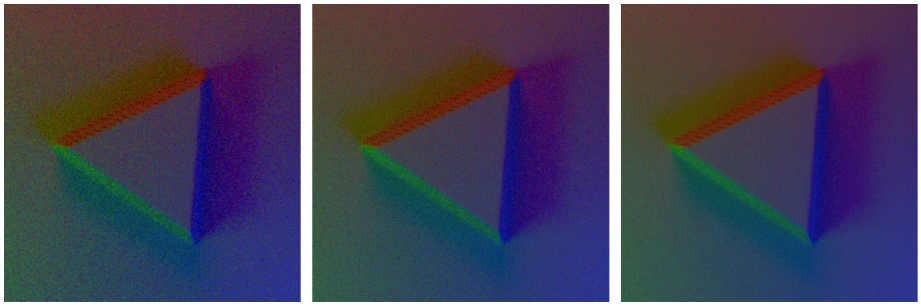



Figure 3.13: **Simulation-Real comparisons for complex 3D shapes**: Comparison of simulated and real tactile images. Real images were collected by pressing the objects $1\,\mathrm{mm}$ against the sensor surface

Table 3.1: Comparison of the simulated tactile images and the real ones on different metrics. The ↓ arrow shows that a lower value is desired and vice-versa.

| | MSE ↓ | SNR ↑ | SMAPE ↓ | SSIM ↑ |
|---|---|---|---|---|
| Diffuse surface + Directional Light[106] | 0.004 | 2.705 | 0.839 | 0.387 |
| Phong shading + Directional Light[28] | 710.618 | -50.152 | 0.828 | 0.388 |
| **Our method** | **0.001** | **8.562** | **0.445** | **0.841** |

and size of the image(h x w). The quality of the rendered image and computation time both increase if we increase any of the mentioned parameters. We found in our experiments $spp = 8, l = 4$ for rendering image size 600 x 600 leads to frame rates of 10Hz and is optimal. While keeping image size and l constant, the timings for spp 4, 8, and 16 are $64\,\mathrm{ms}$, $95\,\mathrm{ms}$, and $174\,\mathrm{ms}$. Figure 3.14 shows the image comparison with varying spp. While keeping spp and l constant, the timings for image sizes 128x128, 256x256, 512x512, and 1024x1024 are $33\,\mathrm{ms}$, $36\,\mathrm{ms}$, $55\,\mathrm{ms}$, and $126\,\mathrm{ms}$. These runtimes were recorded using python3.7 $timeit$ module on a 32 CPU core machine with GeForce RTX 2080Ti for a scene with 318510 geometric faces.



spp 4 spp 8 spp 16

Figure 3.14: **Rendering speed versus noise**: Qualitative image comparison of images rendered at different spp

## 3.3 Optical simulation for advanced tactile sensors

A novel vision-based tactile sensor with a curved tactile sensing surface was introduced in [77]. Since then, there have been numerous attempts to develop tactile sensors with curved sensing surfaces [19, 73]. Curved sensing surfaces enable dexterous robotic manipulation [6] without the need for reorientation of the robotic arm to perceive objects from multiple sides [77]. It also enables a large contact area when the object is being manipulated without arm reorientation. Therefore, there is a significant interest in developing curved tactile surfaces. Since our simulation framework is general enough to allow the simulation of curved tactile sensors, we can guide the design of the same. In the following section, we give an overview of a human fingertip-like curved tactile sensor, propose real2sim simulated model calibration methods, and compare full sensor simulation.

### 3.3.1 Human finger-like curved tactile sensor

Figure 3.15 shows the exploded view of the curved tactile sensor. The sensor consists of a hard plastic shell, soft elastomer, and a coated external layer. The sensor uses light piping through the hard plastic shell to allow light to reach all parts of the tactile sensor. This allows for the recovery of surface normals using color information.

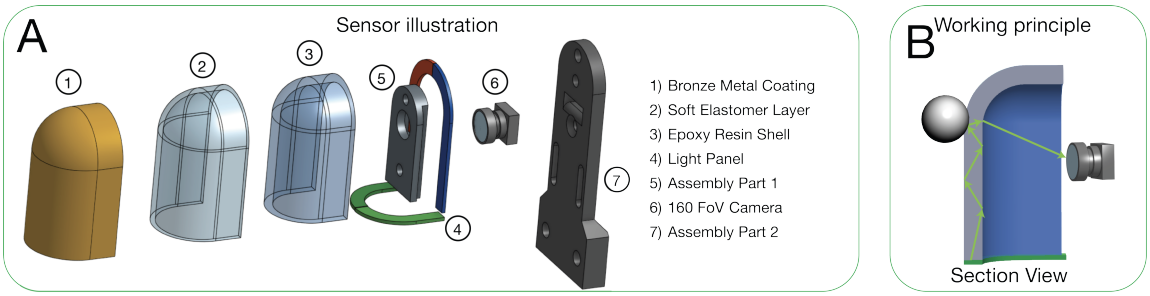
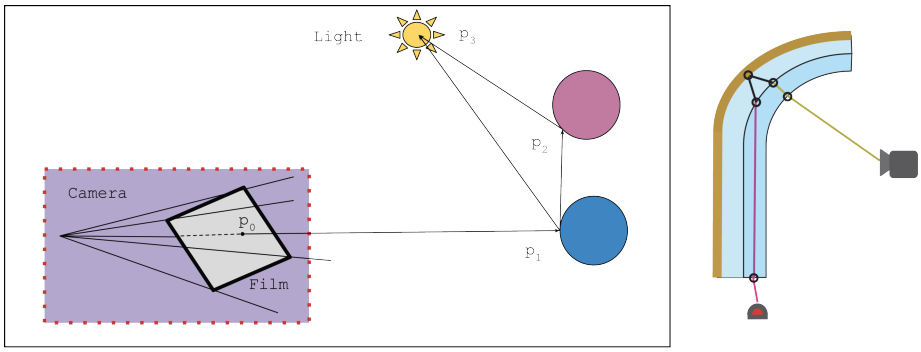



Figure 3.15: **Curved tactile sensor**: (**A**) shows the exploded view of the curved tactile sensor. (**B**) shows the path diagram as light travels from light to camera.




A: Raytracing B: Path diagram

Figure 3.16: **Path Tracing illustration**:(**A**): image construction process in terms of light paths starting from emitters, hitting single or multiple objects, and reaching the camera film. (**B**) Path diagram in a curved tactile sensor

### 3.3.2 Optical simulation method

In this section, we review light transport simulation techniques and discuss the specific algorithm *Langevin Monte Carlo Rendering*, which we use throughout this work to simulate curved tactile sensors. For a detailed description of path tracing, see Appendix A.

We leverage the Markov chain Monte Carlo light transport technique for simulating light paths inside the sensor. Specifically, we use Langevin Monte Carlo [63] to generate images as given out by the tactile sensor scene. We briefly describe light transport integral; how to use Monte Carlo to estimate the integral; leverage Markov chain Monte Carlo in the MC process; use Langevin Monte Carlo to generate samples in the MCMC process, which proposes general changes in light paths by building a differentiable geometry local approximation.

Light transport can be expressed by the following path integral

$$I_j = \int_{\Omega} h_j(\bar{x}) f(\bar{x}) d\mu(\bar{x}), \tag{3.3}$$

where $I_j$ is the pixel value of the j-th pixel, $\Omega$ is the space of all possible light paths. A light transport path $\bar{x} \in \Omega$ is a collection of line segments in the world that represent the trajectory of light traveling in the scene, with the first vertex on

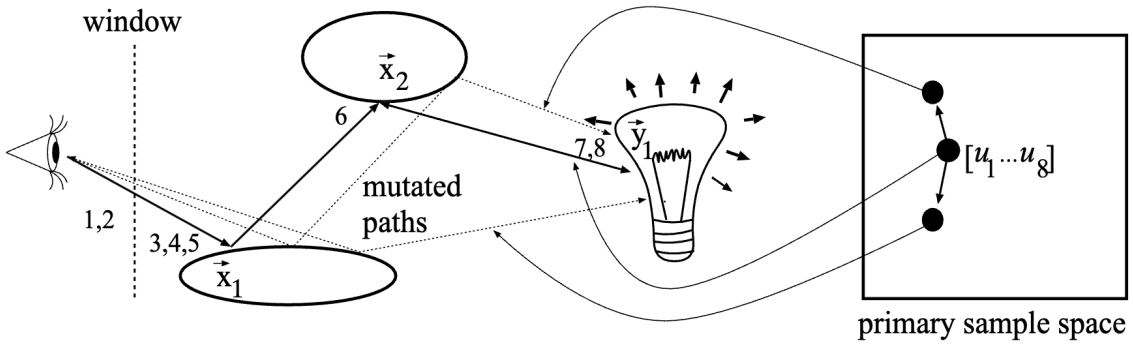



Figure 3.17: **Path mutations in MCMC techniques**: The figure is reproduced from [45]. It shows path mutations in the path-space by changing the vertex locations of the path shown in bold line style. The right side shows how these mutations can be traced to mutations in the primary sample space (PSS) of random numbers. PSSMLT and LMC techniques perform mutation in PSS.

a light source and the last vertex on a camera. Therefore, a path of length k can be represented as a vector of $k+1$ vertices $\bar{x} = (x_0, \ldots, x_k)$. The path contribution function $f$ gives the amount of light energy transported along the path $\bar{x}$. $h_j$ is the response of the pixel $j$ and $\mu$ is the measure associated with the path space $\Omega$. The path integral is solved using Monte Carlo integration as it is a high-dimensional integral. Given some way to generate paths $\bar{x}_i$ with probability $p(\bar{x}_i)$, the Monte Carlo integrator estimator is given as follows.

$$\langle I_j \rangle = \frac{1}{N} \sum_{i=1}^{N} \frac{h_j(\bar{x}_i) f(\bar{x}_i)}{p(\bar{x}_i)} \tag{3.4}$$

In Markov chain Monte Carlo light transport methods, i.e. Metropolis Light Transport(MLT) [98], path generation occurs with a probability proportional to the scalar luminosity of the light path contribution to the image $f^*(\bar{x}) = \text{lum}(f(\bar{x}))$. Therefore the resulting Monte Carlo estimator becomes

$$\langle I_j \rangle = \frac{Pb^*}{N} \sum_{i=1}^{N} \frac{h_j(\bar{x}_i) f(\bar{x}_i)}{f^*(\bar{x}_i)}, \tag{3.5}$$

where P is the number of pixels and the start-up weight $b^*$ is calculated from the set of initial samples generated by bidirectional path tracing techniques. In the original MLT method, the authors proposed three local mutations and used Metropolis-Hastings to select the proposed path. Kelemen et al. [45] proposed to perform MCMC sampling in the space of random variables used to generate paths, called *Primary Sample Space* (PSS). This led to a simple implementation and a simple mutation that is effective at *local exploration* while improving *global exploration* as well. Figure 3.17 shows the mutations in PSS. The Monte Carlo integration using the PSS becomes

$$\langle I_j \rangle = \frac{Pb^*}{N} \sum_{i=1}^{N} \frac{h_j(m(\bar{u}_i))C(\bar{u}_i)}{C^*(\bar{u}_i)}, \tag{3.6}$$

where $C(\bar{u}) = \frac{f_m(m(\bar{u}))}{p_m(m(\bar{u}))}$ for the mapping function $m$, such that $m(\bar{u}) = \bar{x}$.

For simulating curved tactile sensors, we use Langevin Monte Carlo(LMC) [62] that uses differentials of the target function, instead of manually defined mutations, to propose new mutations effective in local exploration. This technique has two technical advantages: a) it can potentially propose better mutations as it takes differential of geometry, illumination, and material property into account while generating new proposals, b) it can lead to higher acceptance rates, thus reducing the chance of a chain getting stuck in local minima. In practice, we found that the algorithm is superior to all previous approaches for exploring highly curved surface geometry with specular or glossy material. The finger-shaped tactile sensor which we aim to build in this work has a highly curved geometry with light piping through the refractive surface; hitting a glossy outer surface and refracting back to the camera.

### 3.3.3 Real2sim simulation model characterization

Previous attempts at tactile sensor design are based on intuitively buying and testing various components. Since our simulation framework is novel in the world of tactile sensor design, we had to calibrate simulation models to accurately reproduce tactile images as generated by our prototype.

In this section, we will go over two simulation model calibration steps that were essential to accurately render images using PBR - a) how to accurately model the BRDF of the coating material used on the outermost sensing surface and b) how to obtain an accurate light profile of LEDs used in our hardware prototype.

**BRDF Characterization**

[36, 56, 77] choose different coating materials made up of metal flakes of various particle sizes and in various types of individual particles. However, these works do not characterize the material or create a model in terms of BRDFs (useful for generating novel designs in simulation), which can be easily shared to identify the correct coating required or analyze the effect on tactile sensor performance. Levin et al. [49] showed that the BRDFs for metal powder coating with varying particle sizes are well approximated as a mixture of diffuse and specular BRDFs. Motivated by this result, we model the BRDFs of 2 coating powders - a)Aluminium Powder($1\mu$m spherical particles) b) Bronze Powder (industrialspec.com $12\mu$m); as a *Blended* BRDF of *Diffuse* and *RoughConductor* components. To calibrate this model, we created a simple setup to capture BRDFs of these metal coating as shown in Figure 3.18B. Our setup consists of a monochrome camera, color filter array, collimated light source and cylinder whose front half(surface facing the camera) is painted with the coating material. We perform High Dynamic Range(HDR) process to capture images such that the pixel values are proportional to radiance. We replicate the scene in simulation and fit parameters of our *Blended* BRDF model to match the radiance along a horizontal strip on the cylinder as shown in Figure 3.18C. Specifically, we fit 3 diffuse reflectance terms, 1 specular roughness term, 6 complex index of refraction term and 1 mixture coefficient term. Through our experiments, the coefficients for semi-specular Bronze coating are (diffuse reflectance=(0.8, 0.5, 0.3), roughness=0.175, eta=(0.475 0.576 0.764), k=(0.877 0.811 0.631), mixture coefficient = 0.85) and coefficients for Aluminium powder with spherical particles are (diffuse reflectance=(0.37, 0.4, 0.37), roughness=NA, eta=NA, k=NA, mixture coefficient = 0.0).

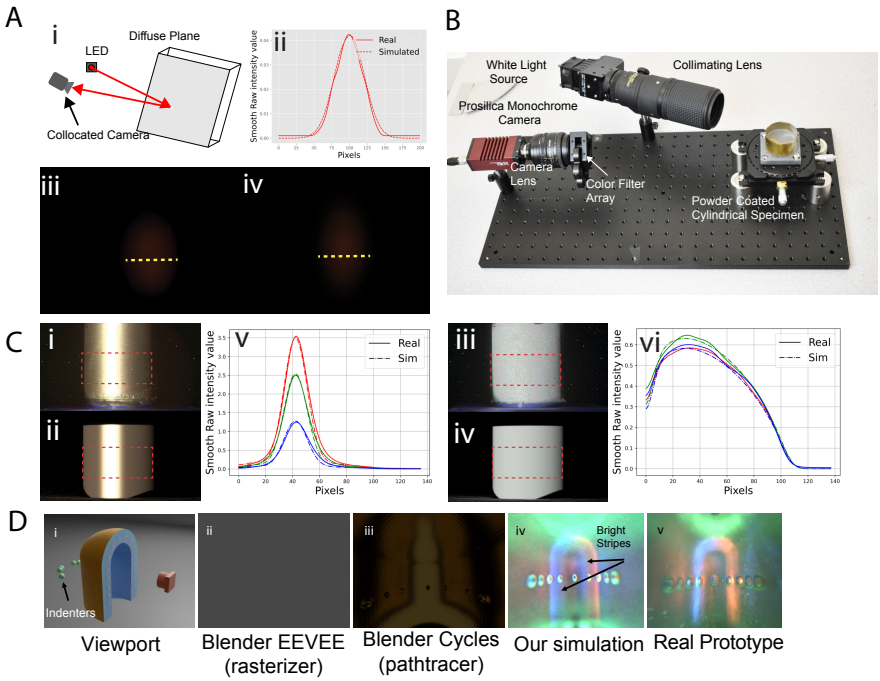



Figure 3.18: **Simulation model characterization for illumination and surface coating**. **(A)** Illumination model characterization: i) Scene setup illustration consists of a collocated camera, light source, and a diffuse plane; ii) Comparison of radiance profile(across the yellow region) between the physical light source and simulated light model; iii) Real image captured using our scene setup from a real LED light source(OSRAM LRT64F); iv) Simulated image using the calibrated light profile. **(B)** BRDF Acquisition Setup: The acquisition setup consists of the collimated light beam(Prizmatix UHP HCRI+ $200\,\mathrm{mm}$ Nikon AF-S lens) shining on a cylinder(fabricated using transparent PDMS mixture) with coating powder under inspection and monochrome camera(Prosilica GX camera) with color filters of wavelengths $450\,\mathrm{nm}$, $53\,\mathrm{nm}$, and $660\,\mathrm{nm}$. **(C)** BRDF Characterization Results: (i) and (iii) show HDR captured using our BRDF acquisition setup, shown in (B). (ii) and (iv) show images obtained using our simulation framework with replicated acquisition scene. In figure (v), we compare the measurements for *Bronze coating* along a horizontal row(as shown in the red box), between the Real (i) and Simulated (ii) image. We can closely match radiance measurements for all 3 color values. We repeat the experiment for *Aluminium coating* and show obtain a good match in figure (vi). **(D)** Full vision-based tactile sensor comparison: (i) Shows the camera, vision-based tactile sensor, and indenter scene setup, which we used to analyze our designs. (ii) and (iii) show the images simulated using Blender EEVEE and Blender Cycles renderer respectively. (iv) HDR image simulated by our framework using calibrated simulation models. Our simulated results are a close match to the physical prototype as we can reproduce a)Bright light stripes due to focused LEDs and b) Light piping of red color from the right to illuminate the spheres on the left and similarly for blue color. (v) HDR image captured with our hardware tactile sensor prototype, when the sensor is indented with a set of spheres.

### 3.3.4 Light Model Characterization

Padmanabha et al. [73], Taylor et al. [90], Wang et al. [100] have used various LED light sources with unknown near-field light profiles. The LED manufacturer provides either very sparse or no data about the light profile for these LEDs.

To capture the near-field illumination through various physical light sources, we developed a new light model with a finite area and Illuminating Engineering Society (IES) profile for a point on that surface. The IES profile is an industry-standard data format to specify radiance emitted along a direction distributed on a unit sphere.

In this section, we present a simple calibration step for finding the IES profile for the light sources. Our setup is as shown in Figure 3.18A. The setup consists of a collocated Raspberry Pi camera(v1), a light source under inspection, and a diffuse plane (calibrated A4 white paper). We perform the HDR process for obtaining radiance image and manually fit a function $f(\theta, \phi) = 2\exp\Big\{-\Big(a\tan\theta\cos\phi\Big)^2 - \Big(b\tan\theta\sin\phi\Big)^2\Big\}$. We reproduce the setup in simulation and render the same scene with the calibrated light profile. The fitted light profile parameters for OSRAM LRT64F and OSRAM LBT64F were $(a, b) = (4, 3.33)$ and $(a, b) = (5, 2.5)$ respectively.

We found that if the LED has a flat lens, then *AreaLight* with the corresponding physical dimension is a good enough model for the light. We specifically characterized the Chanzon 5730 SMD LED using the *AreaLight* model. In Figure 3.19 we show that using this analytical light model we are able to closely match the radiance values in real and simulated images.

### 3.3.5 Fluorescent material characterization

GelSight FinRay sensors use fluorescent paint for illumination. Therefore, in this section, we introduce the fluorescent material model and simplified fluorescent simulation technique in PBR. In the first part, we will describe the simplified model for fluorescent paint and its calibration process. In the second part, we will describe the simplified simulation model of fluorescence lights in the PBR framework that allows us to simulate the full sensor.

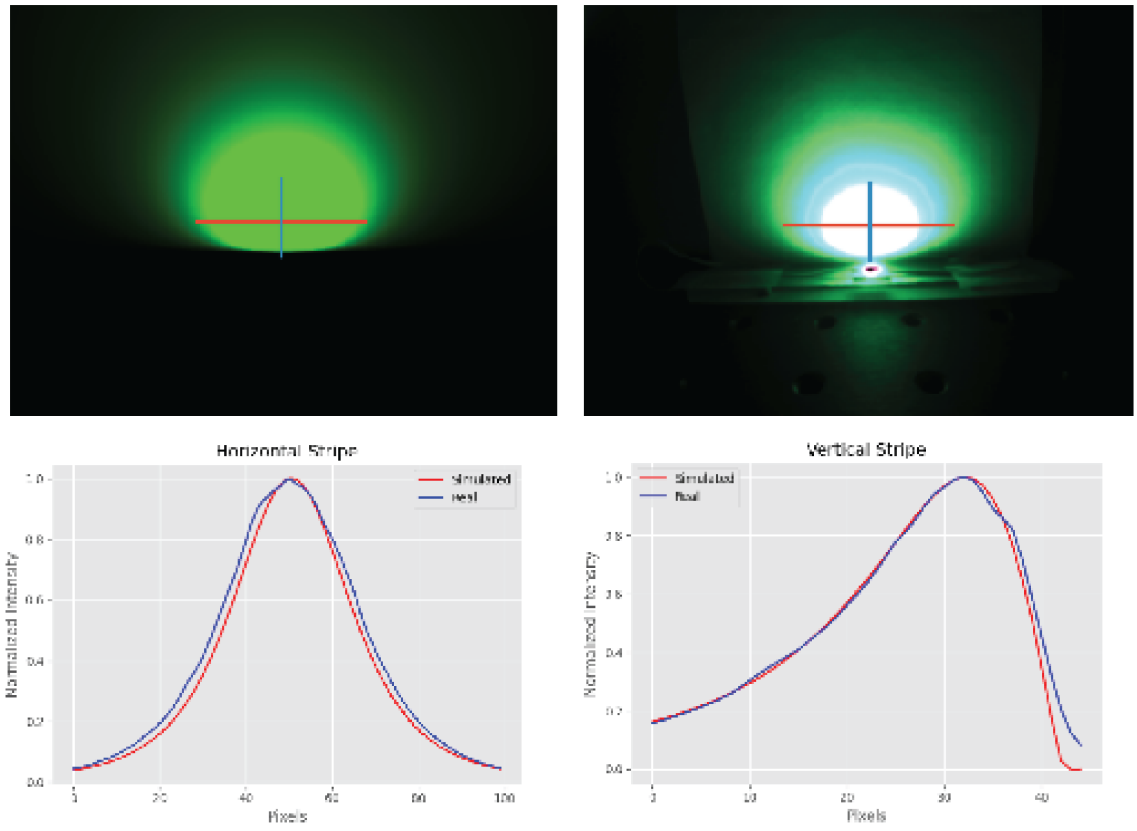



Figure 3.19: **Characterization of flat-lens LEDs**: In this visual, we calibrate the Chanzon 5730 SMD green LED using the *AreaLight* model. The radiance plot shows a close match between the real and simulated images.

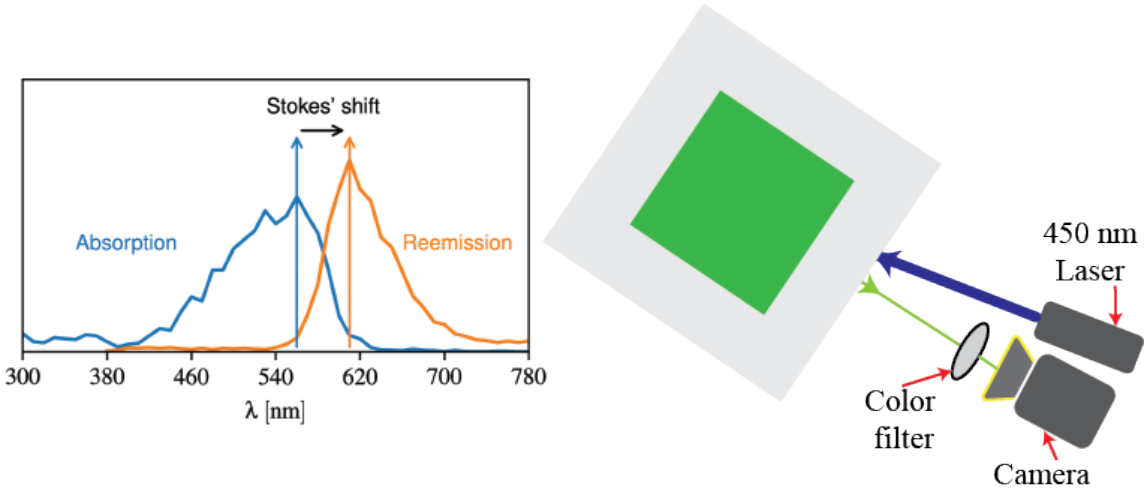



Figure 3.20: **Fluorescent model and calibration setup**: (A) shows a canonical fluorescent material model [38]. It consists of absorption and reemission spectra whose peaks are separated by Stokes' shift. (B) shows the imaging setup we created to capture the reflectance at the excitation wavelength, $\lambda = 450$nm for calibrating fluorescent paints used in GelSight Fin Ray.

**Fluorescent paint calibration** For accurate simulation, we need to calibrate the fluorescent paints used in the sensor. Our imaging setup consists of a CM3-U3-13Y3C-CS 1/2" Chameleon color camera, $450\,\mathrm{nm}$ Blue Alignment Laser Diode Module (Edmund Optics) and 8 color filters with central wavelengths – $405\,\mathrm{nm}$, $450\,\mathrm{nm}$, $500\,\mathrm{nm}$, $532\,\mathrm{nm}$, $560\,\mathrm{nm}$, $600\,\mathrm{nm}$, $630\,\mathrm{nm}$, and $660\,\mathrm{nm}$. We calibrated two fluorescent paints (Liquitex BASICS Acrylic Paint Red Fluorescent ASIN B07F48YG5F and Liquitex BASICS Acrylic Paint Green Fluorescent ASIN B07F48WZWL) made in a flat sample. The calibration setup is shown in Figure 3.20. We assumed that the fluorescent paint is diffuse in nature – for any incident direction, the amount of outgoing light radiance remains the same.

A fluorescent material could be characterized by absorption and emission spectra. The first defines which incident light wavelengths are absorbed and lead to re-emissions. The second describes the amount of re-emission across all incident wavelengths. The difference between the spectral positions of the band maxima of absorption and re-emission is called a Stokes shift [69]. According to [117], if the spectra are not very spiky, absorption and emission spectra can be modeled by a 4-parameter analytic distribution, a variant of skew Cauchy distribution. The

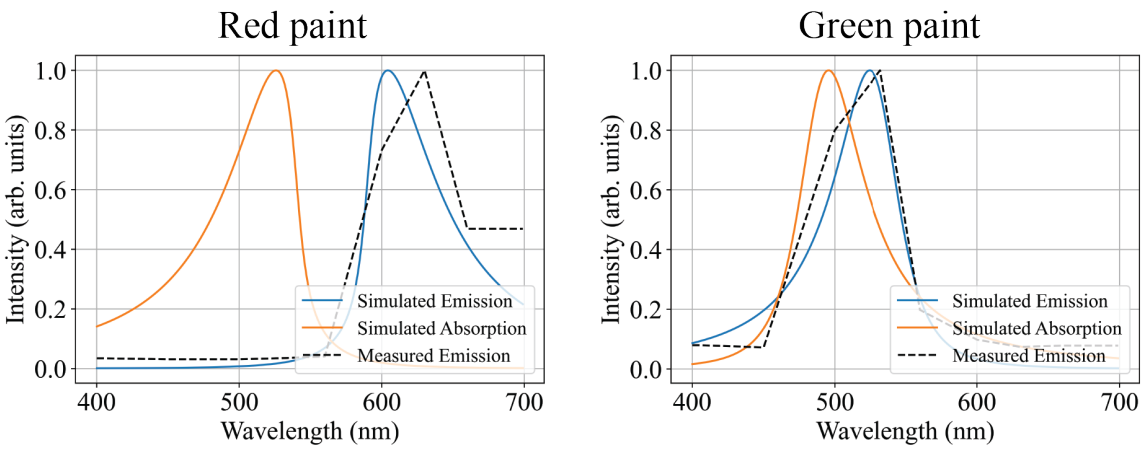



Figure 3.21: **Fluorescent paint calibration**: This shows the comparison of measured emission spectra and simulated emission spectra using a 4D parametric model for red fluorescent paint (left) and green fluorescent paint (right).

spectra value at wavelength $\lambda$ is given by the function

$$\mathrm{f}(\lambda|\lambda_0,\gamma,\omega,\mathrm{h}) \quad = \quad \frac{\mathrm{h}}{[\gamma^2+(\lambda-\lambda_0)^2]}\left\{\frac{1}{\pi}\arctan\left[\frac{\omega(\lambda-\lambda_0)}{\gamma}\right] \quad + \quad \frac{1}{2}\right\} \tag{3.7}$$

where $\lambda_0$ is the peak wavelength, h is height parameter, $\gamma$ is width and $\omega$ is the skewness parameter. We manually fit the measured data and choose Stoke's shift for the paint to be $100\,\mathrm{nm}$ and $50\,\mathrm{nm}$ for red and green fluorescent respectively, based on reflectance data. For calibrating the non-fluorescent reflectance, we collected images in room light and matched them to the closest color in a traditional colorchecker. We found that the non-fluorescent reflectance of the red and green fluorescent paint is very similar to colorchecker *Red* and *Green* colors respectively.

**Fluorescent simulation model** We found that the fluorescent effect leads to a reflectance of around 2% - 5% at the desired wavelength. In addition, it depends on the incident excitation wavelength. For a fast approximate model, we created a textured light source whose intensity is proportional to the distance from the center of the blue light source. The color of the LED is chosen based on our calibration model in the previous section. Pictorially, the fluorescent light source looks as shown in Figure 3.20. Our components modeled in the optical simulation are

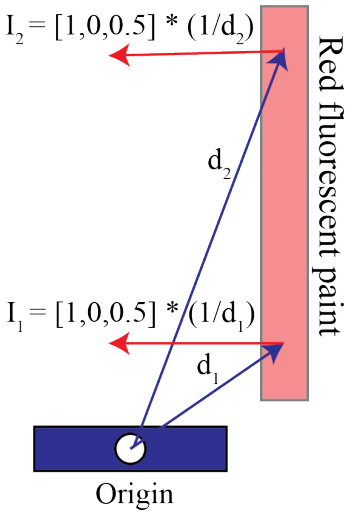



Figure 3.22: **Approximate rendering of fluorescent lights**: This image shows the visual of the efficient rendering of fluorescent paint lights using our parametric reflectance model for simulating GelSight Fin Ray sensor.

shown in Figure 3.22. Thereafter, we use GPU path-tracing to generate all the images.

### 3.3.6 Full sensor Renderings

Given the rendering algorithm and the calibration methods for simulation models, we can reproduce the images which are a close match to the images collected from our hardware prototype. Figure 3.23 compares the simulated tactile images with real world prototype tactile images for human-finger tip sensor and GelSight FinRay sensor. Figure 3.23A highlights 2 key feature matches a) bright stripes of LED light in simulation and physical tactile sensor prototype b) *Light piping* of red color from right to illuminate the spheres on the left and similarly for blue color.

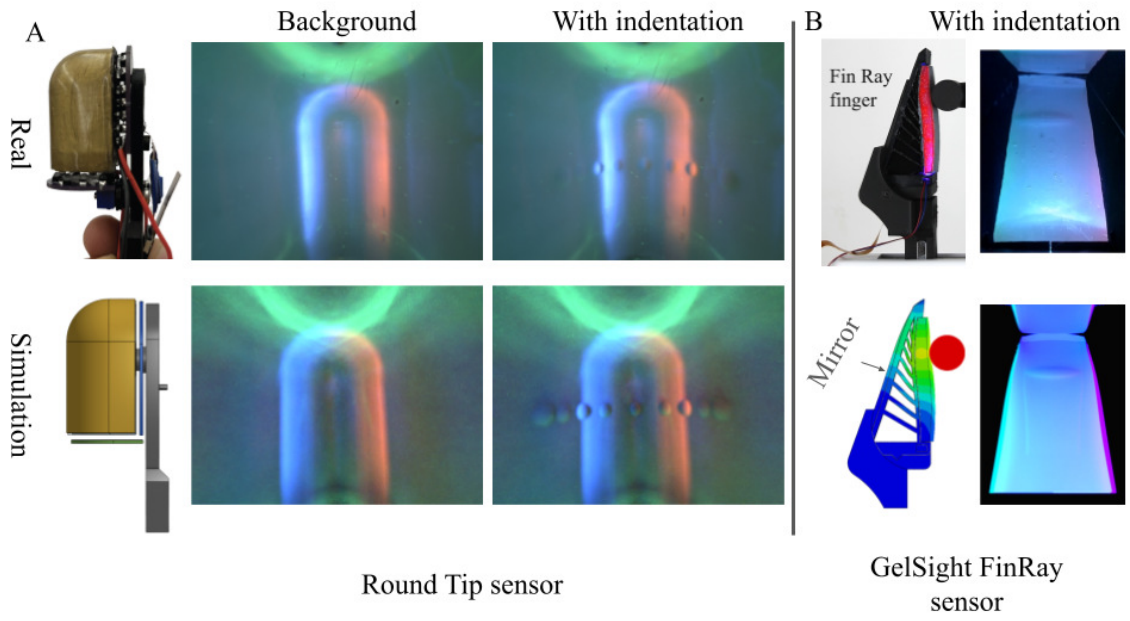



Figure 3.23: **Sim2Real comparison of challenging variants of GelSight sensor**: (**A**) shows the comparison of simulated and real tactile images with and without sphere indentation for human-finger tip sensor. (**B**) shows the comparison of simulated and real tactile images with indentation for GelSight FinRay.

# Chapter 4

# Sensor Design Framework

## 4.1 Introduction

In this chapter, we leverage our optical simulation framework to propose a sensor design framework for vision-based tactile sensors. The design framework for vision-based tactile sensors consists of three steps: sensor generation, physics-based simulation, and automatic design score calculation, as shown in Figure 4.2. In the following section, we present a discussion of design goals and present our framework with a human fingertip-like sensor as a case study. We present a sensor design space and an automated design solution using our framework for curved sensor design. We then highlight the utility of our optimized design in 3D surface reconstruction task in simulation and real-world tasks. We also present two robotic applications, robotic grasping and robotic surface inspection.

## 4.2 Design Goals

A well-designed vision-based tactile sensor [90] can output 3D shape information (geometry normal), detect incipient slip and estimate the spatial distribution of 3D contact forces using simple image processing and calibration steps. The key principle used to obtain high-resolution shape is model-based photometric stereo [44]. This constrains the design space as it requires that at least three colors of light

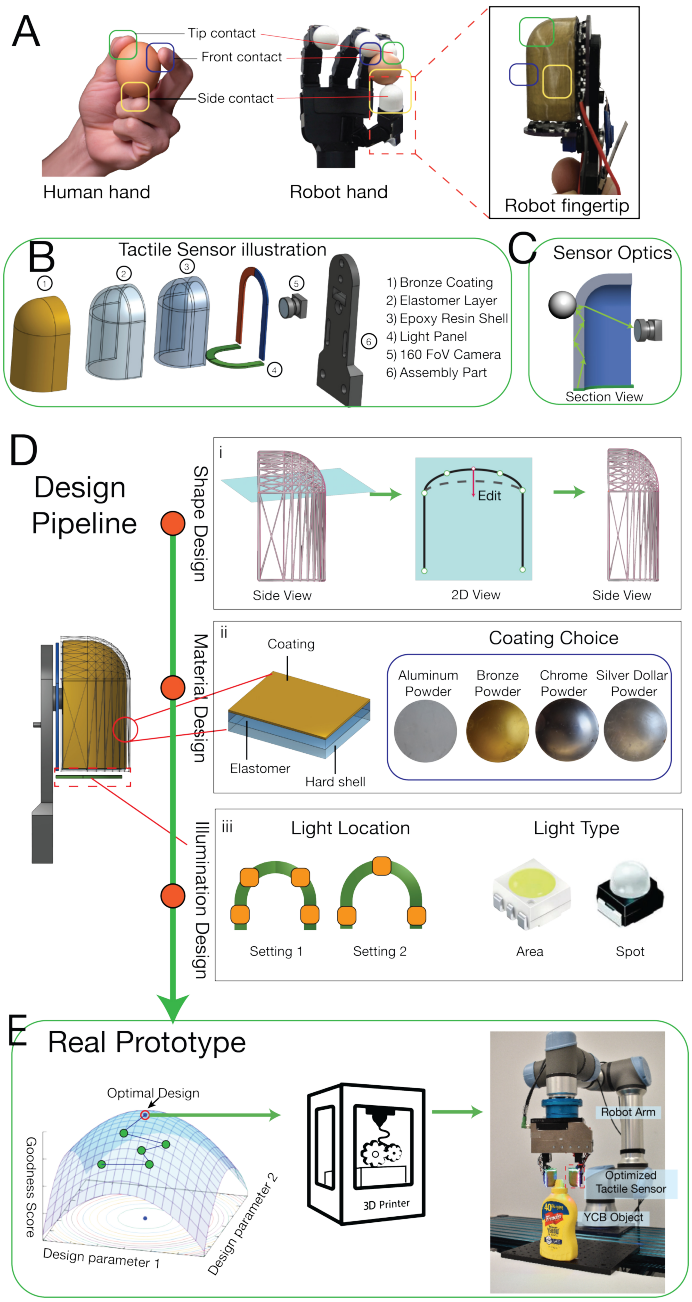



Figure 4.1: **Curved sensor illustration and proposed design framework**. (**A**) A human hand and a robotic hand with tactile sensors manipulating an egg. The right-most figure shows the zoomed-in version of the fingertip GelSight sensor. In (**B**) shows the exploded view of the fingertip sensor and important optical components. (**C**) illustrates a light path propagating inside the sensor and contributing to the tactile image. (**D**) shows the design pipeline - we start with sensor shape generation, using a low dimensional curve parameterization, selection of sensor material properties, and illumination system design, to procedurally a new design. Finally, in (**E**) we use gradient-free optimization to choose the best sensor shape, illumination, and sensing surface coating material. We subsequently manufacture the optimal sensor and test it on various robotic applications.

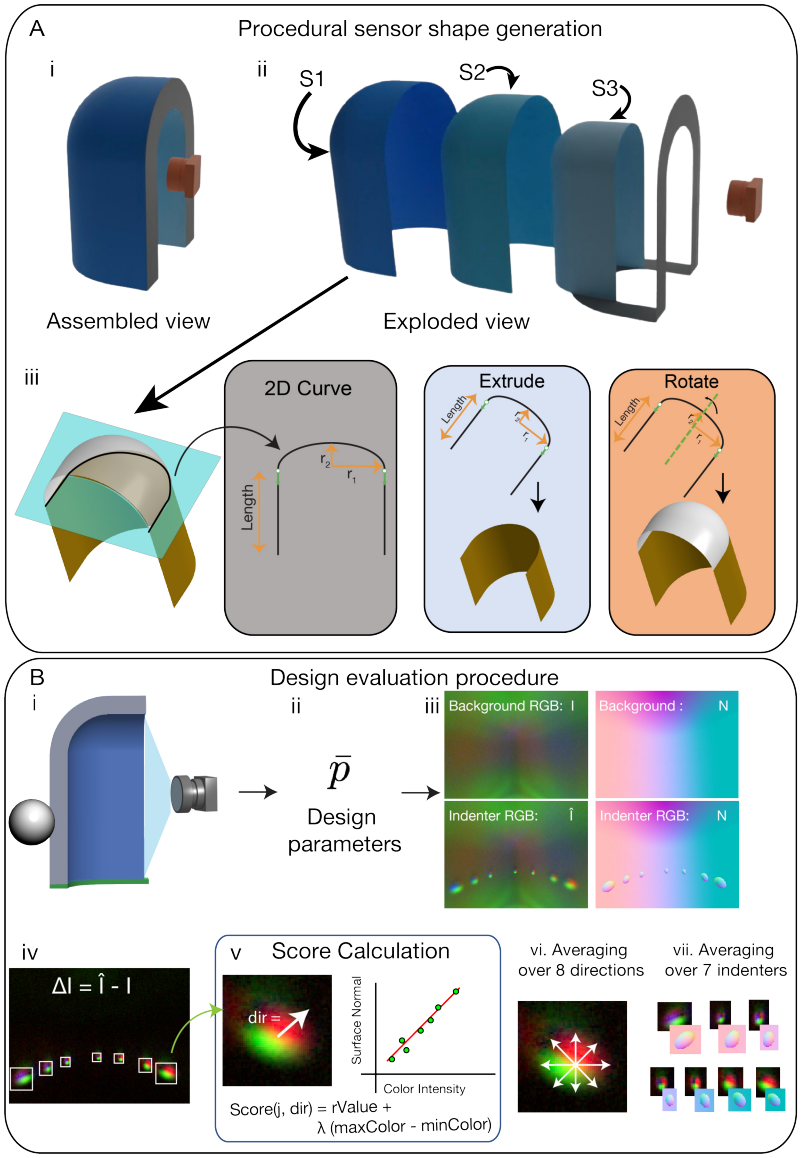



Figure 4.2: **Procedural sensor generation and design evaluation framework**. **(A)** (i) shows the assembled view of a virtual sensor, (ii) shows the exploded view of the sensor, composed of 3 surfaces—Surface 1 (S1), Surface 2 (S2), and, Surface 3 (S3)—and (iii) shows procedural sensor mesh generation using low-dimensional curve parameterization and CAD primitives. **(B)** Sensor design scoring using a new *RGB2Normal* scoring function based on tactile images of surface indentation. This scoring function correlates with 3D shape reconstruction of indentation on tactile sensors.

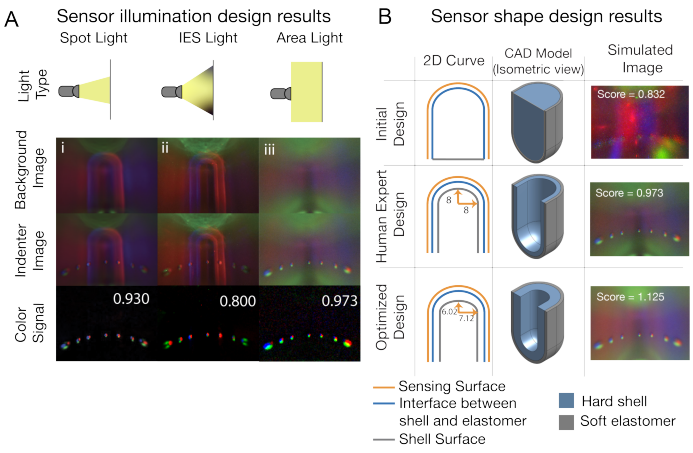



Figure 4.3: **Cuved sensor design results**: (**A**) shows the design of light type and its corresponding objective score. (**B**) shows the 2D curve parameterization for the sensor shape design. (**C**) shows the results of sensor shape optimization. We outperform the initial design and human-expert design in terms of our novel *RGB2Normal* objective score.

uniformly illuminate the sensing surface from directions, which do not lie on a plane [9] - *illumination constraint* [90, 107].

This *illumination constraint* can be challenging to satisfy if light sources are not placed appropriately. For example, DIGIT [47] was designed with the objective of miniaturization and a repeatable manufacturing process. However, for easy assembly designers placed light sources that are directly illuminating the sensing surface. This leads to two problems, as can be seen in Figure 4.3A: a) cast shadows, leading to shadow areas unusable for sensing [44] as shadows can not be directly mapped to a surface normal, b) non-uniform illumination of the sensing surface. Moreover, DIGIT offers a single flat tactile sensing surface as compared to a curved tactile sensing surface. Similarly, OmniTact[73] was designed with the objective of multi-directional sensing with a curved surface and small form factor. However, *illumination constraint* was not a consideration. Therefore, to perform perception, authors had to perform costly calibration steps using manually designed hardware setup and neural network training. Moreover, the sensor is prohibitively expensive due to the use of five endoscopic cameras.

Specifically, our design goals for the tactile sensors are as follows:

- **Accurate 3D surface reconstruction:** Ability to map RGB color information

to surface normals allows multiple types of rich contact state feedback(shape and force[64, 90]) at real-time speeds without large-scale data collection.

- **Curved Sensing surface:** Tactile sensing with a curved sensing area like a human finger is highly desirable[77]. This allows for a larger contact area without performing re-orientations of the robotic arm.

We choose our sensor design space to satisfy *curved sensing surface* design goal. To satisfy the design goal of performing 3D surface reconstruction, we designed a novel objective function, termed as **RGB2Normal** score, for rating various designs by considering indented locations. We used spherical indenters across the sensing surface as shown in Figure 3.18D.i. Algorithm 1 describes the objective function calculation. The key idea is that mapping between color and normal should be linear in background subtracted RGB images at the indented locations. Intuitively, this encodes that the sensor design generates images that will have efficient normal recovery from RGB images and will lead to better 3D shape reconstruction. The proposed metric is a good proxy for 3D surface reconstruction, as it is fast to calculate and does not require any calibration step.

## 4.3 Design Space Overview

Given a simulation framework and an efficient objective function to evaluate sensor design, we now explore a few design spaces and showcase how to improve tactile sensor designs. Specifically, we show a) illumination design by user-guided variation of the light model and b) tactile sensor shape design using low-dimensional sensor shape parameterization and using gradient-free optimization.

### 4.3.1 Illumination Design

We address the question - *what is the best illumination profile of individual light sources?* The objective of the illumination parameter design is to obtain uniform illumination on the sensing surface without any highlights or dark regions and to show a high *RGB2Normal* score.

*Light source type*: In the past sensors[77][90], light sources with lenses have been used for providing illumination inside the sensors. However, there are no

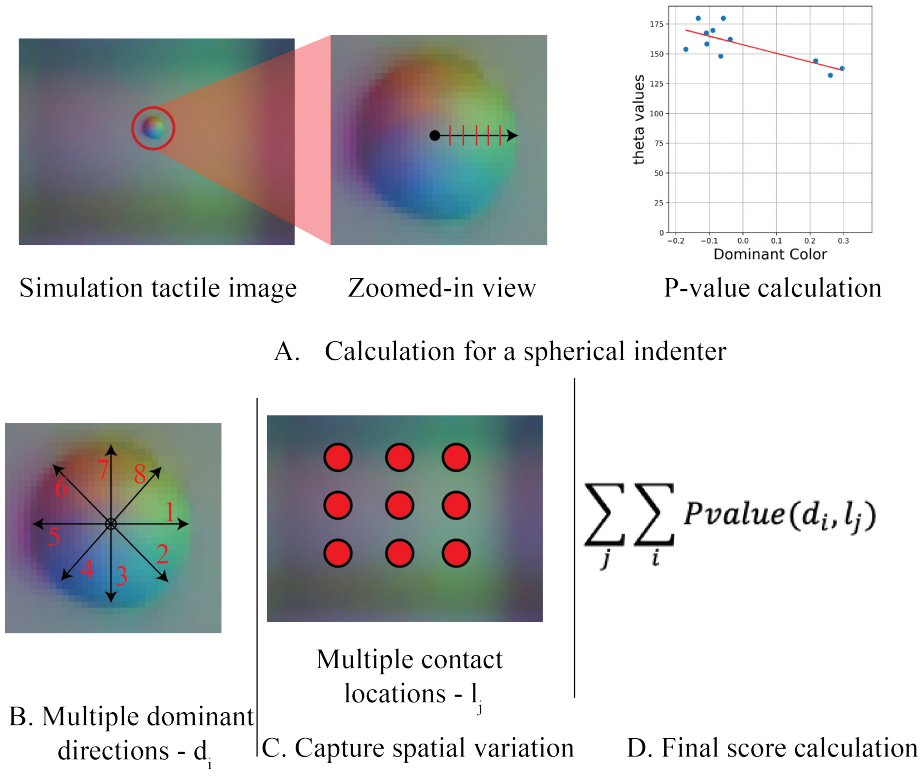



Figure 4.4: **RGB2Normal evaluation criteria method**: In **A**, we show the linearity fit calculation (P-value) for a single indenter location. We use the $\theta$ value of surface normals and dominant color for calculating the linearity score. We average the score across multiple dominant directions (**B**). To account for spatial variation in our evaluation criteria, we average the value across multiple contact locations one by one. The final calculation is given in **D**.

**Algorithm 1** RGB2Normal Score calculation

**Require:** $\bar{p}$: design parameters
$\phi$: threshold

/* *Initialization* */
Score = 0;
numValidFits = 0;
/* *Simulate background RGB image $I$ and indented RGB image $\hat{I}$.$SeeFigure2B.iii$* */
$I \leftarrow$ SimulateRGB($\bar{p}$)
$\hat{I} \leftarrow$ SimulateRGB($\bar{p}$, indenterPixelLoc)
$\Delta I \leftarrow I - \hat{I}$ ▷ *Calculate difference image*
/* *Generate background normals $N$ and indented normals $\hat{N}$. See Figure 2B.iii* */
$N \leftarrow$ SimulateNormals($\bar{p}$)
$\hat{N} \leftarrow$ SimulateNormal($\bar{p}$, indenterPixelLoc)
$\Delta N \leftarrow N - \hat{N}$ ▷ *Calculate difference in surface normals*
/* *Calculate contact mask by checking the difference in surface normal* */
Mask $\leftarrow \Delta N > 0$
/* *Use image processing to extract contact regions. See Figure 2B.iv* */
$C_1, C_2, \ldots, C_k \leftarrow$ findContours(Mask)
numContours $\leftarrow k$
/* *Fit a tight bounding box around the contact region. See Figure 2B.iv* */
**for** $i = 1, \ldots,$ numContours **do**
    $B_i =$ fitBoundingBox($C_i$)
$\theta, \phi \leftarrow$ SphericalCoordinates($N$)
/* *Extract ellipse center by maximizing the $\theta$ field over indented location* */
sphereCenter = findCenter($\theta$)
/* *Iterate over all indenters and calculate local scores. See Figure 2B.vii* */
**for** $j = 1, \ldots,$ numContours **do**
    /* *Iterate over all directions. See Figure 2B.vi* */
    **for all** $dir \in \{\leftarrow, \nwarrow, \uparrow, \nearrow, \rightarrow, \searrow, \downarrow, \swarrow\}$ **do**
        /* *Get pixel coordinates of indented locations along the chosen direction from the center* */
        (**coordX**, **coordY**) $\leftarrow$ extractPixelLocations(sphereCenter, dir, Mask, $\Delta I$, N, ch, $\phi$)
        /* *Project 3D color information to 1D using Principle Component Analysis. See Figure 2B.v* */
        ProjColorVec = PCA(RGB{**coordX**, **coordY**})
        /* *Obtain the fit parameters and goodness of line fit. See Figure 2B.v* */
        slope, rValue $\leftarrow$ lineFit(ProjColorVec, $\theta${**coordX**, **coordY**})
        /* *Add the rValue to the cumulative score if the line fit succeeded. See Figure 2B.v* */
        **if** IsValid(slope) **then**
            numValidFits $\leftarrow$ numValidFits + 1;
            Score $\leftarrow$ Score + $\{rValue + \lambda(\max(\text{ProjColorVec}) - \min(\text{ProjColorVec}))\}$;
/* *Take the mean over all line fit goodness scores* */
Score $\leftarrow$ Score/numValidFits
**return** Score

guidelines on what should be the profile of outgoing light rays. Therefore, in this section, we experiment with 3 light profiles - a)calibrated IES light source b) spot light source c) area light source; and discuss their effect on the sensor illumination. Figure 4.3B shows RGB images generated with and without sphere indentations(as shown in Figure 3.18D.i). We can clearly see that the area light source, which emits light in all directions in the positive half of the hemisphere, is the best-performing illumination design in Figure 4.3B.iii. It doesn't have bright streaks of light and has the highest *RGB2Normal* score of 0.973, as can be noted through the coloration of spheres in the last column. We choose area lights entering both hard and soft regions of the sensor for all the subsequent experiments and in the final prototype for real-world experiments.

### 4.3.2 Sensor shape design using parameterized curves

Changing the shape of a tactile sensor is a challenging problem. The shape of the sensor has a significant impact on tactile perception by modifying the path of light rays from the illumination source to the camera in a non-linear way. In previous vision-based tactile sensing works [77] [73], either the sensing surface is kept flat or chosen arbitrarily. However, there are no guiding principles to design the tactile sensor shape if the sensor form factor has to be modified due to robotic constraints. In this section, we present the first method to procedurally generate sensor designs and use the proposed *RGB2Normal* objective function to automatically generate a tactile sensor. Specifically, we set out to generate curved tactile sensors for a class of tactile sensors proposed in [77], using a low-dimensional parameterization and setup the sensor shape design as an optimization problem. Having a compact representation for shape spaces allows one to design in a controllable manner[78].

We can compose the tactile sensor from 3 curved surfaces as shown in Figure 4.2A. The key idea is to generate a curved surface from 2D curves followed by CAD primitives of *extrusion* and *rotation* about a given axis. The process is as shown in Figure 4.2A. We chose 2 ways to generate 2D curves - a)composition of ellipse arc and straight lines and b)collection of cubic B-splines. Both methods allow modeling sensor shapes (geometries) that are $\mathcal{C}_1$ continuous and can be manufactured using accessible fabrication techniques.

Formally, the sensor shape is given by 3 curved surfaces(shown in blue in Figure 4.2B) - $S_1(x)$, $S_2(x)$ and $S_3(x)$ , each of which is generated with a corresponding 2D curve $C_1(x)$, $C_2(x)$ and $C_3(x)$.

With *ellipse parameterization*, the curve $C_1$ is composed of a half-ellipse with parameters $r_1^d$, $r_2^d$ and add straight line segments with length $h$ on both sides symmetrically, as shown in Figure 4.3C.i.

With *cubic B-splines* parameterization, the curve $C_1$ is composed of a group of cubic B-splines passing through control points. For the curved sensor family used in this paper, we generate a cubic B-Spline $C_1$, by placing 7 control points with locations $\{(p_1, 0)$, $(p_2, p_3)$, $(p5, p6)$, $(0, p4)$, $(-p5, p6)$, $(-p_2, p_3)$, $(-p_1, 0)\}$, as shown in Figure 4.3C.ii.

The 2D curve is then used to generate the outermost sensing surface $S_1$ by extruding for length $e$ along the $z - axis$(as shown in Figure 4.2A.ii) and the right half of the curve $C_1$ is rotated about $y - axis$ for $180^o$ to obtain the top curved part of the sensor(as shown in Figure 4.2A.iii). The process for generating the surface $C_2$ and $C_3$ is similar with appropriate parameter values.

In all designs, we place the camera at the origin and place light sources along the periphery of the shells, as shown in Figure 4.2A.iv. We use the extrusion length $e = 28$mm in all the shapes in this paper.

**Innermost surface optimization**: We use the above parameterization to formulate the problem of the refractive epoxy surface design. This allows us to change the optical properties of the sensor while keeping the mechanical tactile response the same. We optimize for the innermost surface geometry$S_3$ while keeping the $S_2$ and $S_1$ the same. Formally, the optimization problem with ellipse sensor parameterization is defined as follows.

$$\underset{r_1^e, r_2^e}{\operatorname{argmin}} \textit{RGB2Normal}\ (\text{generateSensor}(r_1^e, r_2^e)) \tag{4.1}$$

For our experiments, we used a gradient-free evolutionary algorithm, CMAES[34], for optimization of the refractive surface geometry. We consider 2 baselines, an initial design that consists of a flat plane surface instead of a shell surface and a human-expert design with parameters $r_1^e = 8$ and $r_2^e = 8$. The radius range is manually chosen as $(6, 10)$. The smallest radius was constrained by the camera

dimensions and the largest radius was constrained by the soft-sensing PDMS volume. Our optimization resulted in the design with parameters $r_1^e = 7.12$ $r_2^e = 6.02$. Figure 4.3C.ii shows the comparison of our optimized design with the initial design and a human-expert design[77].

### 3D surface reconstruction evaluation in simulation

A key benefit of vision-based tactile sensors is their ability to provide high-resolution 3D shape information about the contact surface. Local 3D geometry of the object is useful for various robotic tasks, e.g., object pose estimation[10, 83], force distribution estimation [64], and object regrasping [11]. We highlight the utility of our sensor shape design procedure in this task in simulation.

Surface reconstruction requires 2 steps: (a) creating a mapping from color to surface normals; (b) integrating surface normals into surface depth information to recover the point cloud. For calibration, we rendered a single image with 5mm spheres indenting the sensor surface and fitted a tiny Neural Network to predict surface normal from color information. We used perspective Poisson integration to recover the surface depth from the predicted surface normal field.

Figure 4.5 compares the performance between the human-expert design and the optimized sensor design for surface depth recovery. We indent the sensing surface at multiple locations on the whole sensing surface. We notice that both designs can perform well on surface normal recovery. However, the surface normal error map shows that the human-expert design has a high value at non-edge pixels. However, the optimized design has large surface normal errors only on the edges. The probable reason for the high surface normal recovery error for both designs on edge pixels is the sudden change in color values and less training data for edge pixels.

Due to a better surface normal distribution in the optimized design, the surface depth reconstructions are significantly better than the human-expert design. We obtain $26.65\%$ error on *projected depth error metric* as compared to 34.47% error by the human-expert design on 3 different shapes - $5\,\mathrm{mm}$ sphere, natural texture(texture 1), and M4 screw. Figure 4.5 provides the description for all the shapes at multiple sensing locations.

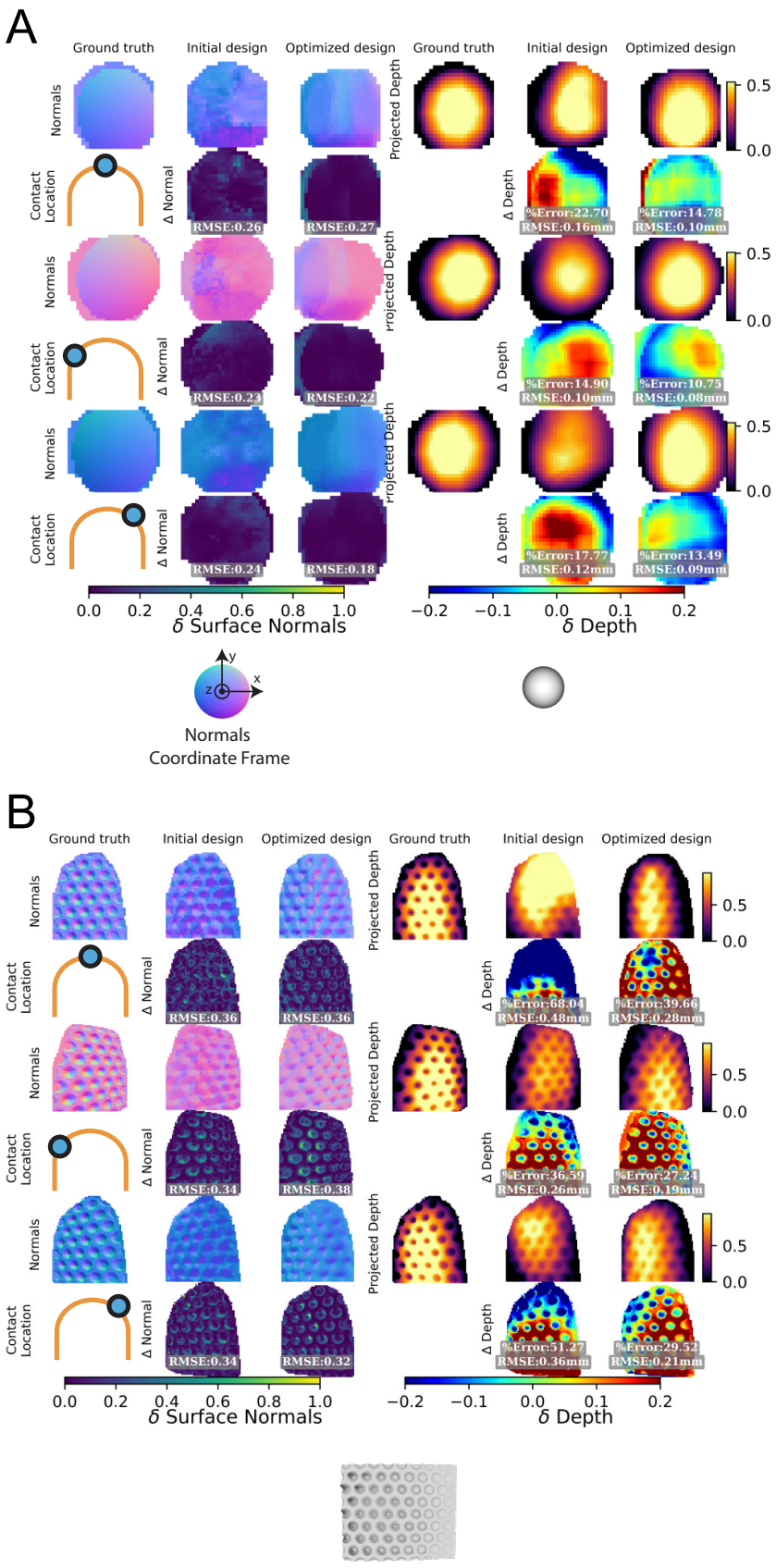





Figure 4.5: **3D surface reconstruction in simulation**: (**A**) and (**B**) shows the result of 3D surface reconstruction for a sphere indenter and a texture indenter.

### 4.3.3 3D surface reconstruction evaluation using manufactured prototypes

We fabricated the designs discussed above to test on real objects. For details on fabrication, please refer to Appendix D. We compare the performance of the two designs qualitatively on surface reconstruction and quantitatively on normal recovery on different parts of the sensing surface. For qualitative comparison, we used 3 objects - icosahedron, cloth texture, and US quarter coin. Figure 4.6 shows the RGB images, the predicted surface normals, and the reconstructed surface depth. For the icosahedron shape, the human-expert design has uneven background due to saturation as compared to the optimized design, there is no error in the background. For the cloth texture, the fine-grained normals are hard to recover due to color saturation along the bright regions. However, optimized design is able to faithfully recover the cloth texture with subtle changes in the normals. For the US quarter coin, we can clearly see the text in the optimized design as compared to the human-expert design. Therefore, qualitatively our optimized sensor design performs significantly better than the human-expert design.

For quantitative surface normal recovery comparisons, we plot predicted surface normal vs. ground truth surface normals at the different parts of the sensing surface, as shown in Figure 4.6.

## 4.4 Characterization of the parameter space

In this section, we use our framework to understand the effect of changing design parameters on the *RGB2Normal* objective function and sensor perception. To this end, we perform simulation while varying - a) the Thickness of the hard epoxy shell, soft PDMS volume, and coating material of the sensing surface; b) refractive indices(IOR) of the Epoxy surface and the interface between soft PMDS volume and hard epoxy shell(as shown in Figure 4.2B). These experiments allow us to derive high-level guidance on designing vision-based tactile sensors.

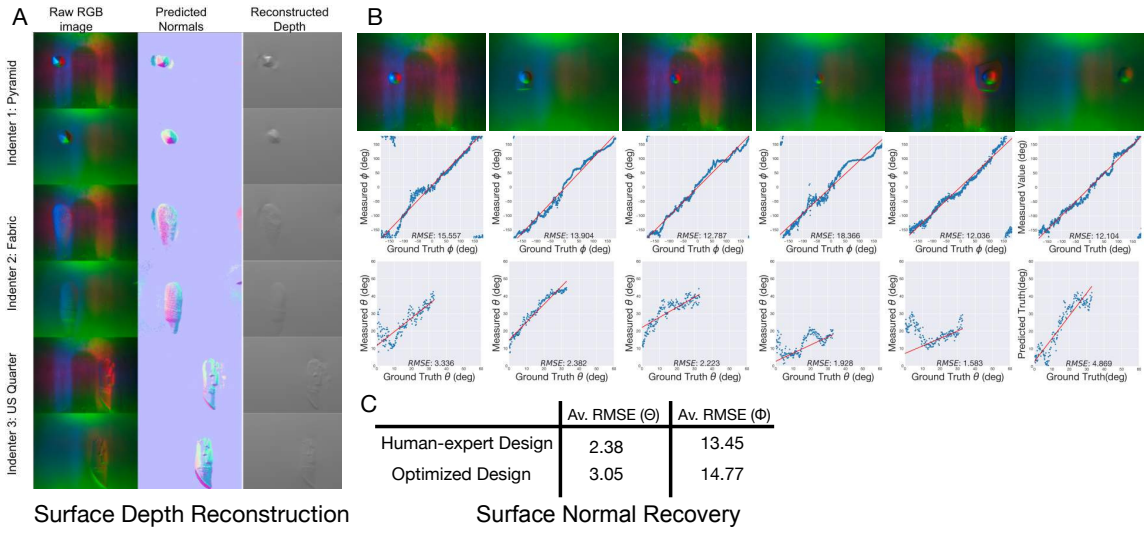



Figure 4.6: **Real world results 3D reconstruction results**: (**A**) shows the qualitative results for human-expert design and optimized sensor design. (**B**) shows the surface normal recovery ability of human-expert design and optimized sensor design. The optimized sensor design is able to recover the surface normal very well on the off-center locations.

### 4.4.1 Effect of changing sensor thickness and surface coating material

In this section, we vary the thickness of hard epoxy shell($t_1$) and soft PDMS volume($t_2$) for 6 classes of coating materials. The thickness parameters $(t_1, t_2)$ is chosen from $[1, 3] \times [1, 3]$ with step-size of 0.5. In all the cases, we kept the sensing surface profile to be fixed to a 2D curve generated using *ellipse curve parameterization* with $r_1 = 14.5$ and $r_2 = 14.5$. These parameters were chosen to be similar to a human finger and be able to accommodate a Fish-eye lens camera. We use the extrusion length $e = 28$mm in all the shapes. All the dimensions are in millimeters. Figure 4.7B shows the regions affected by the thickness parameters. Figure 4.7A.ii shows the effect of varying $(t_1, t_2)$ with a material coating which is completely diffuse(as used in [107]). Similarly, all the subfigures compare designs with a specific coating material specularity for a range of thickness values. By comparing all the subfigures, it is apparent that for a curved tactile sensor higher specularity offers higher sensing performance. Figure 4.7C.ii and Figure 4.7C.v specifically

show the performance of the sensors created using real material properties, diffuse powder, and semi-specular metal powder, respectively. In the bottom row, we can visually inspect the designs by using the simulated RGB images when they are indented by a group of spheres. Note, that our *RGB2Normal* objective function correctly identifies designs with better qualitative perception by showing higher objective function value.

### 4.4.2 Effect of IOR

: For this experiment, we analyze the effect of changing the Index of Refraction(IOR) of Hard Epoxy shell-Air interface $\eta_{\text{epoxy}}$(Figure 4.2B *Epoxy Surface*) and soft PDMS volume-hard Epoxy shell interface$\eta_{\text{PDMS}}$(Figure 4.2A *Interface Surface*). The region is also highlighted in Figure 4.7A. The IOR affects the direction and intensity of light as it crosses the surface boundary between different materials. This experiment is prohibitively expensive to perform in the real world. Generating transparent materials with specific IOR requires experimenting with mixing coefficients in the lab, which is very tedious. Thus, we modify the IOR pair and assess the sensing performance in simulation. We vary the $\eta_{\text{PDMS}}$ from $[1.4, 1.5]$ and $\eta_{\text{epoxy}}$ from $[1.5, 1.6]$. The ranges represent the materials that we can physically manufacture in our lab. We initialize the sensor shape with surfaces generated using 2D curve generated using *ellipse curve parameterization* with $r_1^d = 14.5$, $r_2^d = 14.5$, $t_1 = 4.4$, $t_2 = 1.69$ and $e = 28$. The dimensions were chosen to match the physical prototype we manufactured in the lab. All the dimensions are in millimeters. Figure 4.7B shows the *RGB2Normal* objective function when IOR is varied in the above range. We identify that for $\eta_{\text{PDMS}} = 1.46$ and $\eta_{\text{epoxy}} \in [1.58, 1.6]$ lead to high *RGB2Normal* objective value = 0.81, for the chosen shape. The key observation is that the sensing performance is low when the refractive indices, $\eta_{\text{PDMS}}$ and $\eta_{\text{epoxy}}$ are close to each other, according to Figure 4.7B rows $\eta_{\text{PDMS}} = \eta_{\text{epoxy}}$.

## 4.5 Robotics Applications

To demonstrate the potential of our optimized design, we show 2 robotic tasks by integrating our sensor on a robotic arm. The first task shows the ability to perceive

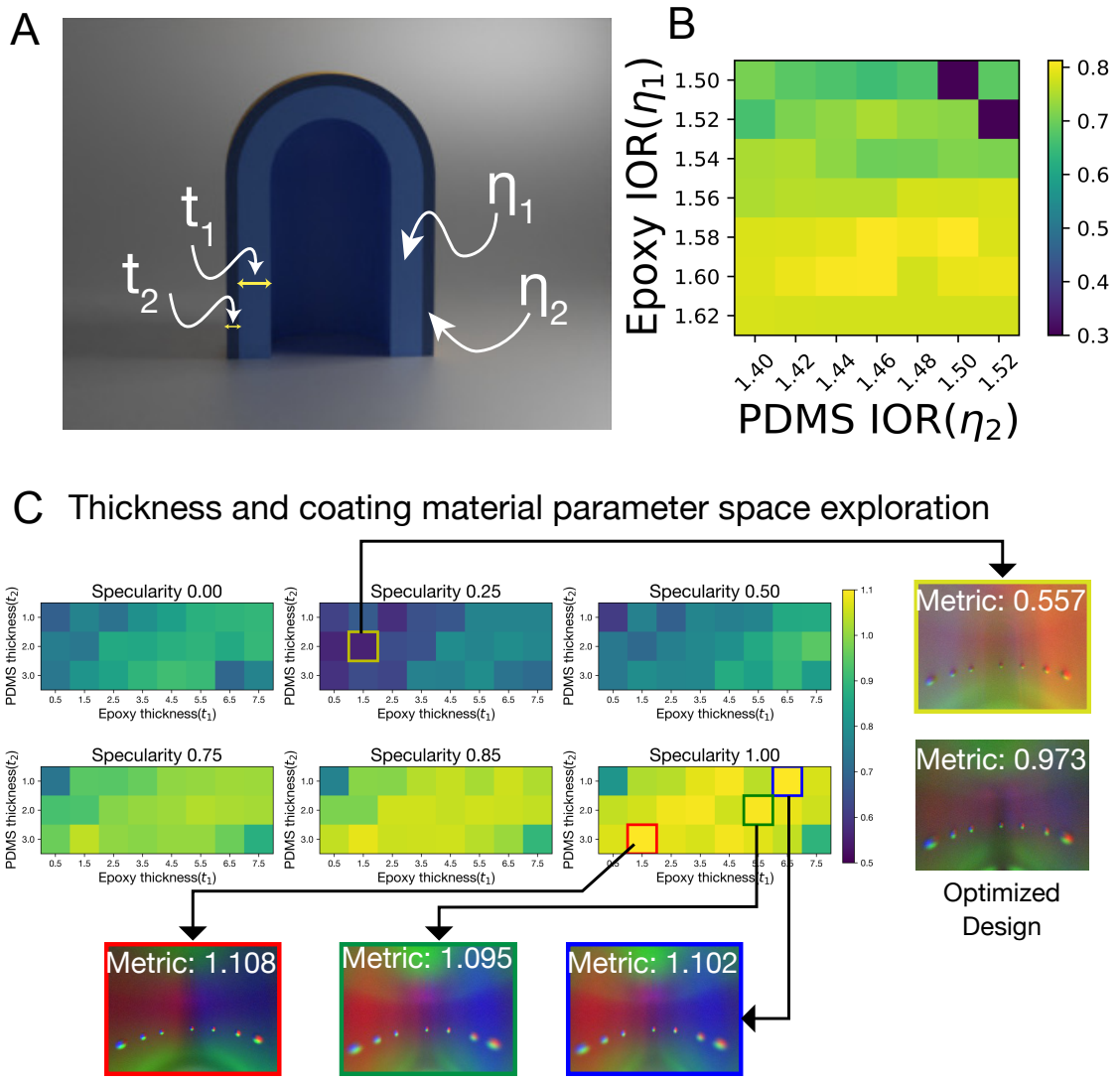



Figure 4.7: **Parameter Space Exploration**: (**A**) shows the sensor illustration with important parameters which we consider for exploration. For parameter space visual, we plot the *RGB2Normal* objective score for a given design. (**B**) plots the variation of refractive indices $\eta_1$ and $\eta_2$ for hard shell and soft elastomer region, respectively. (**C**) plots the variation of elastomer thickness and hard shell thickness for 6 sensing surface coating materials. We also show the tactile image with a surface spherical surface indentation for some cases to qualitatively compare the designs.

and grasp various daily-use objects from the YCB dataset. The second task shows the utility of a curved sensing surface in automatic robotic surface inspection. In all the cases, we compare 2 designs - human expert design and optimized design. The human expert design is referred to as *Design A* in the following text.

### 4.5.1 Robotic Grasping experiment

In this section, we show the grasping capability of the optimized sensor. Figure 4.8 shows the sensor mounted on a 5-degree-of-freedom robotic arm and Weiss parallel-jaw gripper. We grasp 3 objects from YCB[12] dataset - a Yellow Mustard bottle, a Lego piece, and a door lock key. These objects were chosen specifically to highlight that even with a limited degree of freedom in the parallel jaw gripper, our tactile sensor can sense and grasp these objects due to the curved sensing surface.

### 4.5.2 Robotic Surface Inspection

In this section, we perform the robotic surface inspection using our robotic setup. Surface inspection of industrial parts is a fast-growing market [68]. Combining vision and touch is the state-of-the-art approach[75] for surface inspection in high-precision and extreme environments (e.g. in nuclear plants). In this experiment, we press the tactile sensor on a 3D-printed surface that contain text bumps. We use Google Image Recognition for identifying the text from RGB images captured by each sensor.

Figure 4.9A shows the experiment setup which consists of 5 DoF robot arms, a parallel jaw Weiss gripper, and our manufactured tactile sensor prototype. Figure 4.9C shows the experiment when specimens containing 3 different text heights - $1.5\,\mathrm{mm}$, $1.25\,\mathrm{mm}$, and $1.0\,\mathrm{mm}$ are inspected by tactile sensors. *Design A* sensor's ability decreases to identify the text as the text height becomes goes from $1.5\,\mathrm{mm}$ to $1.0\,\mathrm{mm}$. However, the optimized sensor is able to identify most of the text in all cases. We also tested the text recognition ability at different parts of the sensing surface. For this experiment, we chose the text height of $1.0\,\mathrm{mm}$. *Design A* completely fails to recognize text, if the text appears vertically due to bad illumination design. However, the optimized sensor is able to identify most of the text at all the

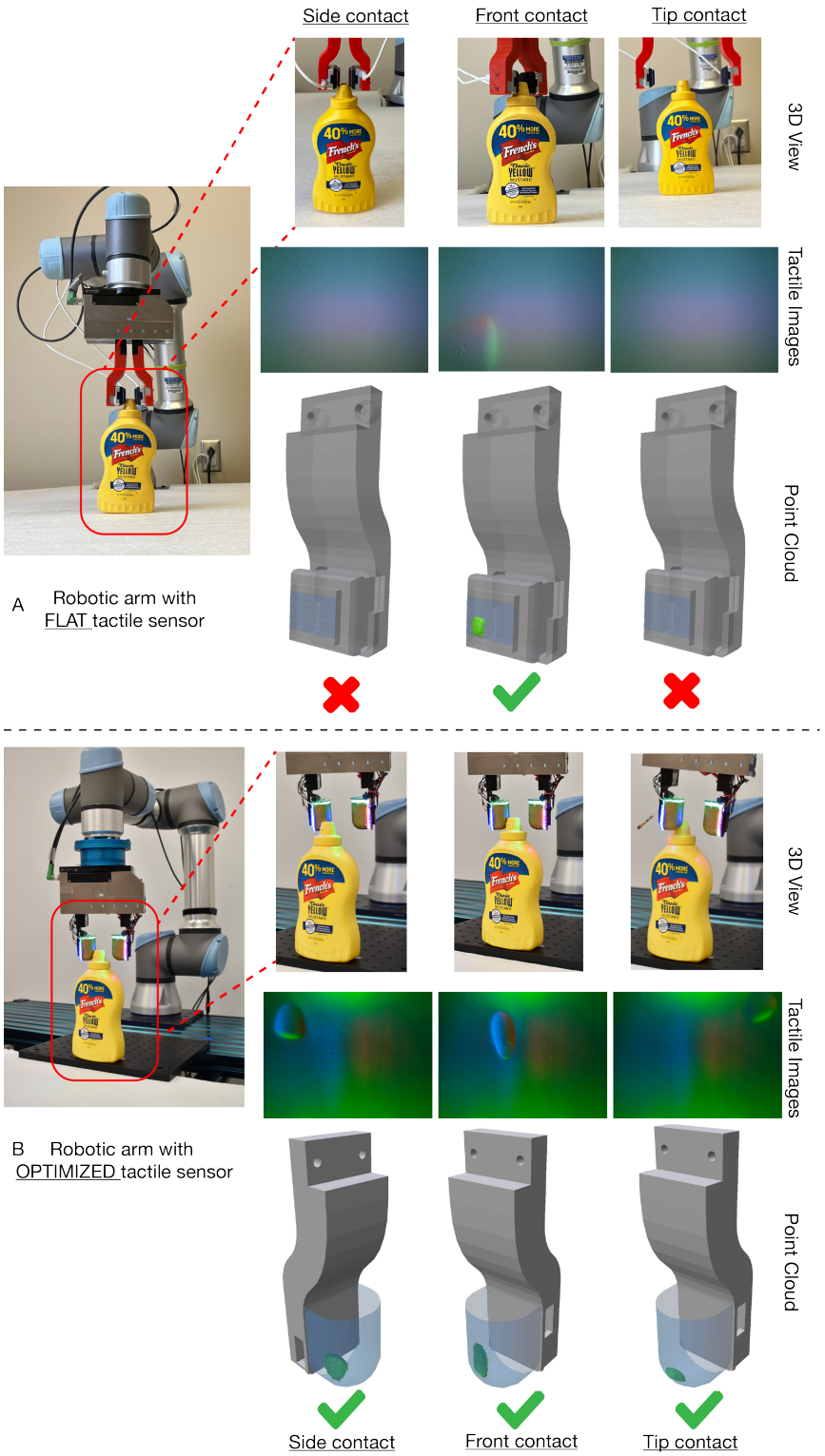



Figure 4.8: **Robotic grasping**: Due to the curved tactile sensor, we can manipulate objects without reorienting the robotic arm. In this visual, we show the contact signal when the robot, endowed by a tactile sensor, approaches objects from three directions - front, side, and tip. In each part, we show the robot view, tactile image, and 3D reconstruction for various approach directions. **Part A** shows the use of GelSight Mini with a flat sensing surface. It is only able to perceive contact when the robot approaches from the front direction. **Part B** shows our optimized sensor giving a high-resolution contact signal from all robot approach directions.

locations as shown in Figure 4.9D column 2. Therefore, our optimized sensor is able to recognize small artifacts across the curved sensing surface.

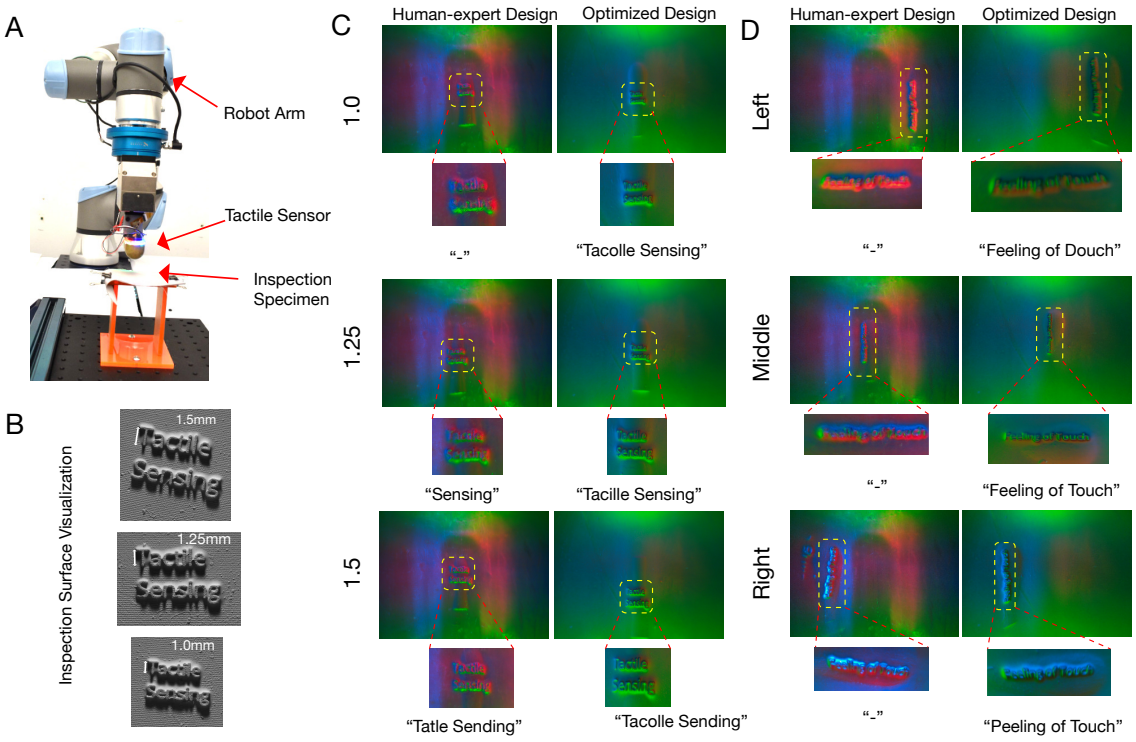



Figure 4.9: **Robotic surface inspection**: (**A**) shows the robotic setup for surface inspection with a planar inspection specimen placed under the sensor. In (**B**) we show the zoomed-in view of the text under inspection. In (**C**), we compare the performance between human-expert design and optimized design for surface inspection of various text sizes - $1.0\,\mathrm{mm}$, $1.25\,\mathrm{mm}$ and $1.5\,\mathrm{mm}$. Clearly, optimized can recognize the text very well in all the cases, as shown in the zoomed-in view and recognized text below each image. In (**D**), we compare the 2 sensor designs, by indenting the sensor at all parts of the curved sensing surface and in various orientations. Only the optimized sensor is able to recognize any texts when the indentation is vertical and at non-central locations.

# Chapter 5

# Objective functions for sensor performance quantification

There has been a proliferation of VBTS designs aimed at robotics in the last decade. However, it is unclear how to assess the performance of the sensor without testing in the specific task. Researchers have shown improved performance with their sensor design in specific robotic tasks, for which data on other sensors may not be available. Therefore, there is a need to develop objective functions that can be uniformly applied to all the sensor that have a common working principle, for example, GelSight-like sensors.

We propose objective functions based on our experience in GelSight-like tactile sensor design and applications. An objective function tries to encode a design goal while being computationally efficient to calculate. In our experience, the objective function design is a challenging problem because of the two reasons: (1) the ultimate goal depends on the specific use-case of the tactile sensor and (2) it is unclear which modality (image, depth or contact area) is the most relevant for robotic manipulation.

We introduced an objective function in Section 4.2 to characterize the quality of the mapping between color (RGB) and surface normal. In this chapter, we introduce two new objective functions, *NormDiff* and *As-orthographic-as-possible* (AOAP), that account for surface reconstruction quality in the presence camera noise and optical distortion due to various optical elements. These objective functions give a single

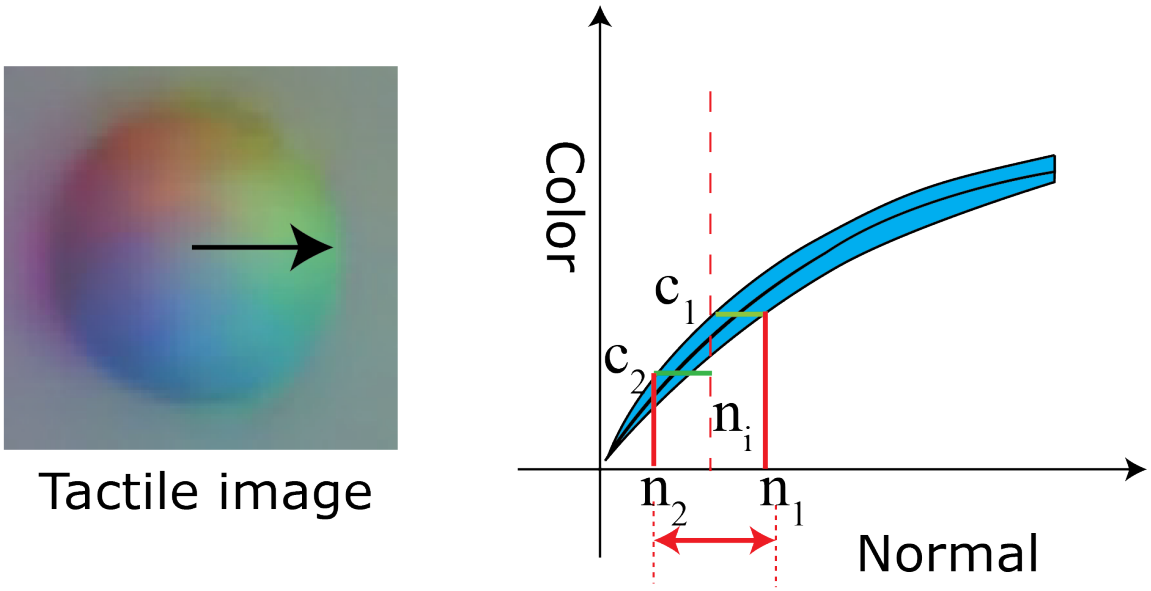



Figure 5.1: **NormDiff objective function:** (A) shows the tactile image with an indentation; (B) shows the canonical example of color-normal plot. For a chosen normal, $n_i$, the color noise between $[c_2, c_1]$. This leads to confusion range in normal to be $[n1 - n2]$.

number for each design and allow us to rate the various designs.

## 5.1 NormDiff objective function

We introduce the *NormDiff* objective function as an alternative function to evaluate the sensor's capability to measure 3D shapes. The motivation of the function design is the same as the RBG2Normal function design introduced in Section 4.2, but here we do not use the constraint that the RGB vector is expected to be linear to the surface normal value. Instead, we expect that the RGB value corresponding to a specific surface normal vector should be very distinct from the ones of other surface normal values. The level of "distinctiveness" is denoted by the measurement uncertainty, and we calculate it based on the prominent camera noise models [74]. In this model, the RGB noise is proportional to the sensor response or the RGB value at each pixel.

For an intuitive explanation with a canonical illustration, please refer to Fig-

ure 5.1. In the canonical example, the color and normal are both represented as 1D quantities. The color-normal curve along with standard deviation is shown in Figure 5.1B. At normal value $n_i$ the color values vary between $c_1$ and $c_2$. For color value $c_1$ and $c_2$, the range of normal values is $[n_i, n_1]$ and $[n_2, n_i]$ respectively. Therefore, at normal value $n_i$, the confusion in estimating normal is $(n_1 - n_2)$ and *NormDiff* objective function value for an indenter is the negative of the confusion, $-(n_1 - n_2)$.

In the original implementation, we use the lookup table (LUT) to create a mapping between the RGB color values and $(\theta, \phi)$ surface normal coordinates, as this is a common practice in GelSight sensors [107]. To calculate the confusion value per indenter, we follow these steps.

1. We identify indented pixels by using the difference between surface normals before and after indentation. For each pixel, we record the $(r_n, g_n, b_n, \theta_n, \phi_n)$ tuple to create a dataset.
2. To assess the recovery quality for a surface normal, $(\theta_i, \phi_i)$, we find the nearest tuple in our dataset, $\mathcal{R}_1 = \{p_j\}, j = 1(1)U$, among the indented pixels.
3. For each data-point, $p_j$, we have the corresponding tuple, $(r_j, g_j, b_j, \theta_j, \phi_j)$. In the color space, we obtain the range of color values by adding noise to the RGB value, $(r_j, g_j, b_j)$. In our implementation, we added noise that was $30\,\%$ of the RGB value. We find all the data points, $\mathcal{N}_1 = \{q_k\}, k = 1(1)N$, in the color space within this RGB range.
4. Each data-point, $q_k$, has its corresponding normal value, $(\theta_k, \phi_k)$. We calculate the maximum and minimum values of the $\theta$ and $\phi$ values between all the points in $\mathcal{N}_1$. The confusion for each data-point, $p_j$, is then the weighted sum of the range $\theta$ and the range $\phi$.
5. We take an average across all the data-point $(p_j)$ to obtain the confusion to recover surface normal, $(\theta_i, \phi_i)$.
6. We repeat Step 2-5 for other $(\theta_i, \phi_i)$ pairs.
7. The final value of the objective function is negative of the average of the confusion from the previous step.

## 5.2 As-orthographic-as-possible objective function

GelSight-like tactile sensors are known to capture the high-resolution surface geometry of the indenting surface. However, the presence of refractive and reflective optical elements can distort the sensing surface view. We introduce an objective function to measure this distortion systematically. The key idea of this objective function is that angle of incident of rays when they reach sensing surface should be as close to zero as possible. If the sensing surface is a plane, then this condition would make the rays as it they were shot from a orthographic camera. Therefore, we name this objective function, *as-orthographic-as-possible* (AOAP). We also add a regularizer term in this objective function to encourage sensing surface coverage to avoid mode collapse. The calculation procedure is as follows:

1. Shoot rays from the cameras and perform ideal refraction and reflection on surfaces until the rays hit the sensing surface or escape the sensor. Record the hit position, hit triangle face index and incidence angle of all the rays.
2. Find unique sensing surface triangle faces, $U$, hit by all the camera rays. The total number of triangle faces in sensing surface is given by $T$

The final objective function is as given below

$$\mathcal{O}_3 = \frac{1}{N}\sum_i \mathbf{n}_i \cdot \omega_i + k_1 \frac{U}{T}$$

, where $k_1 = 0.01$ and N is equal to the number of pixels in our experiments.

## 5.3 Discussion

We provide a starting point for the sensor designers to optimize design parameters using our objective functions. Our objective functions capture perceptual quality of surface normal encoding without (Section 4.2) and with sensor noise (Section 5.1 and geometric quality by a measure of distortion Section 5.2. Some other design goals that might be relevant for the new design of objectives include sensing surface coverage and manufacturing constraints. We expect that this will provide some guidance on how to design objective functions that are good proxies for ultimate

tactile application.

# Chapter 6

# Modular and interactive design framework

In this chapter, we describe our modular and objective-driven design framework for GelSight-like tactile sensors. In Section 6.1, we describe sensor modeling through our modular pipeline and introduce key modules used to generate initial sensor design. In Section 6.2, we describe the optical component parameterization that allows for easy customization and optimization for improving sensor performance.

## 6.1 GelSight sensor modularization

We decompose sensor modeling into five parts: *Soft elastomer*, *Support structure*, *Opaque coating*, *Light*, and *Camera*. For each part, we introduce a design module to create and optimize the corresponding part. Each part can be either modeled from scratch or initialized from our component library. We distill common optical components based on camera-based sensor literature into a component library (see Appendix B). This enables novice users to model sensors without any experience with VBTS sensors.

Figure 6.1 shows an illustration of how a GelSight Mini tactile sensor could be decomposed into real-world components. These components are analogous to the parts (modules) in our design framework. These modules can then be used to create a digital design that can be subsequently optimized.

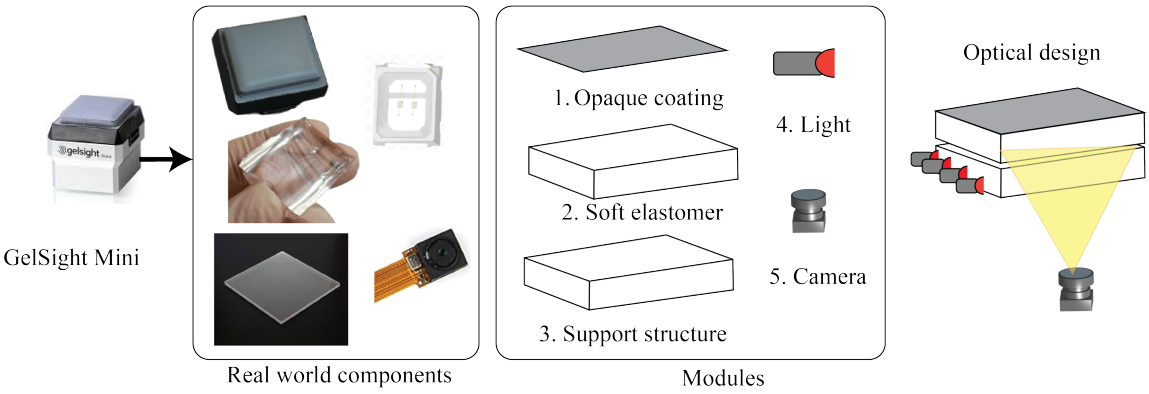



Figure 6.1: **Sensor modularization**: This figure illustrates how a tactile sensor can be modularized into our proposed modules. These modules can then be used to create a digital design for further optimization.

The key elements of each module are the choice of reference surface for the shape description, the selection of optical materials, and the choice of discrete elements from the component library. For a short tutorial on the modules, see Appendix C.

## 6.2 Design component parameterization

Our design modules automatically parameterize various sensor components to allow editing and automatic parameter selection. In this section, we describe the parameterization of the key components. The choice of parameterization was made on the basis of the authors' expertise in VBTS tactile sensor design.

**Optical component shape.** All the components in the sensor are represented by triangle meshes. The number of triangles in a mesh can be arbitrarily large and not amenable for optimization. Therefore, inspired by [103], we apply cage-based deformation to parameterize the optical component, as shown in Figure 6.2. We automatically generate a cuboidal cage with 27 cage vertices such that the cage completely encloses the component. We can increase the resolution of the cage interactively if more precise control is desired. The cage-based representation can be applied to any surface mesh irrespective of how it was generated (B-Rep representation) and bounds the dimensionality of the optimization problem for automatically choosing component shape.

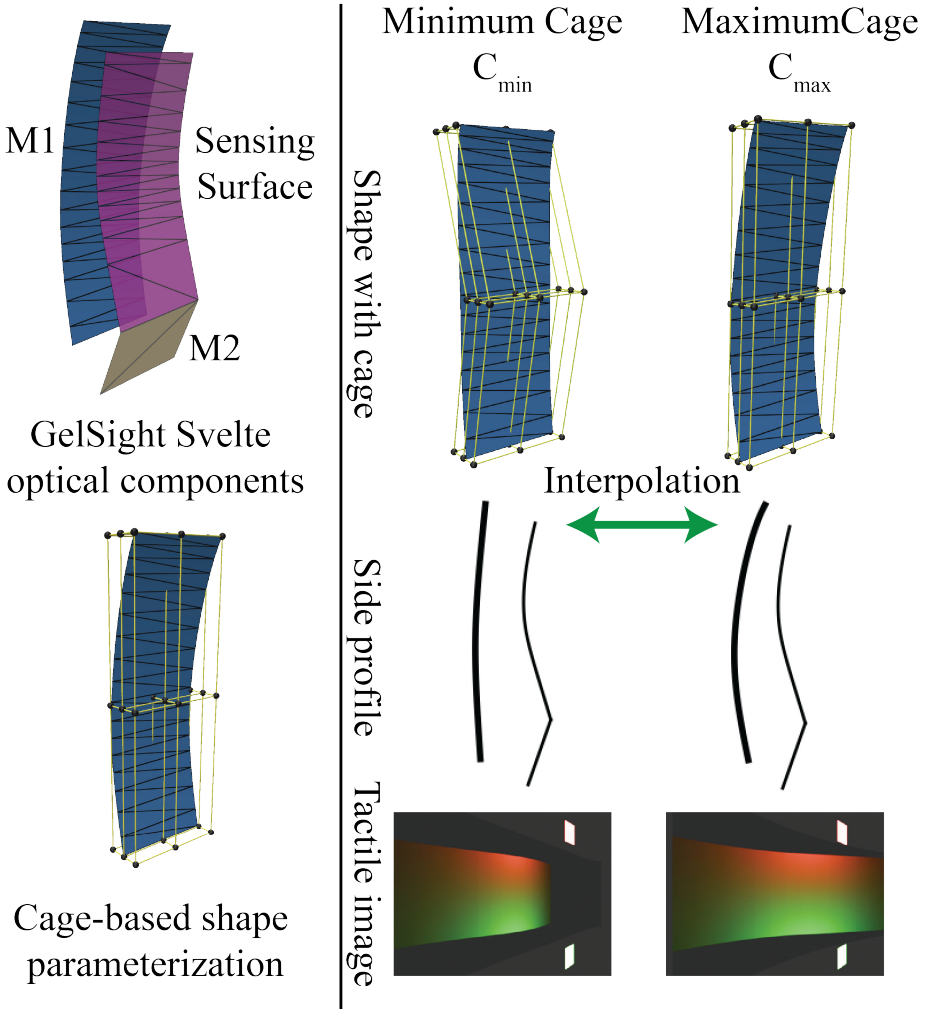



Figure 6.2: **Cage-based shape parameterization**: The left column shows the GelSight Svelte tactile sensor optical components and cage-based shape representation of mirror element, *M1*. The right column shows the user input $\mathcal{C}_{\text{min}}$ and $\mathcal{C}_{\text{max}}$. We show the deformed surface, 2D profile and the corresponding tactile image to qualitatively represent the change in tactile signal by changing *M1* mirror element in the sensor. Therefore, shape optimization of *M1* is critical to obtaining the best sensing performance.

For optimization, we require users to specify the minimum ($\mathcal{C}_{\text{min}}$) and the maximum cage ($\mathcal{C}_{\text{max}}$) parameters for the component shape. Then the current shape is given as

$$\mathcal{C}_{\text{current}} = (1 - \mathrm{A}) \cdot \mathcal{C}_{\text{min}} + \mathrm{A} \cdot \mathcal{C}_{\text{max}}$$

, where each element in $\mathrm{A}$, $\mathrm{A}_i \in [0, 1]$ is of the same dimension as $\mathcal{C}_{\text{current}}$. In shape optimization experiments, we choose $\mathcal{C}_{\text{current}} \in \mathrm{R}^{81}$.

**Optical material.** Generally, optical material properties are characterized by a Bidirectional scattering distribution function (BSDF), a 4-D function parameterized by incoming and outgoing directions. However, specifying or measuring a full BSDF model is extremely challenging [24]. Therefore, we leverage physically motivated *analytic* BSDF models proposed in the computer graphics community [41] for modeling the camera-based tactile sensors. The relevant optical material models are *RoughDielectric*, *RoughConductor*, and *Diffuse*. We performed optical experiments to calibrate these models for various surface materials (sensing surface coating and transparent surfaces) in real sensors. For a detailed description of the optical materials see Appendix B.

*RoughDielectric* can be used for all the refractive and transparent surfaces in the sensor like elastomer and clear support structure. We use the *RoughConductor* model for representing the sensing surface opaque coating material. This model has seven relevant properties: RGB reflectance (3D), refractive index ($\eta$) (3D), and specularity $\rho$ (1D). We can synthesize all the relevant coatings used in the GelSight family by varying the 1D **specularity** property of this model. Therefore, for material optimization, we use *RoughConductor* optical model and vary the specularity ($\rho$) value from $[0, 1]$ to obtain the best coating material for the specific sensor design.

**Light source.** For light sources, we use modified *PointLight* and *AreaLight* model from Mitsuba [40]. Specifically, for lights that have a spherical lens, we leverage the IES light profile provided by the manufacturer and add it to *PointLight* to scale the intensity value along a specific outgoing direction. For LEDs with a flat lens, we use the dimensions provided by the manufacturer and scale the *AreaLight* accordingly to provide an approximate model. For a detailed description of the light sources see Appendix B.

For light design, we modify the *location* and *orientation* of the light group us-

ing a forward design approach. We group lights based on their color. For light optimization, we vary the **light type** for all lights in the design. Specifically, we consider 3 types of variations: a) single light type (area vs. point) for all the lights in the sensor model (number of parameters=1); b) single light type for all the lights in a group (number of parameters=number of light groups; generally number of light groups=3); c) change light type of each light individually (number of parameters=number of lights in the sensor; can vary from 15 to 30).

**Camera.** We choose the perspective camera model in all designs. We allow varying three parameters: height, width, and field-of-view (FoV) of the camera. For a detailed description of the available cameras see Appendix B. For optimization, we iterate through the list of available cameras in our library and set the corresponding parameters to evaluate the sensor design.

## 6.3 OptiSense studio: design toolbox for digital camera-based sensors

We implement our design framework in a simulation-driven design toolbox. We describe the design interface in Section 6.3.1. Following the description, we give the design guideline using the key steps followed to generate sensor designs in Section 6.3.2 and describe specific design space parameterization available for optimization in the toolbox in Section 6.3.4. The whole framework is shown in Figure 6.3.

### 6.3.1 Design interface

Figure 6.4 shows an overview of the digital design interface. Our design environment is built on top of Blender (version 4.1.0) [15] using its Python API for scripting. The relevant elements of the interface are a 3D viewport to visualize the design in 3D; a collection panel to group components used for modeling and optical simulation; design component modules, a simulation module, and an optimization module in the add-on panel.

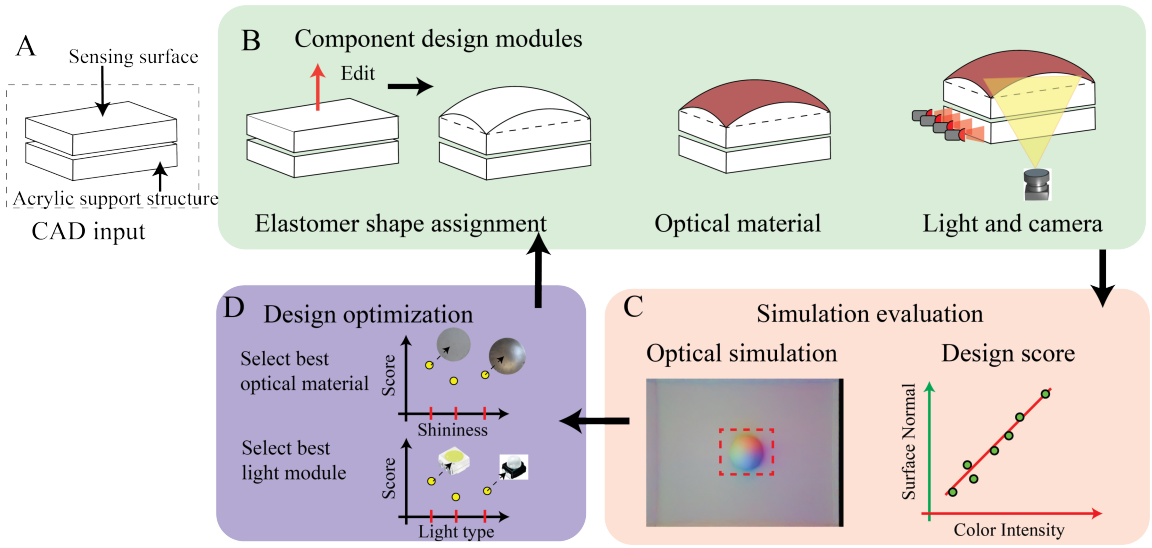



Figure 6.3: **Modular sensor design framework for novice users**: Given the user shape input in A, we model the sensor design with multiple modules in simulation as shown in B. We then evaluate the sensor performance based on the simulated indentation test in C. This is then coupled with optimization methods to choose the optimal light module and optical coating material for the sensor design.

### 6.3.2 Modeling the sensor

We modularize the design procedure into three steps and provide a library of optical components to aid in design. The user workflow consists of the following steps (see Figure 6.5).

The user starts with a sensor design idea or starts from a previous design. The user provides sensing surface geometries, initial light location, and camera location to the *OptiSense Studio*. Thereafter the users follow the given steps to generate an optical sensor design which can be simulated and perform design optimization in our software. For a brief tutorial, please refer to appendix Appendix C

**Step 1**: Setting reference geometry

Users can create sensors of arbitrary geometry by importing shape reference geometry as *.obj* surface meshes. We found that users prefer to choose their favorite CAD tools (Solidworks, Autodesk Fusion 360, and OnShape) for shape design. Specifically, users need to select *Sensing surface reference*, *Camera reference*, *Light reference*, and *Support structure reference*. Users can also select multiple surfaces for

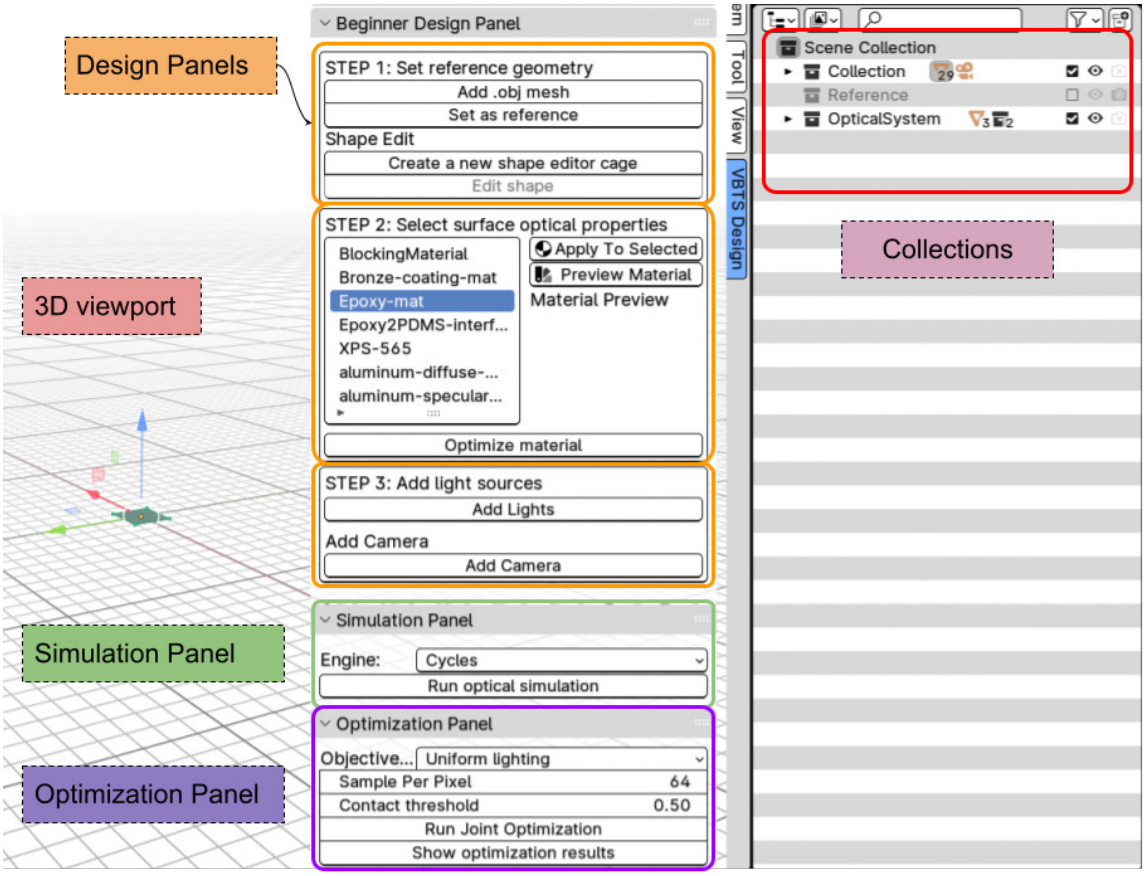



Figure 6.4: **OptiSense Studio Interface**: The interface of OptiSense Studio, which is built in Blender. The interface consists of a 3D viewport to visualize the model, various panels to perform parameterized digital design and select from component collections.

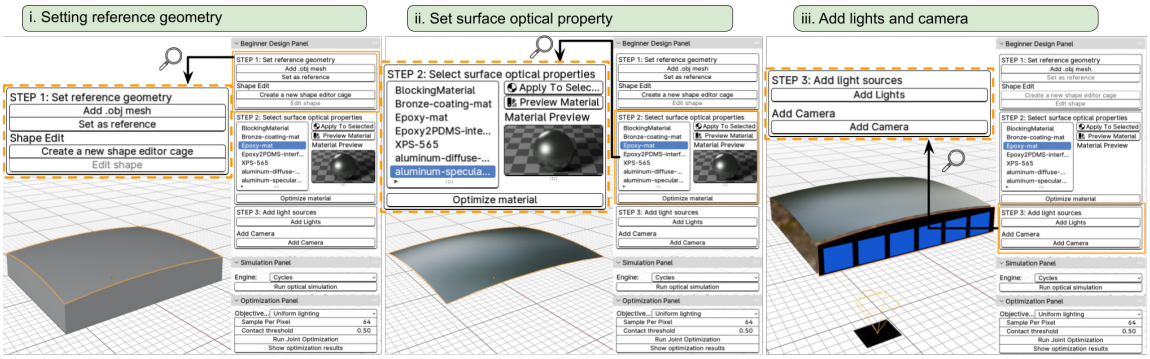



Figure 6.5: **Digital design guideline**: the three steps of the interactive design pipeline. i) Importing CAD shapes and setting them as reference geometries for optical elements; ii) Assigning material properties to the component from the component library or using user-defined materials; iii) Adding lights and the camera using reference geometries. The lights and the camera are chosen from the component library.

reference. Additionally, users can select *Indenter reference*, *Blocking reference*, *Interface reference*, and multiple light references as optical elements.

**Step 2**: Selecting optical properties for surfaces

After the addition of key surfaces, the user needs to assign optical properties to each surface. Based on the literature review of camera-based tactile sensors we provide a library of optical materials, which support refraction with rough interfaces, reflection with rough interfaces, and blocking of light paths. Specifically, we provide diffuse coating material [22], semi-specular coating material [77], PDMS refractive surface [22], and Epoxy rough refractive surface [107]. For a detailed overview refer to the Appendix B. All the materials were obtained by performing optical experiments in the lab and fitting analytical models available in the physics-based rendering and material modeling literature.

**Step 3**: Adding light sources and camera

Next the user chooses the light reference surface from *Reference* collection and selects the light type in the pop-up menu. Our library contains physically accurate light models based on the data available from the LED manufacturers. We provide point light sources with IES profiles sourced from manufacturers, spotlights with cut-off angles sourced from LED datasheets, and area light sources with dimensions sourced from LED datasheets. For a detailed overview refer to the Appendix B.

This allows accurate simulation and accurate physical light placement such that the sensor design preview is physically accurate.

For adding the camera, the process is similar to adding lights. Choose *Camera reference* surface from the *Reference* collection, click on **Add Camera** button, and select the desired camera. Our library provides commonly used Raspberry Pi cameras with field-of-view ranging from $60°$ to $160°$. For a detailed overview refer to the Appendix B.

### 6.3.3 Interactive optical simulation

The optical simulation techniques discussed in Section 3.3 generate physically correct tactile images. However, we found that in practice the rendering method could take $30\,\mathrm{min}$ to $1\,\mathrm{hour}$ for generating noise free images. For an interactive framework, this can seriously affect the designer's workflow due to time limits. Another key issue is the noise pattern produced by the rendering algorithm. The previously discussed rendering method uses MCMC techniques that produce correlated noise in the image. These noise artifacts prevent intermediate results from being useful for getting an initial idea of the sensing performance. Due to the above issues, we experimented with Stochastic Progressive Photon Mapping (SPPM) [32] and Image Denoising [1] for generating approximate tactile images in significantly reduced rendering time.

SPPM algorithm starts with a highly smoothed version of the image and progressively refines the image to produce noise-free and sharper features. This algorithm is a bidirectional technique, i.e., it generates path from both, light and camera. The bidirectional feature is specifically suited when the sensor contains point light sources. Moreover, SPPM is suitable for GPU acceleration [16, 33].

Image denoising is a technique of reconstructing images from their noisy versions using Neural Networks. The technique is studied in the context of the Path Tracing Appendix A rendering algorithm. The key idea is to learn a reconstruction kernel that recovers the original image from a noisy image using neighborhood noisy color values and extra features, such as normal and albedo. Though the technique was developed for path tracing which generates uncorrelated noise, it works well with the SPPM algorithm for our sensor setup.

Therefore, our interactive optical simulation pipeline uses SPPM with spp=64 to generate image with small amount of noise and uses OIDN [1] to generate final noise-free tactile image.

### 6.3.4 Design improvement: forward and inverse methods

Using the previous modules, the user has a digital design, simulation tool, and evaluation criteria. These components can be combined to perform forward and inverse design of the camera-based tactile sensors.

For the forward design process, the user can select any parameter, manually change that parameter in OptiSense Studio, and evaluate the design. We found that the light location setting is one parameter that can benefit from this approach. We show experiments in Section 6.4.3 to optimize light location using this approach.

In the inverse design process, the user can perform an automated search over the parameterized space, as discussed in Section 6.2. We show experiments to jointly optimize light type and sensing surface material using this approach for GelSight Mini, in the next section. We also show experiments to optimize the sensing surface material for a new sensor, GelBelt, in the GelSight family in the next section.

## 6.4 Experiments

In this section, we leverage our design framework to model and optimize four different types of GelSight sensors: commercial GelSight Mini [51], GelBelt with rolling capability, omnidirectional GelSight360 [94], and mirror-based GelSight Svelte [115]. These sensors have various simulation and design challenges. GelSight Mini and GelSight360 use *light piping* for uniform illumination on the sensing surface. GelSight360 and GelSight Svelte have curved sensing surfaces. In GelSight Svelte, the camera view is guided through multiple mirror surfaces to cover a curved finger-like sensing surface. We can simulate all sensors without any sensor-specific calibration. The key objectives of our experiments are that through our design framework users can easily edit previous sensor shapes (GelSight Mini), create new sensors (GelBelt), explore the design space of existing sensors (GelSight360), and optimize tactile perception by changing the shapes of optical components

automatically (GelSight Svelte).

### 6.4.1 Sim2Real comparisons

In this section, we compare our simulated tactile images with real tactile images for various sensors with different optics. Note that our simulation techniques are based on first principles and do not require any sensor-specific calibration data. Figure 6.7 shows the comparison for the GelSight Mini and GelBelt tactile sensors.

### 6.4.2 Case study I: Curved customizations of GelSight Mini

The GelSight Mini is one of the few commercially available tactile sensors and is adapted from the sensor design introduced in [51]. The sensor is shown in Figure 6.6Ai. It allows easy integration in robotic fingers that have a flat surface. However, for general-purpose robots, there might be a need to make the sensing surface non-planar. We show the design iteration of the nonplanar sensing surface of GelSight Mini in this section. First, we assemble the initial design using our design framework. Secondly, we modify the sensing surface shape using our cage-based representation. Thirdly, we iterate on the optical material properties of the sensing surface using our design. The simulation time to generate each GelSightMini tactile image takes 6 seconds on a M2 MacBook Air.

**Modeling GelSight Mini in OptiSense Studio**. We take the CAD provided by GelSight Inc. (original sensor vendor) and create an optical design as shown in Figure 6.6A(iv). Due to the availability of design optical components in our library, the optical design can be created in minutes, and a simulated tactile image can be generated. In Figure 6.6, we show the real and simulated sensor images after pressing the spherical indenter on the surface in parts (iii) and (v), respectively. The simulated image closely matches the real one. The simulation time for GelBelt is 8.9 seconds. In Table 6.1, we show the quantitative comparison between simulated tactile images and real-world prototype tactile images. Figure 6.7 compares the default version GelSight Mini with the simulated model.

**Editing sensing surface shape**. We modify the shape of the sensing surface by moving the control point of the cage-based representation, which is automatically generated. To obtain the cylindrical surface, we moved the cage control points in

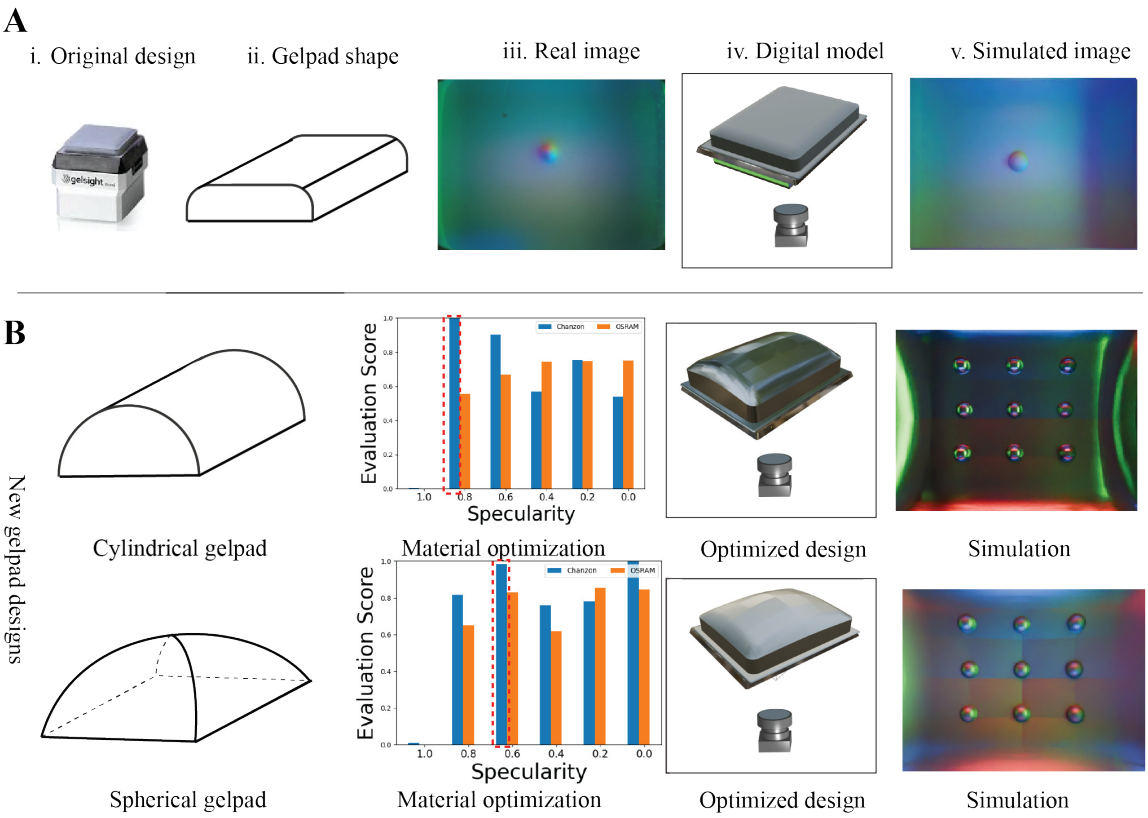



Figure 6.6: **Modeling and customization results of GelSight Mini**: A) shows the GelsightMini sensor (i) with default flat sensing surface (ii) and a real-world tactile image with a sphere indenter; (iv) shows the digital design and (v) shows the simulated image for this flat design. We created 2 curved variants—cylindrical and spherical—in B) by editing the initial sensing shape and show optimization results. For each new shape, we show the gelpad shape, coating material versus evaluation score plot for two light types, optimized digital design, and simulated tactile image with 9 spherical indenters.

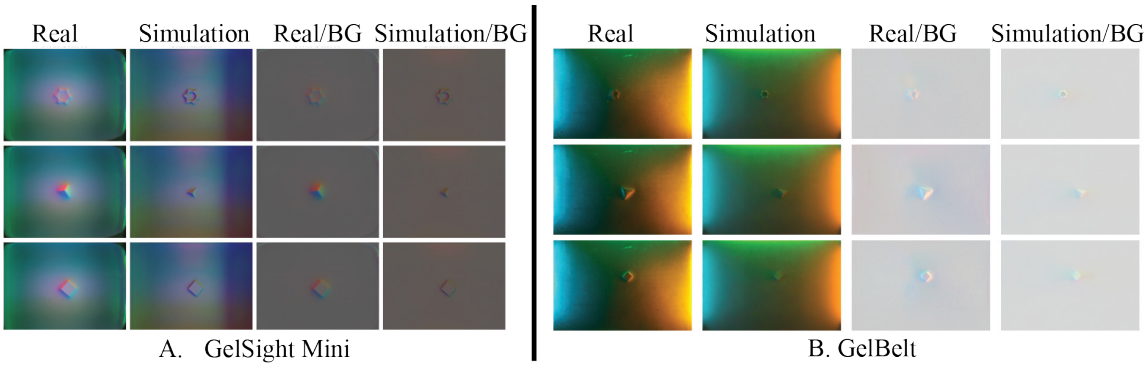



Figure 6.7: **Qualitative comparison between simulation and real-world tactile images**: We show image collected from the real-world prototype and simulation models for a set of objects for default version of GelSight Mini and manufactured GelBelt. Qualitatively, there are minor differences between the images for GelSight Mini because of manufacturing defects in the real sensor. Our data processing pipeline uses background-subtracted image (image/bg) for surface reconstruction. Therefore, we also compare image minus background image, image/bg, for both sensors. As can be seen in the last two columns, qualitatively the difference image is very similar for both sensors.

the middle row along the z-axis by $6\,\mathrm{mm}$ (this parameter was arbitrarily chosen to make the sensor cylindrical). To obtain the spherical surface, we moved the cage control center point along the z-axis by $9\,\mathrm{mm}$. (this parameter was arbitrarily chosen to make the sensor cylindrical). This shows the ability of our shape editing tool to create custom sensors with curved shapes for dexterous manipulation.

**Optimizing coating material**. After modifying the sensing surface shape, we try to obtain the best optical coating material for the sensing surface, while keeping the light locations fixed. We leverage the inverse design process to optimize the material. In both cases, we plot the normalized evaluation criteria for the best assessment.

For the cylindrical sensing surface, we find that the coating with specularity = 0.2 gives the best evaluation score, as shown in the first row of Figure 6.6 B.

For the spherical sensing surface, we find that the coating with specularity = 0.4 gives the best evaluation score, as shown in the second row of Figure 6.6 B.

In both cases, we identify the specific material for the curved sensing shapes and are able to obtain sensing performance similar to the flat sensing surface design,

Table 6.1: Comparison between simulated tactile images and the real ones on different metrics. The ↓ arrow shows that a lower value is desired and vice versa.

| GelSight Mini | | | |
|---|---|---|---|
| Indenter | SSIM ↑ | PSNR ↑ | MAE ↓ |
| square | 0.6602 | 21.1371 | 17.3402 |
| corner | 0.6574 | 20.9745 | 17.4338 |
| star | 0.6659 | 21.0729 | 17.4395 |
| **GelBelt** | | | |
| Indenter | SSIM ↑ | PSNR ↑ | MAE ↓ |
| square | 0.8883 | 19.1918 | 20.4762 |
| corner | 0.8928 | 19.4044 | 20.0913 |
| star | 0.8862 | 19.1698 | 20.4846 |

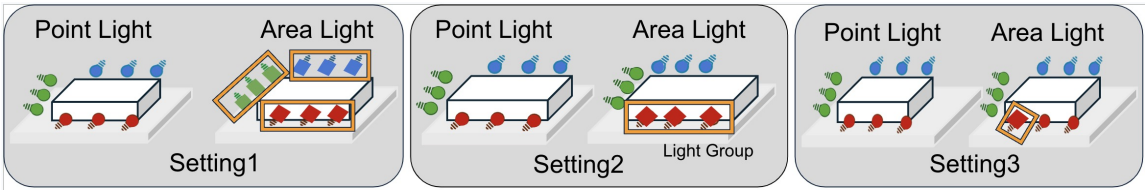



Figure 6.8: **Light Type Optimization Settings:** (A) shows *Setting 1*, in which we change the light type of all the lights simultaneously; (B) shows *Setting 2*, in which we change light type of all lights in a *Light Group*; (C) shows *Setting 3*, in which we change the light type of each light individually.

optimized by tactile sensing experts using manual iteration.

### Light improvement

For the optimization of light sources, we vary the light type. We consider 2 light types: *PointLight* and *AreaLight*. These light models are representative of real light sources present in our component library (Appendix B). Each light position and orientation are kept fixed. GelSight-like VBTS sensors have two or three light panels that contain lights of the same type. We call these similar lights collections a *light group*. For light improvement, we consider three different settings, which are described below.

**Setting 1.** In Figure 6.8A, we change the light type of all the lights in the

tactile sensor at the same time and evaluate the designs. The number of possible combinations will be equal to the number of light types available in the component library, which is two (*PointLight* and *AreaLight*) for now.

**Setting 2.** In Figure 6.8B, we change the light type of lights in a single light group independently and evaluate the designs. For example, in the GelSight Mini sensor, there are three light groups and two types of light models available. Therefore, the number of possible combinations is $2^3$, which is 8.

**Setting 3.** As shown in Figure 6.8C, we change the type of light individually and evaluate each design. For example, in the GelSight Mini sensor, there are two types of light models available and eleven lights in total. As a result, the number of settings is equal to $2^{11}$, which is 2048.

**Results.** After modifying the coating material, we choose the material settings to be diffuse. We obtain the best illumination design for the sensor while keeping the light locations and orientations fixed. We change the light type according to three settings introduced Section 6.4.2. Table 6.2 shows the results for *Setting 1* and *Setting 2*.

Table 6.2: Comparison between different light optimization designs on different metrics. In the table, "P" stands for *PointLight* and "A" stands for *AreaLight*.

| GelSightMini Cylindrical | | |
|---|---|---|
| Setting 1 | *RGB2Normal* | *NormDiff* |
| *AreaLight* | 0 | 0.6643 |
| *PointLight* | 0 | 0.6621 |
| Setting 2 | *RGB2Normal* | *NormDiff* |
| PPP | 1 | 0.6622 |
| PAA | 0.7678 | 1 |
| AAP | 0.2769 | 0.9639 |

### 6.4.3 Case study II: Rapid design of GelBelt

The roller version of GelSight was first introduced in [13] to allow rapid perception of large surfaces. The authors created a cylindrical casing and molded elastomer

over the casing. By repositioning the sensor by simply rolling, the sensor reduced the perception time while capturing consecutive images. We experiment with a new design that combines the benefit of rolling with the excellent tactile perception of a flat-sensing surface. The new design uses 2 rollers and a belt that rolls between them. The design concept is visualized in Figure 6.9Ai. The new design decouples the sensing surface design with the rolling phenomenon so that we can extend the sensing area without a direct impact on the wheel size.

We show the forward design and inverse process using our design tool in Figure 6.9. We consider variations in the choice of coating material.

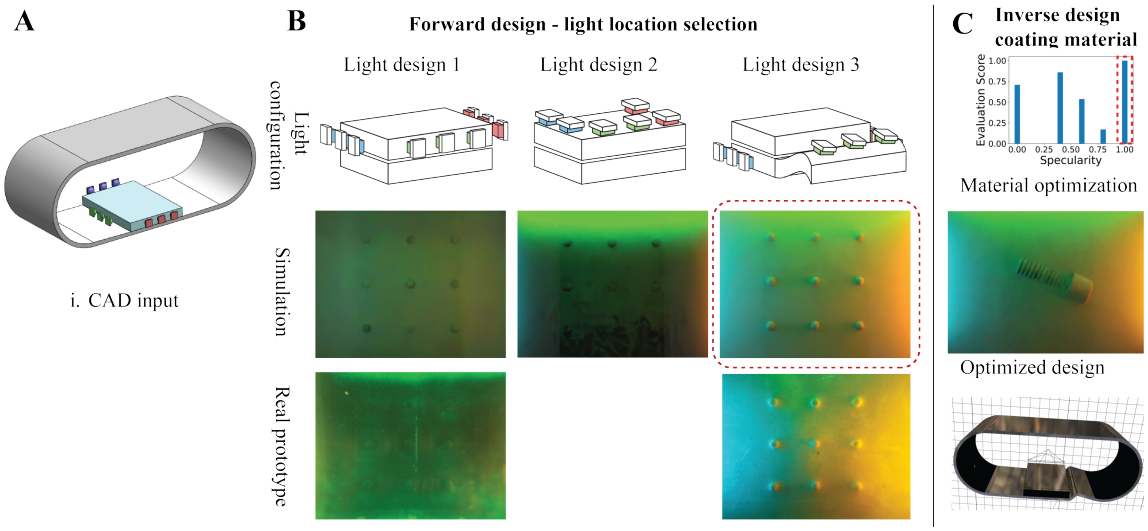



Figure 6.9: **Designing a new GelSight-like sensor, GelBelt**: We start with CAD design in (**A**) and create an optical design in OptiSense Studio. In (**B**), we perform forward design for the selection of light locations. A human manually places lights at three plausible locations and uses simulation-driven evaluation criteria to select the best light configuration. It is evident that the perception of spherical indenters improves significantly with this approach. We also compare the images generated using the physical prototypes of GelBelt with Light design 1 and Light design 3. The simulated and real tactile images are shown in the middle and bottom rows. We see a close match in the tactile images and that superiority of Light Design 3 in real and simulated images. In (**C**), after selecting the best light configuration, we optimize the coating material using the inverse design procedure.

**Forward design process with light type and light locations.**

Using the simulation toolbox, we investigate several configurations of the light

locations and LED type to get the best design for the sensor, as shown in Figure 6.9B. Initially, it is aimed to have the LEDs illuminating on the side of the acrylic to mimic the GelSight mini light configuration. However, both in simulation and experiments it is shown that light poorly reaches the sensing surface. This is because, unlike GelSight sensors, the silicone is not cured or attached on top of the acrylic letting a thin layer of air be trapped in between resulting in the total internal reflection of the light in acrylic. To tackle this problem, we reconfigure the illumination system and rerun the simulation. As shown in Figure 6.9B, lights are placed at different locations. The best configuration for the blue and red lights is to have them on the sides of the sensing surface but illuminating through silicone instead of acrylic. This configuration cannot be used for the green light which limits its location to be somewhere next to the acrylic. Several configurations are tested for the green light. It was observed that having the green light illuminated at an angle through the silicone showed acceptable results. Then, in an improved design, the silicone is bent over a small roller to better guide the light through the sensing surface. In this way, the green light can travel further on the sensing surface.

We manufactured the real-world prototypes of the Light design 1 and Light design 3 to compare the light design tactile images. Figure 6.9B bottom row shows the real prototype images for Light design 1 and Light design 3. We can clearly see that improvement in perception of spherical indenters using virtual forward design leads to direct improvements in real world prototypes. Therefore, our toolbox can enable illumination design completely virtually.

**Inverse design of sensing surface coating material**. After optimizing the light location, we try to obtain the best optical coating material for the sensing surface, while keeping light locations fixed. We leverage the inverse design process to optimize material. We plot the normalized evaluation criteria for the best assessment. The results are visualized in Figure 6.9C. We identify specularity=1 to be the best design using this process.

**Real-world prototype of optimal GelBelt sensor**.

To verify the results of the simulation, a prototype based on the optimal design is fabricated. Figure 6.10 compares the output images of the proposed real-world sensor with that of the simulation for the indentation of a screw, an electronic breadboard, and a gear rack. It is observed that the simulation output of the

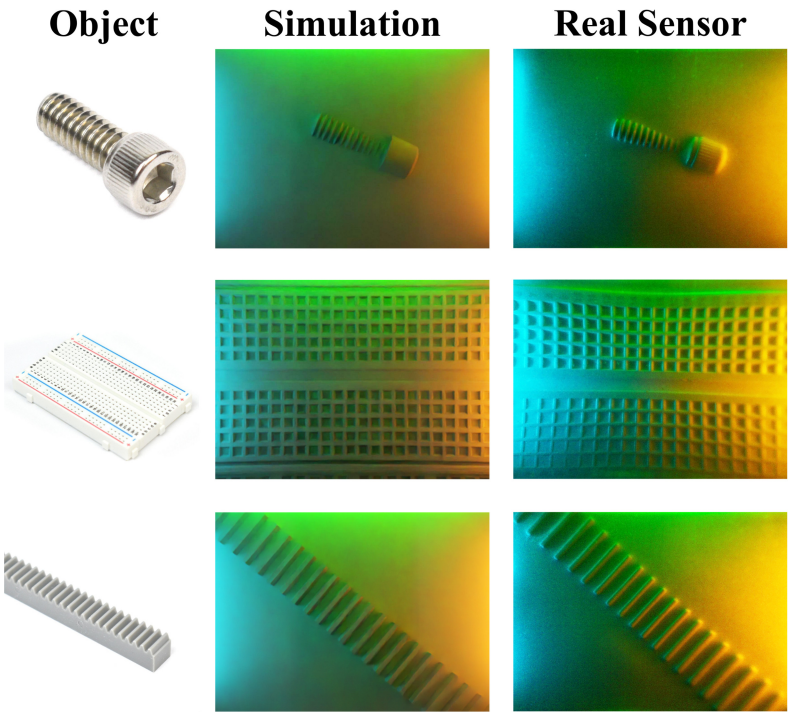



Figure 6.10: **GelBelt tactile image comparisons**: The simulated and real output of the optimized GelBelt sensor when contacting a screw, a breadboard, and a rack. It is observed that the real sensor performance highly matches the simulation and well shows the object geometries.

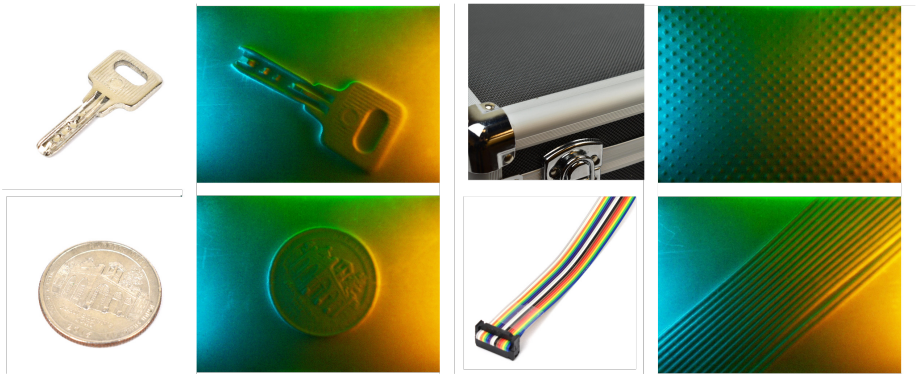


Figure 6.11: **GelBelt sensor tactile images indented with daily-use objects**: It is observed that the optimized roller sensor can sense fine details of the objects and surfaces.

approximate geometry highly matches the images of the real sensor in all cases. This similarity highly supports the validity of our design framework. That being said, researchers and designers can benefit from using OpticSense Studio and its modules to predict the outcomes and optimize their design before fabricating the real-world sensor. The performance of the Gelbelt sensor while indented by various small parts is shown in Figure 6.11. In Table 6.1, we show the quantitative comparison between simulated tactile images and real-world prototype tactile images. Similarly to previous results, the GelBelt sensor can sense fine texture on the surface in a relatively extended area.

### 6.4.4 Case study III: Optical component shape variation and light variation for GelSight360

In this section, we consider the GelSight360 sensor, which was introduced in [94]. The authors used *light-piping* and embedded lights to create a VBTS tactile sensor that could provide sensing in the forward direction without any occlusion. In this sensor, illumination design requires figuring the light color setting and surface shape of multiple optical components for best perception. This makes the design problem particularly challenging. We first discuss light color variation and then discuss optical component shape variation using our objective functions.

**Light type variation.** In this experiment, we consider various light configuration with *RGB2Norm* and *NormDiff* objective function. The vertical lights in the sensor are divided into 8 light groups as shown in Figure 6.12B. Table 6.3 shows the objective function values. According to the objective functions, *GBBRRGGB* has the best mapping between RGB to normal with the highest robustness to camera noise. Note that performing the light variation in the real-world requires designing and manufacturing new LED boards and embedding them in the resin shell. The manufacturing time for this experiment could be prohibitively and expensive for designers.

**Optical component shape variation.** We keep the sensing surface the same as the original design and change other optical components to show the effect. We consider two experiment settings: (a) *Setting 1*- only change the innermost resin surface; (b) *Setting 2* - change the resin surface, interface surface between resin and

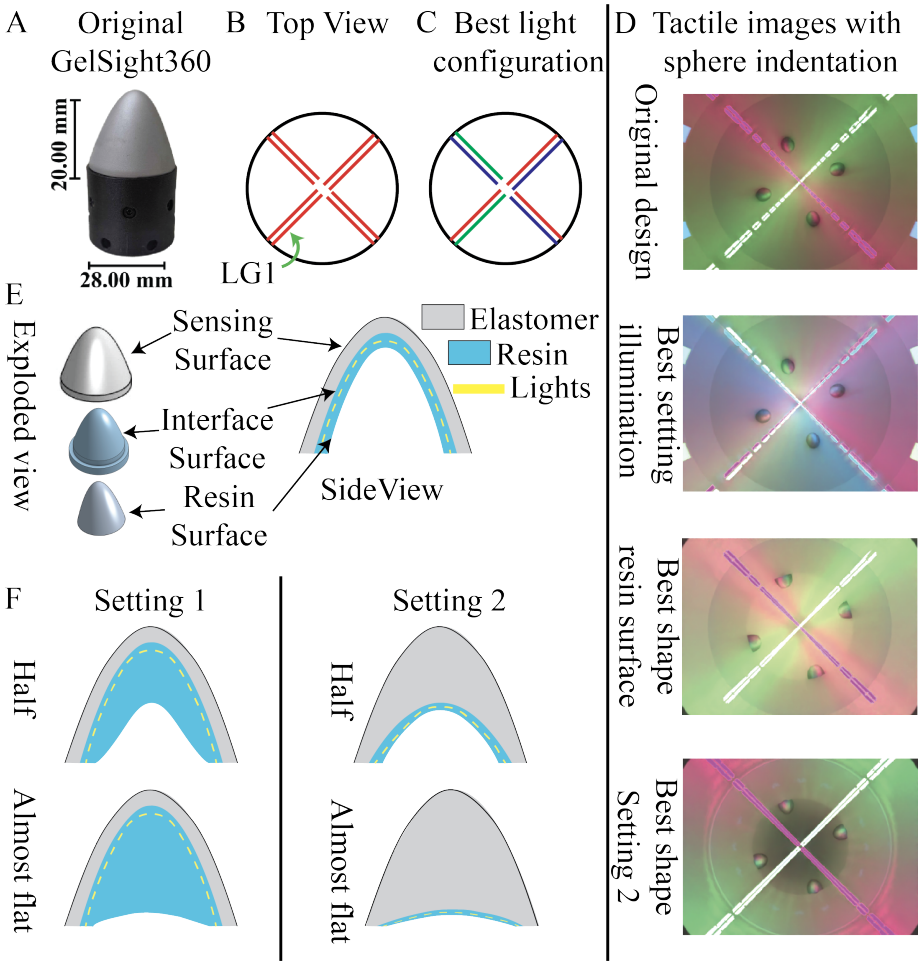



Figure 6.12: **GelSight360 shape and light variation description**: (**A**) Original GelSight360 sensor as introduce in [94]. (**B**) Top view is shown with first Light Group (LG) shown as LG 1. We number the LG anti-clockwise starting from LG 1. (**C**) shows the best light configuration in the combinations considered in our experiments. (**D**) shows the rendered tactile images for original sensor, best illumination setting, best resin shape design and best shape design in Setting 2 considered in our experiments. (**E**) shows the exploded view with labels and the corresponding side view with key components marked. (**F**) shows the variations in Setting 1 and Setting 2 considered in our experiments, namely, *Half* and *Almost flat*.

Table 6.3: **GelSight360 light type variation**: "R", "G" and "B" stands for red, green, and blue light color respectively. This table notes the score of the sensor designs with different illumination setting. Higher scores are better.

| Light configuration | *NormDiff* | *RGB2Norm* |
|---|---|---|
| GRRGGRRG (original) | 0.3138 | 0 |
| GBBRRGGB | 0 | 0.0597 |
| GRRBBGGR | 0.5047 | 0.0228 |
| RGBRGBRG | 0.2664 | 0.0927 |
| **GBRBRGBR** | 0.8976 | 0.5205 |

Table 6.4: **GelSight360 shape variation**: This table notes the objective functions scores for sensor designs with shape variations according to *Setting 1* (only resin surface change) and *Setting 2* (resin, interface and vertical light shape change). We observe that *Almost flat* setting performs best across different shapes choices. Higher scores are better.

| Setting 1 | *NormDiff* | *RGB2Norm* |
|---|---|---|
| Original | 0.3138 | 0.0 |
| Half | 0.8649 | 0.5257 |
| **Almost flat** | 1 | 0.7922 |
| Setting 2 | *NormDiff* | *RGB2Norm* |
| Original | 0.3138 | 0.0 |
| Half | 0.5211 | 0.6494 |
| **Almost flat** | 0.9687 | 1 |

elastomer, and light stripes embedded in the resin. We consider three settings in each case: original shape, half shape, and making the biggest surface almost flat, as shown in Figure 6.12F. Table 6.4 shows the objective function values. We notice that *Almost flat* shape variation performs best in both settings with Setting 2 being the optimal among all the designs considered for this sensor in our experiments. As can be seen in Figure 6.12D bottom-most tactile image, perception of sphere indenters has improved as compared to initial design.

### 6.4.5 Case study IV: Sensor shape optimization for GelSight Svelte

In this section, we consider the GelSight Svelte sensor, which was introduced in [115]. The authors used multiple mirrors to route the camera view to the full human-finger-shaped sensing surface. This allows sensing along the entire finger instead of just the tip. The optics of the sensor was selected using 2D raytracing simulations. However, the authors were unable to simulate the tactile images prior to manufacturing. During our investigation, we noticed the "smearing" issue when indenters are pressed on the sensing surface, as shown in Figure 6.13. The amount of distortion depends on the indenting location on the sensing surface. This effect is caused by the larger back mirror design. The original design only considered sensing surface coverage in the 2D side view.

We consider the design of the back mirror surface to alleviate this issue and improve sensing performance. We use *As-orthographic-as-possible* (AOAP) objective function in this experiment. To show a proof of concept, we first consider a simplified sensing surface and focus on improving perception at the center of the sensing surface. The key optical surfaces are shown in Figure 6.2A top. We consider cage-based parameterization of the larger mirror surface, M1. This reduces the search space by orders of magnitude from $22\,680$ to $81$. We initialize the cage on the original mirror shape. We choose optimization parameters, $\mathcal{C}_{\text{min}}$ such that M1 is flat and $\mathcal{C}_{\text{max}}$ such that M1 has the largest curvature possible without intersecting with the sensing surface. We use CMA-ES [34] for optimizing the shape parameters.

The optimization curve is shown in Figure 6.14A. The AOAP score for the initial and optimized design is $0.236$ and $0.635$. As can be seen from the rendered tactile images in Figure 6.14B, the "smearing" effect or distortion is almost gone in the optimized design. We created real-world sensor prototypes to validate our simulation experiments. In Figure 6.14C, we show our sensor prototypes of initial and optimized design. The corresponding tactile images and zoomed-in view clearly shows that sim2real works well for GelSight Svelte sensors. Thus, our shape optimization pipeline could be used to obtain the best optical component shapes to reduce optical distortion and improved shape perception. Note that this approach can be applied to any optical surface design.

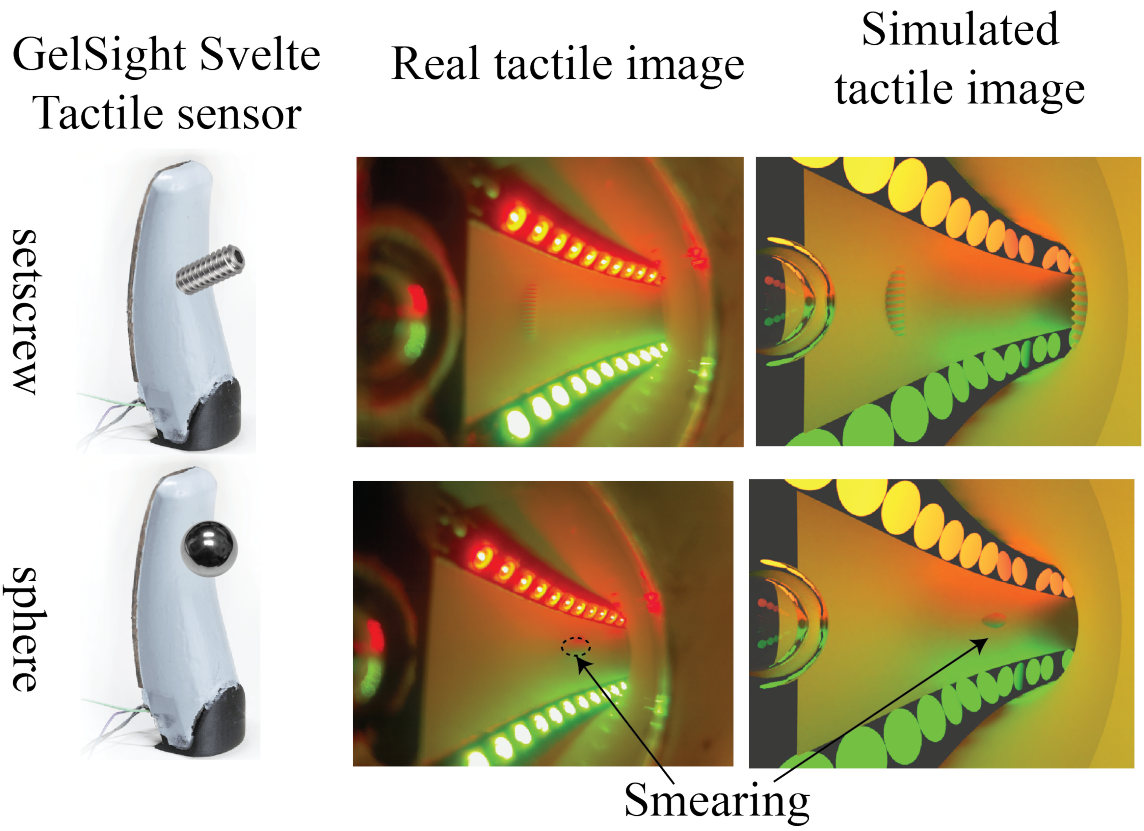



Figure 6.13: **GelSight Svelte issues**: We compare the simulated image against the real-world prototype tactile images. The simulated images are a close match to the real images. The top and bottom row shows images with setscrew and ideal sphere indenters at different sensing surface locations. As can be seen from the bottom row, the distortion depends on the indentation location. In the bottom row, the ideal sphere is "smeared" or distorted substantially, and is hard to perceive physical shape properties like sphere radius.

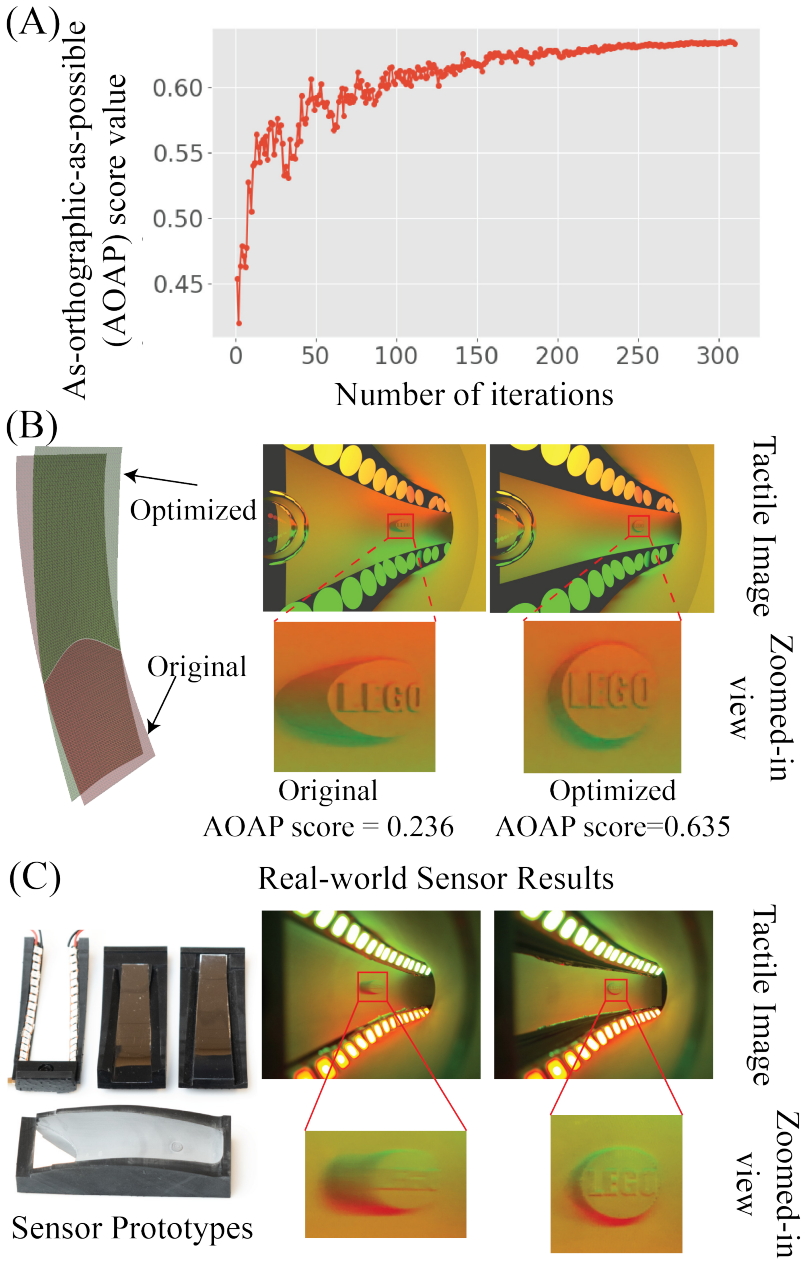



Figure 6.14: **GelSight Svelte shape optimization results**: We show the shape optimization results for the GelSight Svelte sensor. (**A**) shows the AOAP function score during the CMAES optimization procedure. (**B**) shows the initial and optimized larger mirror surface mesh in red and green respectively. We also simulated tactile images for the two designs. The optimized design has significantly reduced distortion as compared to the initial design. (**C**) We manufactured sensor prototypes to compare the improvement in real world for initial and optimized design. The left visual shows our prototype. We show the tactile images from the real world prototypes and their zoomed-in view. The real and simulated images of the initial design both show significant distortion of the lego block. The issue is resolved completely in real and simulated images for optimized design.

## 6.5 Discussion

We achieve four key objectives through the case studies. First, we show the ease of use of OptiSense Studio by creating a digital twin of GelSight Mini, GelSight360, and GelSight Svelte within minutes. Secondly, we are able to obtain parameterized designs and show that our parameterization enables users to easily explore the design space of various GelSight-like tactile sensors across various design axes: illumination, coating material, and geometric shape of optical components. Thirdly, we are able to perform design optimization for all the components of a sensor for a range of GelSight-like sensors with complicated optics. Lastly, we are able to convert a concept design of a new sensor (GelBelt), for a new application (large area sensing), into a valid sensor design and perform different forward and inverse design optimizations to obtain the best sensor design.

# Chapter 7

# Conclusions and Future Research

In this thesis, we proposed the first objective-driven design tool for GelSight-like tactile sensors. In doing so, we created an optical tactile simulator and a library of calibrated sensor components. We also defined new efficient objective-driven computational methods for automatic parameter selection of the complex optical system. We proposed a modular design interface for interactive modeling and design space exploration, which is based on interactive rendering algorithms. Finally, we show the utility of our framework for the design optimization of a range of vision-based tactile sensors for robotic manipulation and tactile perception. We manufactured the novel designs and investigated the performance boost between virtual and real designs.

Although this thesis takes significant steps towards automatic GelSight-like sensor design, there is a gap in enabling sensor design by novice users with no knowledge of sensing. To design a complete GelSight-like tactile sensor, a user needs a sequence of actions. Therefore, we need a framework that takes the operations introduced in this thesis and generates a valid sequence of actions to compose a full sensor. An approach in robotic structure design introduced in [114] might be a good starting point towards that goal.

In the robotics community, the current approach is to first design robotics structures such as robot hands and then add sensing to those structures. If the user wants to incorporate VBTS into those structures, it requires a complete redesign of the corresponding structure. Our framework does not offer any guidance for this

workflow.

## 7.1 Future Directions

The work described in this thesis demonstrates how to rapidly prototype VBTS and optimize their optical parameters, which can then be manufactured directly in the real world. However, our results thus far are only the first steps toward applying these concepts to computational design for sensing. Our work is closely related to soft robotics design, as tactile sensors need to be embedded in the robotic structure or co-designed for optimal performance. Since the advances in the co-design of VBTS and robotic structures are relatively recent, the field of developing the necessary design tools is still relatively new, with many interesting challenges and open research problems. We outline some of the most important open challenges that we think are crucial to pushing the field forward.

**Combining marker simulation and physical simulation approximation with optical simulation for design** The tactile sensors have a soft surface as a contact layer with the environment. It is useful to measure the deformation of the soft elastomer during interaction. These deformations are helpful to measure the tangential force, shear and torque signals. We presented a technique to simulate the sorting of edges caused by the static indentation on the elastomer in Section 3.2.2. Subsequently, various researchers [29, 86] have proposed better approximate models for simulation of the soft elastomer. These models propose a model of local movement of gelpad nodes. [116] use similar idea to propose marker motion generation on the gelpad surface. In our experiments, we found that these models fail to generalize for the range of sensors with complex optics considered in this thesis. However, we believe that incorporating these approximate models into our simulation framework could enable sensor design for various other GelSight-like sensors with some extensions.

**Combining physical simulation with optical simulation.** Recently, researchers have incorporated GelSight-like sensors in compliant structures [56, 57]. These sensor structures are meant to be deformed substantially when they are interacting with objects. These deformations can not be modeled with rigid body simulators or by interpenetrating objects with sensing surfaces. Therefore, to generate valid tactile images, we need to incorporate soft body deformations. We used Finite

Element Methods (FEM) in [66] to simulate GelSight FinRay sensor deformations before passing obtained shapes to our design pipeline. However, using interactive and efficient soft body simulators is needed for design problems. We believe that this could enable the co-design of soft robotics structures with high-resolution tactile sensing for robot proprioception and contact sensing.

**Expanding the design framework for other VBTS sensors.** Although the pipeline performed well for the design and simulation of the GelSight family of sensors, we expect that our approach has the potential to incorporate optical components required for other types of camera-based sensors. We hope to incorporate an approximate model of fluorescent lighting as introduced in [55] to allow the design of the FinRay GelSight family. Also, 9DTact [52] introduced absorption layers which can be added as another optical material in our component library. Other camera-based tactile sensors record a single color intensity [52] or a change in intensity [101] to recover tactile signals (surface depth or force). For example, in the 9DTact sensor family, the single-color intensity image is used to recover surface normals. To extend our design toolbox for this sensor, users can add new evaluation criteria. Users can use the general idea—to create a mapping between the measured image signal and tactile signal—of the evaluation criteria proposed in this work.

**Robotics-focused objective functions.** In this thesis, we propose various objective functions to improve the design. Those objective functions are focused on improving the tactile signal fidelity, especially color image and sensor coverage. This works well if the application area for tactile sensing is in perception, for example, aircraft inspection. However, for robotic applications, it is unclear what is the most important tactile signal and what is the resolution required. In the literature, various researchers have co-designed robots and their control strategies by using machine learning. This enables direct optimization of the robot structure and control for the robot application. Creating a pipeline to score VBTS designs directly for the task (dexterous manipulation) by using machine learning might be an interesting future direction.

**Efficient optical simulation.** Since optical simulation is at the heart of our framework, it is critical to have an efficient and general optical tactile simulator. Although we showed that it is possible to simulate complex optical illumination, such as *light piping* and generate tactile images that interact with a range of sensors. We found

that sometimes the optical framework fails in unintuitive ways. Figure 7.1 shows the image of GelSight Mini with various settings: diffuse BSDF and specular BSDF with very high roughness 0.99 in the top and bottom rows. The rendering algorithm used for all the images is PSSMLT [46] implemented in Mitsuba1 [40]. With these settings, all the images in Figure 7.1 should be similar. However, we notice that the image with PointLight contains mainly noise for both the BSDF setting. The issue disappears if we use the SPPM [32] rendering technique and diffuse BSDF with point light, as shown in Figure 7.2. However, SPPM is not able to generate the correct image for PointLight and specular BSDF.

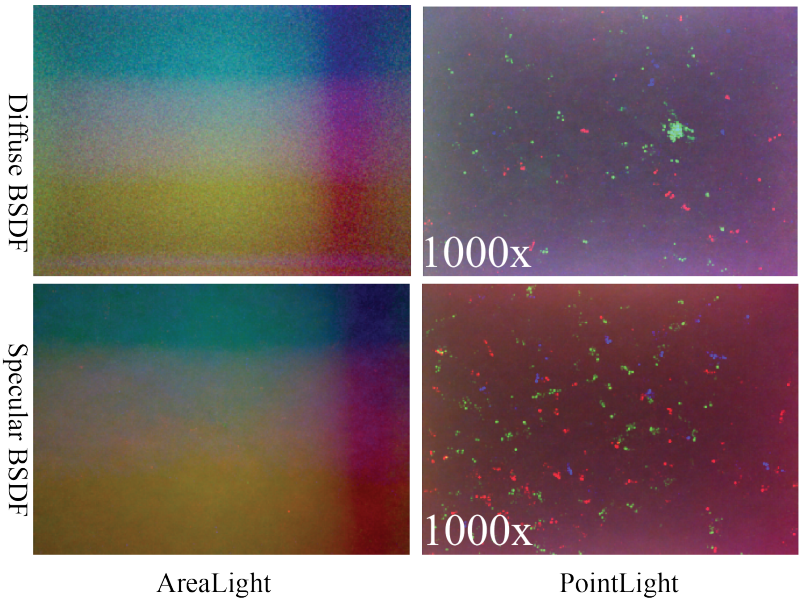



Figure 7.1: **Rendering failures with PointLight**: The figure shows tactile images for the GelSight Mini sensor generated using PSSMLT rendering technique. The rows contain tactile images with different coating materials: Diffuse and Specular with very high roughness 0.99. The columns contain tactile images with different light types: AreaLight and PointLight. As can be seen in the right column, PSSMLT produces noise with PointLight for both BSDF cases. This result is an unexpected failure case.

This is because of two reasons: mismatch between optical properties of real and virtual components; and high computational cost of finding valid light paths from point light sources and a high number of refractive elements with delta BSDFs in VBTS sensors. There has been recent progress [25, 30, 61] in finding paths through

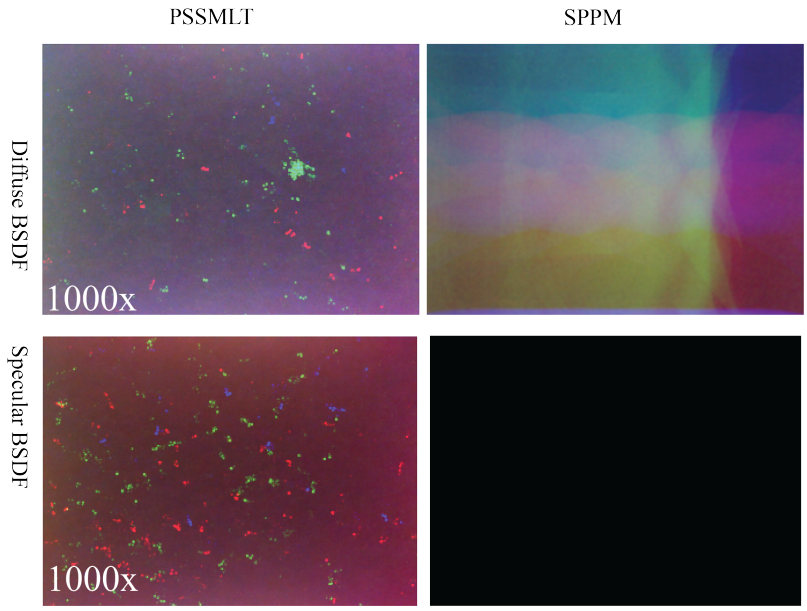



Figure 7.2: **Rendering failures between algorithms**: The figure shows tactile images for GelSight Mini sensor generated with PointLight and two different BSDF settings: Diffuse and Specular with high roughness $0.99$ The top row shows that SPPM algorithm is able to generate reasonable tactile image with PointLight and Diffuse BSDF as compared to PSSMLT, which produces noise. However, SPPM fails catastrophically with PointLight and specular BSDF with high roughness, while PSSMLT generates almost noise image.

refractive surfaces efficiently. However, those techniques fail if the light is a delta light source.

**Fabrication errors and constraints** The fabrication process of these sensor requires multiple steps like 3D printing, molding, spray-coating and assembly. Each of these steps can impose various constraints on the sensor design. Moreover, each of these manufacturing steps can introduce a margin of error, for example, the 3 printed surface shape may be within $10\,\%$ of the optimized surface shape. Our design framework does not incorporate these errors into the objective functions. We tried to tackle this issue by performing a sensitivity analysis on various design parameters in Section 4.4. However, incorporating these errors more explicitly into the design process can enable better designs. One way to formally consider sensor performance as a stochastic function is to use Bayesian optimization [93].

## 7.2 Lessons learned

Firstly, we have learned that there is a need to balance automation and user input. In addition to the well-known conflict between ease of use and design freedom, such balance is important in the development of efficient design tools. User preference can provide us with good design initialization to prune the search space of the optimization algorithm. This is specifically true for the shape design of the free-forming optical components in the sensors.

Secondly, approximate and quick simulation is better than very accurate and slow simulation for tactile sensor design. Since the design space of the VBTS tactile sensor is relatively large. It is essential to allow users to decide which design spaces are most important to optimize. We found that in most of the VBTS sensors we do not have reflective sharp caustics. Therefore, techniques like path-guiding or photon mapping can generate approximate tactile images. Moreover, it might be interesting to incorporate new denoising methods [14, 110] to further improve the efficiency of these techniques.

# Appendix A

# Path tracing

Consider the scene shown in Figure 3.16. The light radiance received at point $p_0$ is

$$L(p, \omega_0) = L_e(p, \omega_0) + \int_{S^2} f(p, \omega_0, \omega_i) \\ L(t(p, \omega_i), -\omega_i)|cos\theta_i| d\omega_i$$

where $L_e(p, \omega_0)$ is light emitted by point $p$ towards direction $\omega_0$; $f(p, \omega_0, \omega_i)$ is the material model, which gives the fraction of light emitted in direction $\omega_0$, when receiving light from direction $\omega_i$; $t(p, \omega_i)$ is the point in the light path visited prior to hitting point $p$ at an angle $\omega_i$; $|cos\theta_i|$ is the Jacobian of the solid angle $\Omega$ w.r.t. polar coordinates. On a high level, the first term represents light emitted at $p_0$ and the second term represents light emitted by all the points in the scene towards $p_0$ sampled according to some probability distribution. The rendering equation can not be solved analytically, as it is a recursive equation (as term $L(p, \omega)$ appears on both side of the equation) in high dimension for a generic scene. To solve it, path

integral formulation of the above equation is considered, which is as following:

$$L(p_1 \to p_0) = L_e(p_1 \to p_0) \tag{A.1}$$

$$+ \int_A L(p_2 \to p_1) h(p_2 \to p_1 \to p_0) dA(p_2) \tag{A.2}$$

$$+ \int_A \int_A L(p_3 \to p_2) h(p_3 \to p_2 \to p_1) dA(p_3) dA(p_2) \tag{A.3}$$

$$+ \dots \tag{A.4}$$

According to the above equation, we need to generate all the paths starting from light sources, hitting different objects in the scene and reaching the camera. In practice, we just need some paths which carry most of the power from light sources to the points on the camera film and probabilistically terminate the computation. Each integral in the above equation in itself is solved through Monte Carlo integration with sampling probabilities biased towards points $p_i$, which will have more light contribution.

On a high level, the rendering involves following steps:

- Sample point on the camera film based on the camera model
- Sample points on the objects in the scene using some probability distribution
- Try to connect the object point to the light source
- Collect the light contribution of that path multiplied by the probability of that path
- Probabilistically terminate paths after certain max length based on some criteria

# Appendix B

# Component library

We provide 7 optical materials, obtained 6 light models and 6 camera types as shown in Figure B.1.

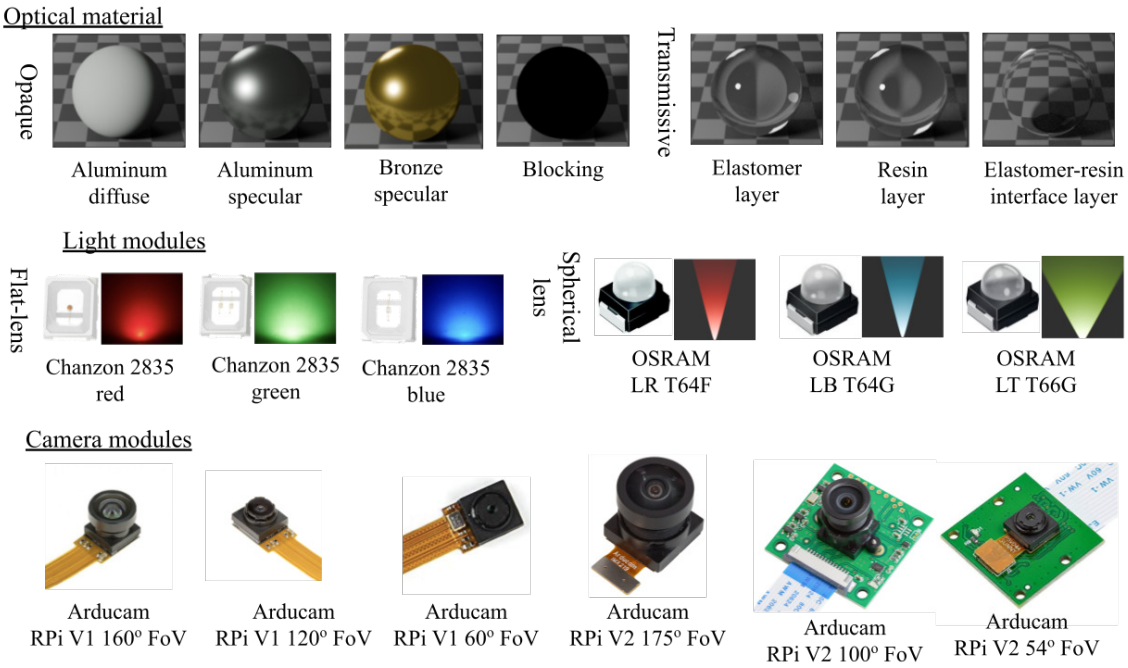



Figure B.1: **Component library in OptiSense Studio**: This figure shows the various components present in the library provided with our design interface. These components cover the design space of the GelSight sensor family and provide relevant design spaces to develop new sensors.

# Appendix C

# Design tutorial with our design interface

In this section, we give a short tutorial to setup a camera-based sensor design, using GelSight Mini as an example.

**Step 1: Adding shapes**

To add shapes click on the **Add .obj mesh** button and assign them as reference geometry by clicking on the **Set as reference** button.

Users can edit the surface shape using cage-based representation [102]. To create a cage around the shape, select the shape and click the **Create a new cage** button. To edit the shape, click the **Edit shape** button, select the cage vertices to move, and then move the vertices to change the shape of the surface. As noted in [102], the cage-based representation is differentiable and in the future is amenable to differentiable sensor shape design.

**Step 2: Assign optimal material property**

To assign the optical material, the user selects the desired optical surface from the *OpticalSystem* collection, selects the desired material from the library and then clicks on **Apply To Selected** button. The user can also preview the material before assigning it by selecting the material and clicking the **Preview Material** button.

**Step 3: Add light and camera**

For adding the lights, choose *Light reference* surface from the *Reference* collection, click on **Add Light** button, and select the desired light. Our library provides commonly

used LEDs with flat and spherical lenses, commonly used in camera-based sensors. For a detailed overview refer to the Appendix B.

To add the camera, the process is similar to adding lights. Choose *Camera reference* surface from the *Reference* collection, click on **Add Camera** button, and select the desired camera. Our library provides commonly used Raspberry Pi cameras with field of view ranging from $60°$ to $160°$. For a detailed overview, refer to Appendix B.

# Appendix D

# Fabrication of the optimized curved VBTS sensor

After obtaining the best design for GelBelt using our digital design framework, we manufactured a real-world prototype and tested its feasibility. PLA was used to 3D print the frame of the sensor and the handles. Wheels were 3D printed by Form 3+ printer in Black Resin Material to have a smoother print surface. The Acrylic part of the sensor was laser-cut to the shape and fixed in its housing using thin double-sided tape. The belt is made of Silicone XP565 (Silicone Inc.) while coated with Aluminum powder on the contact surface. Silicone itself cannot slide on the acrylic due to the high frictional force between two surfaces, therefore, an intermediate layer is required to complete the task of sliding. For this purpose, wide clear tape is attached to the inner side of the belt as it showed acceptable stickiness to silicone on the glue side while having a small friction with acrylic on the other side. To make the prototype of the sensor, The belt was fabricated by having a flat mold. It should be mentioned that the belt could be fabricated using a circular mold to have continuous rolling over the surface which will be considered in the future. After Silicone was cured and coated with aluminum powder, the belt was removed from the mold and wide tape was attached to the uncoated side of it. To have the complete belt, the belt was bent all over the rollers and then attached together using the wide tape. Regarding the lights, several SMD 3528 LEDs were linearly arranged for each of the red, green, and blue lights. The lights were soldered on a PCB and

then fixed on the frame of the sensor using screws.

# Bibliography


[1] Attila T. Áfra. Intel® Open Image Denoise, 2024. https://www.openimagedenoise.org. 6.3.3

[2] Arpit Agarwal, Mohammad Amin Mirzaee, Xiping Sun, and Wenzhen Yuan. A modularized design approach for vision-based tactile sensors based on real2sim. . 1.2

[3] Arpit Agarwal, Achu Wilson, Timothy Man, Adelson Edelson, Ioannis Gkioulekas, and Wenzhen Yuan. Vision-based tactile sensor design using physically based rendering. . 1.2

[4] Arpit Agarwal, Timothy Man, and Wenzhen Yuan. Simulation of vision-based tactile sensors using physics based rendering. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1–7. IEEE, 2021. 1.2

[5] Arpit Agarwal, Abhiroop Ajith, Chengtao Wen, Veniamin Stryzheus, Brian Miller, Matthew Chen, Micah K. Johnson, Jose Luis Susa Rincon, Justinian Rosca, and Wenzhen Yuan. Robotic defect inspection with visual and tactile perception for large-scale components. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 10110–10116, 2023. doi: 10.1109/IROS55552.2023.10341590. 1.1

[6] Michael Ahn, Henry Zhu, Kristian Hartikainen, Hugo Ponte, Abhishek Gupta, Sergey Levine, and Vikash Kumar. Robel: Robotics benchmarks for learning with low-cost robots. In *Conference on robot learning*, pages 1300–1313. PMLR, 2020. 3.3

[7] Jérémie Allard, Stéphane Cotin, François Faure, Pierre-Jean Bensoussan, François Poyer, Christian Duriez, Hervé Delingette, and Laurent Grisoni. Sofa-an open source framework for medical simulation. 2007. 3

[8] Iris Andrussow, Huanbo Sun, Katherine J Kuchenbecker, and Georg Martius. Minsight: A fingertip-sized vision-based tactile sensor for robotic manipulation. *Advanced Intelligent Systems*, 5(8):2300042, 2023. 2.1

[9] Ronen Basri, David Jacobs, and Ira Kemelmacher. Photometric stereo with general, unknown lighting. *International Journal of computer vision*, 72:239–257,

2007. 3.2.1, 4.2

[10] Maria Bauza, Oleguer Canal, and Alberto Rodriguez. Tactile mapping and localization from high-resolution tactile imprints. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 3811–3817. IEEE, 2019. 4.3.2

[11] Roberto Calandra, Andrew Owens, Dinesh Jayaraman, Justin Lin, Wenzhen Yuan, Jitendra Malik, Edward H Adelson, and Sergey Levine. More than a feeling: Learning to grasp and regrasp using vision and touch. *IEEE Robotics and Automation Letters*, 3(4):3300–3307, 2018. 4.3.2

[12] Berk Calli, Arjun Singh, James Bruce, Aaron Walsman, Kurt Konolige, Siddhartha Srinivasa, Pieter Abbeel, and Aaron M Dollar. Yale-cmu-berkeley dataset for robotic manipulation research. *The International Journal of Robotics Research*, 36(3):261–268, 2017. 4.5.1

[13] Guanqun Cao, Jiaqi Jiang, Chen Lu, Daniel Fernandes Gomes, and Shan Luo. Touchroller: A rolling optical tactile sensor for rapid assessment of textures for large surface areas. *Sensors*, 23(5):2661, 2023. 6.4.3

[14] Chuhao Chen, Yuze He, and Tzu-Mao Li. Temporally stable metropolis light transport denoising using recurrent transformer blocks. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, (4), 2024. 7.2

[15] Blender Online Community. *Blender - a 3D modelling and rendering package*. Blender Foundation, Stichting Blender Foundation, Amsterdam, 2018. URL `http://www.blender.org`. 6.3.1

[16] Tomáš Davidovič, Jaroslav Křivánek, Miloš Hašan, and Philipp Slusallek. Progressive light transport simulation on the gpu: Survey and improvements. *ACM Trans. Graph.*, 33(3), jun 2014. ISSN 0730-0301. doi: 10.1145/2602144. URL `https://doi.org/10.1145/2602144`. 6.3.3

[17] Paul Debevec, Tim Hawkins, Chris Tchou, Haarm-Pieter Duiker, Westley Sarokin, and Mark Sagar. Acquiring the reflectance field of a human face. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 145–156, 2000. (document), 3.2.3, 3.9

[18] Zihan Ding, Nathan F Lepora, and Edward Johns. Sim-to-real transfer for optical tactile sensing. *arXiv preprint arXiv:2004.00136*, 2020. 2.2

[19] Won Kyung Do, Bianca Jurewicz, and Monroe Kennedy III. Densetact 2.0: Optical tactile sensor for shape and force reconstruction. *arXiv preprint arXiv:2209.10122*, 2022. (document), 1.1, 2.1, 3.3

[20] Won Kyung Do, Bianca Jurewicz, and Monroe Kennedy. Densetact 2.0: Optical tactile sensor for shape and force reconstruction. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 12549–12555. IEEE, 2023.

2.1

[21] Siyuan Dong, Wenzhen Yuan, and Edward H Adelson. Improved gelsight tactile sensor for measuring geometry and slip. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 137–144. IEEE, 2017. 2.1

[22] Siyuan Dong, Wenzhen Yuan, and Edward H. Adelson. Improved gelsight tactile sensor for measuring geometry and slip. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 137–144, 2017. doi: 10.1109/IROS.2017.8202149. (document), 2.1, 6.3.2

[23] Elliott Donlon, Siyuan Dong, Melody Liu, Jianhua Li, Edward Adelson, and Alberto Rodriguez. Gelslim: A high-resolution, compact, robust, and calibrated tactile-sensing finger. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1927–1934, 2018. doi: 10.1109/IROS.2018.8593661. 1.1, 2.1, 2.1, 2.3

[24] Jonathan Dupuy and Wenzel Jakob. An adaptive parameterization for efficient material acquisition and rendering. *ACM Trans. Graph.*, 37(6), dec 2018. ISSN 0730-0301. doi: 10.1145/3272127.3275059. URL https://doi.org/10.1145/3272127.3275059. 6.2

[25] Zhimin Fan, Jie Guo, Yiming Wang, Tianyu Xiao, Hao Zhang, Chenxi Zhou, Zhenyu Chen, Pengpei Hong, Yanwen Guo, and Ling-Qi Yan. Specular polynomials. *arXiv preprint arXiv:2405.13409*, 2024. 7.1

[26] Jeremy A Fishel and Gerald E Loeb. Sensing tactile microvibrations with the biotac—comparison with human sensitivity. In *2012 4th IEEE RAS & EMBS international conference on biomedical robotics and biomechatronics (BioRob)*, pages 1122–1127. IEEE, 2012. 2.2

[27] Daniel Fernandes Gomes and Shan Luo. Geltip tactile sensor for dexterous manipulation in clutter, 2021. 2.1

[28] Daniel Fernandes Gomes, Achu Wilson, and Shan Luo. Gelsight simulation for sim2real learning. In *ICRA ViTac Workshop*, 2019. 2.2, 3.2.3, 3.1

[29] Daniel Fernandes Gomes, Shan Luo, and Paolo Paoletti. Beyond Flat GelSight Sensors: Simulation of Optical Tactile Sensors of Complex Morphologies for Sim2Real Learning. In *Proceedings of Robotics: Science and Systems*, Daegu, Republic of Korea, July 2023. doi: 10.15607/RSS.2023.XIX.035. 7.1

[30] Ana Granizo-Hidalgo and Nicolas Holzschuch. Interactive rendering of caustics using dimension reduction for manifold next-event estimation. *Proceedings of the ACM on Computer Graphics and Interactive Techniques*, 7(1):1–16, 2024. 7.1

[31] Jianzhe Gu, Yuyu Lin, Qiang Cui, Xiaoqian Li, Jiaji Li, Lingyun Sun, Cheng Yao, Fangtian Ying, Guanyun Wang, and Lining Yao. Pneumesh: Pneumatic-driven truss-based shape changing system. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*, CHI '22, New York, NY, USA, 2022. Association for Computing Machinery. ISBN 9781450391573. doi: 10.1145/3491102.3502099. URL https://doi.org/10.1145/3491102.3502099. 2.4

[32] Toshiya Hachisuka and Henrik Wann Jensen. Stochastic progressive photon mapping. *ACM Transactions on Graphics (TOG)*, 28(5):1–8, 2009. 6.3.3, 7.1

[33] Toshiya Hachisuka and Henrik Wann Jensen. Parallel progressive photon mapping on gpus. In *ACM SIGGRAPH ASIA 2010 Sketches*, SA '10, New York, NY, USA, 2010. Association for Computing Machinery. ISBN 9781450305235. doi: 10.1145/1899950.1900004. URL https://doi.org/10.1145/1899950.1900004. 6.3.3

[34] Nikolaus Hansen. The cma evolution strategy: A tutorial. *arXiv preprint arXiv:1604.00772*, 2016. 4.3.2, 6.4.5

[35] Barrett Heyneman and Mark R Cutkosky. Biologically inspired tactile classification of object-hand and object-world interactions. In *2012 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pages 167–173. IEEE, 2012. 1.1

[36] Francois R. Hogan, Jose Ballester, Siyuan Dong, and Alberto Rodriguez. Tactile dexterity: Manipulation primitives with tactile feedback. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 8863–8869, 2020. doi: 10.1109/ICRA40945.2020.9196976. 3.3.3

[37] Alain Hore and Djemel Ziou. Image quality metrics: Psnr vs. ssim. In *2010 20th international conference on pattern recognition*, pages 2366–2369. IEEE, 2010. 3.2.3

[38] Qingqin Hua, Vojtěch Tázlar, Alban Fichet, and Alexander Wilkie. Efficient storage and importance sampling for fluorescent reflectance. In *Computer Graphics Forum*, volume 42, pages 47–59. Wiley Online Library, 2023. (document), 3.20

[39] Wenzel Jakob and Steve Marschner. Manifold exploration: A markov chain monte carlo technique for rendering scenes with difficult specular transport. *ACM Transactions on Graphics (TOG)*, 31(4):1–13, 2012. 3.1.1

[40] Wenzel Jakob et al. Mitsuba renderer. 2010. 3.2.3, 6.2, 7.1

[41] Wenzel Jakob et al. BSDFs - Mitsuba 3 — mitsuba.readthedocs.io. 2024. URL https://mitsuba.readthedocs.io/en/latest/src/generated/plugins_bsdfs.html. 6.2

[42] Jiaqi Jiang, Xuyang Zhang, Daniel Fernandes Gomes, Thanh-Toan Do, and Shan Luo. Rotipbot: Robotic handling of thin and flexible objects using rotatable tactile sensors. *arXiv preprint arXiv:2406.09332*, 2024. 2.1

[43] Micah K Johnson and Edward H Adelson. Retrographic sensing for the measurement of surface texture and shape. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1070–1077. IEEE, 2009. 2.1

[44] Micah K Johnson, Forrester Cole, Alvin Raj, and Edward H Adelson. Microgeometry capture using an elastomeric sensor. *ACM Transactions on Graphics (TOG)*, 30(4):1–8, 2011. 2.1, 4.2, 4.2

[45] Csaba Kelemen, László Szirmay-Kalos, György Antal, and Ferenc Csonka. A simple and robust mutation strategy for the metropolis light transport algorithm. In *Computer Graphics Forum*, volume 21, pages 531–540. Wiley Online Library, 2002. (document), 3.17, 3.3.2

[46] Csaba Kelemen, Laszlo Szirmay-Kalos, Gyorgy Antal, and Ferenc Csonka. A Simple and Robust Mutation Strategy for the Metropolis Light Transport Algorithm. *Computer Graphics Forum*, 2002. ISSN 1467-8659. doi: 10.1111/1467-8659.t01-1-00703. 7.1

[47] Mike Lambeta, Po-Wei Chou, Stephen Tian, Brian Yang, Benjamin Maloon, Victoria Rose Most, Dave Stroud, Raymond Santos, Ahmad Byagowi, Gregg Kammerer, et al. Digit: A novel design for a low-cost compact high-resolution tactile sensor with application to in-hand manipulation. *IEEE Robotics and Automation Letters*, 5(3):3838–3845, 2020. (document), 2.1, 2.1, 4.2

[48] Jeongseok Lee, Michael X Grey, Sehoon Ha, Tobias Kunz, Sumit Jain, Yuting Ye, Siddhartha S Srinivasa, Mike Stilman, and C Karen Liu. Dart: Dynamic animation and robotics toolkit. *Journal of Open Source Software*, 3(22):500, 2018. 3

[49] Anat Levin, Daniel Glasner, Ying Xiong, Fredo Durand, William Freeman, Wojciech Matusik, and Todd Zickler. High spatial resolution brdfs with metallic powders using wave optics analysis. 2013. 3.3.3

[50] Jianhua Li, Siyuan Dong, and Edward Adelson. Slip detection with combined tactile and visual information. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7772–7777. IEEE, 2018. 1.1

[51] Rui Li, Robert Platt, Wenzhen Yuan, Andreas ten Pas, Nathan Roscup, Mandayam A Srinivasan, and Edward Adelson. Localization and manipulation of small parts using gelsight tactile sensing. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3988–3993. IEEE, 2014. (document), 2.1, 2.1, 6.4, 6.4.2

[52] Changyi Lin, Han Zhang, Jikai Xu, Lei Wu, and Huazhe Xu. 9dtact: A

compact vision-based tactile sensor for accurate 3d shape reconstruction and generalizable 6d force estimation. *IEEE Robotics and Automation Letters*, 2023. 7.1

[53] Zhonglin Lin, Jiaquan Zhuang, Yufeng Li, Xianyu Wu, Shan Luo, Daniel Fernandes Gomes, Feng Huang, and Zheng Yang. Gelfinger: A novel visual-tactile sensor with multi-angle tactile image stitching. *IEEE Robotics and Automation Letters*, 8(9):5982–5989, 2023. doi: 10.1109/LRA.2023.3302191. (document), 2.1, 2.1

[54] Chao Liu, Srinivasa G Narasimhan, and Artur W Dubrawski. Near-light photometric stereo using circularly placed point light sources. In *2018 IEEE International Conference on Computational Photography (ICCP)*, pages 1–10. IEEE, 2018. 2.2

[55] Sandra Q. Liu and Edward H. Adelson. Gelsight fin ray: Incorporating tactile sensing into a soft compliant robotic gripper. In *2022 IEEE 5th International Conference on Soft Robotics (RoboSoft)*, pages 925–931, Edinburgh, United Kingdom, 2022. IEEE. doi: 10.1109/RoboSoft54090.2022.9762175. (document), 2.1, 2.1, 7.1

[56] Sandra Q Liu and Edward H Adelson. Gelsight fin ray: Incorporating tactile sensing into a soft compliant robotic gripper. In *2022 IEEE 5th International Conference on Soft Robotics (RoboSoft)*, pages 925–931. IEEE, 2022. 3.3.3, 7.1

[57] Sandra Q Liu and Edward H Adelson. A passively bendable, compliant tactile palm with robotic modular endoskeleton optical (romeo) fingers. *arXiv preprint arXiv:2404.08227*, 2024. 7.1

[58] Sandra Q Liu, Yuxiang Ma, and Edward H Adelson. Gelsight baby fin ray: A compact, compliant, flexible finger with high-resolution tactile sensing. In *2023 IEEE International Conference on Soft Robotics (RoboSoft)*, pages 1–8. IEEE, 2023. 2.1

[59] Sandra Q Liu, Leonardo Zamora Yañez, and Edward H Adelson. Gelsight endoflex: A soft endoskeleton hand with continuous high-resolution tactile sensing. In *2023 IEEE International Conference on Soft Robotics (RoboSoft)*, pages 1–6. IEEE, 2023. 2.1

[60] Guillaume Loubet, Nicolas Holzschuch, and Wenzel Jakob. Reparameterizing discontinuous integrands for differentiable rendering. *ACM Transactions on Graphics (TOG)*, 38(6):1–14, 2019. 3.2.2

[61] Guillaume Loubet, Tizian Zeltner, Nicolas Holzschuch, and Wenzel Jakob. Slope-space integrals for specular next event estimation. *ACM Transactions on Graphics (TOG)*, 39(6):1–13, 2020. 7.1

[62] Fujun Luan, Shuang Zhao, Kavita Bala, and Ioannis Gkioulekas. Langevin

monte carlo rendering with gradient-based adaptation. *ACM Trans. Graph.*, 39(4), jul 2020. ISSN 0730-0301. doi: 10.1145/3386569.3392382. URL https://doi.org/10.1145/3386569.3392382. 3.1.1, 3.3.2

[63] Fujun Luan, Shuang Zhao, Kavita Bala, and Ioannis Gkioulekas. Langevin monte carlo rendering with gradient-based adaptation. *ACM Trans. Graph.*, 39(4):140, 2020. 3.3.2

[64] Daolin Ma, Elliott Donlon, Siyuan Dong, and Alberto Rodriguez. Dense tactile force estimation using gelslim and inverse fem. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 5418–5424. IEEE, 2019. 4.2, 4.3.2

[65] Yuxiang Ma, Edward Adelson, et al. Gellink: A compact multi-phalanx finger with vision-based tactile sensing and proprioception. *arXiv preprint arXiv:2403.14887*, 2024. 2.1

[66] Yuxiang Ma, Arpit Agarwal, Sandra Q. Liu, Wenzhen Yuan, and Edward H. Adelson. Scalable simulation-guided compliant tactile finger design. In *2024 IEEE 7th International Conference on Soft Robotics (RoboSoft)*, pages 1068–1074, 2024. doi: 10.1109/RoboSoft60065.2024.10521969. 7.1

[67] Miles Macklin, Kenny Erleben, Matthias Müller, Nuttapong Chentanez, Stefan Jeschke, and Viktor Makoviychuk. Non-smooth newton methods for deformable multi-body dynamics. *ACM Transactions on Graphics (TOG)*, 38 (5):1–20, 2019. 3

[68] MarketsandMarkets. Surface inspection market worth $5.3 billion by 2025 - exclusive report by marketsandmarkets™, Feb 2020. URL https://www.prnewswire.com/news-releases/surface-inspection-market-worth-5-3-billion-by-2025--exclusive-report-by-marke html. 4.5.2

[69] Alan D McNaught, Andrew Wilkinson, et al. *Compendium of chemical terminology*, volume 1669. Blackwell Science Oxford, 1997. 3.3.5

[70] Stephan Mühlbacher-Karrer, Andre Gaschler, and Hubert Zangl. Responsive fingers—capacitive sensing during object manipulation. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4394–4401. IEEE, 2015. 1.1

[71] Yashraj S Narang, Karl Van Wyk, Arsalan Mousavian, and Dieter Fox. Interpreting and predicting tactile signals via a physics-based and data-driven framework. *arXiv preprint arXiv:2006.03777*, 2020. 2.2

[72] Merlin Nimier-David, Delio Vicini, Tizian Zeltner, and Wenzel Jakob. Mitsuba 2: a retargetable forward and inverse renderer. *ACM Transactions on Graphics (TOG)*, 38(6):203, 2019. 3.2.2

[73] Akhil Padmanabha, Frederik Ebert, Stephen Tian, Roberto Calandra, Chelsea Finn, and Sergey Levine. Omnitact: A multi-directional high-resolution touch sensor. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 618–624. IEEE, 2020. 1.1, 2.1, 3.3, 3.3.4, 4.2, 4.3.2

[74] Mary Pagnutti, Robert E Ryan, George Cazenavette, Maxwell Gold, Ryan Harlan, Edward Leggett, and James Pagnutti. Laying the foundation to use raspberry pi 3 v2 camera module imagery for scientific and engineering purposes. *Journal of Electronic Imaging*, 26(1):013014–013014, 2017. 5.1

[75] Francesca Palermo, Liz Rincon-Ardila, Changjae Oh, Kaspar Althoefer, Stefan Poslad, Gentiane Venture, and Ildar Farkhatdinov. Multi-modal robotic visual-tactile localisation and detection of surface cracks. In *2021 IEEE 17th International Conference on Automation Science and Engineering (CASE)*, pages 1806–1811. IEEE, 2021. 4.5.2

[76] Matt Pharr, Wenzel Jakob, and Greg Humphreys. *Physically based rendering: From theory to implementation*. Morgan Kaufmann, 2016. 3, 3.1.1

[77] Branden Romero, Filipe Veiga, and Edward Adelson. Soft, round, high resolution tactile fingertip sensors for dexterous robotic manipulation. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4796–4802, 2020. doi: 10.1109/ICRA40945.2020.9196909. (document), 2.1, 2.1, 3.1.1, 3.3, 3.3.3, 4.2, 4.3.1, 4.3.2, 4.3.2, 6.3.2

[78] Adriana Schulz. *Computational design for the next manufacturing revolution*. PhD thesis, Massachusetts Institute of Technology, 2018. 4.3.2

[79] Adriana Schulz, Jie Xu, Bo Zhu, Changxi Zheng, Eitan Grinspun, and Wojciech Matusik. Interactive design space exploration and optimization for cad models. *ACM Trans. Graph.*, 36(4), jul 2017. ISSN 0730-0301. doi: 10.1145/3072959.3073688. URL https://doi.org/10.1145/3072959.3073688. 2.4

[80] Carmelo Sferrazza and Raffaello D'Andrea. Design, motivation and evaluation of a full-resolution optical tactile sensor. *Sensors*, 19(4):928, 2019. 2.2

[81] Carmelo Sferrazza, Thomas Bi, and Raffaello D'Andrea. Learning the sense of touch in simulation: a sim-to-real strategy for vision-based tactile sensing. *arXiv preprint arXiv:2003.02640*, 2020. 2.2

[82] Yu She, Sandra Q Liu, Peiyu Yu, and Edward Adelson. Exoskeleton-covered soft finger with vision-based proprioception and tactile sensing. In *2020 ieee international conference on robotics and automation (icra)*, pages 10075–10081. IEEE, 2020. 2.1

[83] Yu She, Shaoxiong Wang, Siyuan Dong, Neha Sunil, Alberto Rodriguez, and Edward Adelson. Cable manipulation with a tactile-reactive gripper. *The*

*International Journal of Robotics Research*, 40(12-14):1385–1401, 2021. 4.3.2

[84] Kfir Shem-Tov, Sai Praveen Bangaru, Anat Levin, and Ioannis Gkioulekas. Towards reflectometry from interreflections. In *2020 IEEE International Conference on Computational Photography (ICCP)*, pages 1–12. IEEE, 2020. 3.2.2

[85] Michael A Sherman, Ajay Seth, and Scott L Delp. Simbody: multibody dynamics for biomedical research. *Procedia Iutam*, 2:241–261, 2011. 3

[86] Zilin Si and Wenzhen Yuan. Taxim: An example-based simulation model for gelsight tactile sensors. *IEEE Robotics and Automation Letters*, 7(2):2361–2368, 2022. 2.3, 7.1

[87] Zilin Si, Zirui Zhu, Arpit Agarwal, Stuart Anderson, and Wenzhen Yuan. Grasp stability prediction with sim-to-real transfer from tactile sensing. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7809–7816, 2022. doi: 10.1109/IROS47612.2022.9981863. 2.3

[88] Edward J Smith, Roberto Calandra, Adriana Romero, Georgia Gkioxari, David Meger, Jitendra Malik, and Michal Drozdzal. 3d shape reconstruction from vision and touch. *arXiv*, 2020. 2.2, 3.2.3

[89] Russell Smith et al. Open dynamics engine. 2005. 3

[90] Ian Taylor, Siyuan Dong, and Alberto Rodriguez. Gelslim3.0: High-resolution measurement of shape, force and slip in a compact tactile-sensing finger. *CoRR*, abs/2103.12269, 2021. URL `https://arxiv.org/abs/2103.12269`. 2.1, 3.3.4, 4.2, 4.2, 4.3.1

[91] Ian Taylor, Siyuan Dong, and Alberto Rodriguez. Gelslim3. 0: High-resolution measurement of shape, force and slip in a compact tactile-sensing finger. *arXiv preprint arXiv:2103.12269*, 2021. (document), 1.1, 2.1, 2.3

[92] Art Tevs, Ivo Ihrke, and Hans-Peter Seidel. Maximum mipmaps for fast, accurate, and scalable dynamic height field rendering. In *Proceedings of the 2008 symposium on Interactive 3D graphics and games*, pages 183–190, 2008. 3.2.2

[93] Yunsheng Tian, Ane Zuniga, Xinwei Zhang, Johannes P Dürholt, Payel Das, Jie Chen, Wojciech Matusik, and Mina Konaković Luković. Boundary exploration for bayesian optimization with unknown physical constraints. *arXiv preprint arXiv:2402.07692*, 2024. 7.1

[94] Megha H Tippur and Edward H Adelson. Gelsight360: An omnidirectional camera-based tactile sensor for dexterous robotic manipulation. *arXiv preprint arXiv:2304.04268*, 2023. (document), 1.4, 2.1, 2.1, 6.4, 6.4.4, 6.12

[95] Megha H. Tippur and Edward H. Adelson. Rainbowsight: A family of generalizable, curved, camera-based tactile sensors for shape reconstruction. In *2024 IEEE International Conference on Robot Automation (ICRA)*, pages 1–8,

2024. 2.1

[96] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033. IEEE, 2012. 3

[97] Eric Veach. *Robust Monte Carlo methods for light transport simulation*. Stanford University, 1998. 3.1.1

[98] Eric Veach and Leonidas J Guibas. Metropolis light transport. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 65–76, 1997. 3.3.2

[99] Bruce Walter, Stephen R Marschner, Hongsong Li, and Kenneth E Torrance. Microfacet models for refraction through rough surfaces. *Rendering techniques*, 2007:18th, 2007. 3.2.2

[100] Shaoxiong Wang, Yu She, Branden Romero, and Edward Adelson. Gelsight wedge: Measuring high-resolution 3d contact geometry with a compact robot finger. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6468–6475, 2021. doi: 10.1109/ICRA48506.2021.9560783. 3.3.4

[101] Benjamin Ward-Cherrier, Nicholas Pestell, Luke Cramphorn, Benjamin Winstone, Maria Elena Giannaccini, Jonathan Rossiter, and Nathan F Lepora. The tactip family: Soft optical tactile sensors with 3d-printed biomimetic morphologies. *Soft robotics*, 5(2):216–227, 2018. 1.1, 2.2, 7.1

[102] Jie Xu, Tao Chen, Lara Zlokapa, Michael Foshey, Wojciech Matusik, Shinjiro Sueda, and Pulkit Agrawal. An end-to-end differentiable framework for contact-aware robot design. *arXiv preprint arXiv:2107.07501*, 2021. C

[103] Jie Xu, Tao Chen, Lara Zlokapa, Michael Foshey, Wojciech Matusik, Shinjiro Sueda, and Pulkit Agrawal. An end-to-end differentiable framework for contact-aware robot design. *arXiv preprint arXiv:2107.07501*, 2021. 6.2

[104] Akihiko Yamaguchi and Christopher G Atkeson. Combining finger vision and optical tactile sensing: Reducing and handling errors while cutting vegetables. In *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*, pages 1045–1051. IEEE, 2016. 1.1

[105] Shenli Yuan, Shaoxiong Wang, Radhen Patel, Megha Tippur, Connor Yako, Edward Adelson, and Kenneth Salisbury. Tactile-reactive roller grasper. *arXiv preprint arXiv:2306.09946*, 2023. 2.1

[106] Wenzhen Yuan, Siyuan Dong, and Edward H. Adelson. Gelsight: High-resolution robot tactile sensors for estimating geometry and force. *Sensors*, 17(12):2762, 2017. doi: 10.3390/s17122762. 3.2, 3.2.1, 3.2.3, 3.2.3, 3.1

[107] Wenzhen Yuan, Siyuan Dong, and Edward H Adelson. Gelsight: High-

resolution robot tactile sensors for estimating geometry and force. *Sensors*, 17(12):2762, 2017. 1.1, 2.1, 2.1, 4.2, 4.4.1, 5.1, 6.3.2

[108] Nur'Aqilah Zainuddin, Nur Farahin Anuar, Ahmad Luqman Mansur, Nur Izzati Mohd Fauzi, Wan Fazlida Hanim, and Sukreen Hana Herman. Resistive-based sensor system for prosthetic fingers application. *Procedia Computer Science*, 76:323–329, 2015. 1.1

[109] Tizian Zeltner, Iliyan Georgiev, and Wenzel Jakob. Specular manifold sampling for rendering high-frequency caustics and glints. *ACM Transactions on Graphics (TOG)*, 39(4):149–1, 2020. 3.1.1

[110] Zheng Zeng, Lu Wang, Bei-Bei Wang, Chun-Meng Kang, and Yan-Ning Xu. Denoising stochastic progressive photon mapping renderings using a multi-residual network. *Journal of Computer Science and Technology*, 35:506–521, 2020. 7.2

[111] Jinhui Zhang, Haimin Yao, Jiaying Mo, Songyue Chen, Yu Xie, Shenglin Ma, Rui Chen, Tao Luo, Weisong Ling, Lifeng Qin, et al. Finger-inspired rigid-soft hybrid tactile sensor with superior sensitivity at high frequency. *Nature Communications*, 13(1):5076, 2022. 1.1

[112] Mabel M Zhang, Monroe D Kennedy, M Ani Hsieh, and Kostas Daniilidis. A triangle histogram for object classification by tactile sensing. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4931–4938. IEEE, 2016. 2.2

[113] Shixin Zhang, Yuhao Sun, Fuchun Sun, Yiyong Yang, and Bin Fang. Pfs 1.0: A development tool applied to vision-based tactile sensor process formulation and fabrication. *Sensors and Actuators A: Physical*, 367:115090, 2024. 2.3

[114] Allan Zhao, Jie Xu, Mina Konaković-Luković, Josephine Hughes, Andrew Spielberg, Daniela Rus, and Wojciech Matusik. Robogrammar: Graph grammar for terrain-optimized robot design. *ACM Trans. Graph.*, 39(6), nov 2020. ISSN 0730-0301. doi: 10.1145/3414685.3417831. URL https://doi.org/10.1145/3414685.3417831. 7

[115] Jialiang Zhao and Edward H Adelson. Gelsight svelte: A human finger-shaped single-camera tactile robot finger with large sensing coverage and proprioceptive sensing. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 8979–8984. IEEE, 2023. (document), 1.4, 2.1, 2.1, 6.4, 6.4.5

[116] Yongqiang Zhao, Kun Qian, Boyi Duan, and Shan Luo. Fots: A fast optical tactile simulator for sim2real learning of tactile-motor robot manipulation skills. *IEEE Robotics and Automation Letters*, 2024. 7.1

[117] Yinqiang Zheng, Imari Sato, and Yoichi Sato. Spectra estimation of fluorescent

and reflective scenes by using ordinary illuminants. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 188–202. Springer, 2014. 3.3.5

[118] Lara Zlokapa, Yiyue Luo, Jie Xu, Michael Foshey, Kui Wu, Pulkit Agrawal, and Wojciech Matusik. An integrated design pipeline for tactile sensing robotic manipulators. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 3136–3142, 2022. doi: 10.1109/ICRA46639.2022.9812335. 2.4