Title
Deformation in Robotics: Applications in Tactile Sensing and Grasp Planning

Author
Huang, Isabella

Publication Date
2023

Peer reviewed|Thesis/dissertation

Deformation in Robotics: Applications in Tactile Sensing and Grasp Planning

by

Isabella Huang

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Doctor of Philosophy

in

Engineering – Electrical Engineering and Computer Sciences

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Ruzena Bajcsy, Chair
Professor Shankar Sastry
Professor Hannah Stuart

Spring 2023

Deformation in Robotics: Applications in Tactile Sensing and Grasp Planning

Abstract

Deformation in Robotics: Applications in Tactile Sensing and Grasp Planning

by

Isabella Huang

Doctor of Philosophy in Engineering – Electrical Engineering and Computer Sciences

University of California, Berkeley

Ruzena Bajcsy, Chair

If robots are to function in unconstrained, dynamic, and real-world environments, it is critical that are able to interact with deformable materials. Deformable materials have historically been overlooked in traditional robotics due to prevailing assumption of robot and object rigidty. These assumptions, though perfectly appropriate for constrained environments such as factory settings, are often broken in the real-world. At the same time, deformable interactions are uniquely challenging due to the infinite-dimensional, non-linear nature of the deformable materials involved. To address the challenges of introducing and implementing deformation in robotics, we present this thesis work in two parts. First, we study deformation on the *robot* side, where we design a soft tactile sensor and demonstrate its use in human-robot interaction and automation. Second, we also explore deformation on the *object* side, where in particular we focus on optimizing grasp strategies over 3D field quantities.

In our first group of work, we propose and fabricate a novel soft tactile device that utilizes an embedded 3D depth-sensing camera to produce interpretable signals for geometry and force sensing. We demonstrate that this sensor is inherently safe and functional for applications including physical upper limb assistance for humans, contour following for domestic wiping tasks, as well as geometry-dependent learning from demonstration tasks for general contact-rich manipulation tasks. In the second part of the thesis, we develop grasp planners for 3D deformable objects (e.g., fruits, internal organs, containers) for applications in food processing, robotic surgery, and household automation. In particular, we optimize for field quantities that are not only inaccessible in the real world, but have also been, until recent years, computationally intractable to model. We create DefGraspSim, a finite element method-based physics simulator of arbitrary grasps on arbitrary 3D meshes over a wide range of material parameters. We also create DefGraspNets, a graph neural network-based forward dynamics model that is not only up to 1500x faster than DefGraspSim, but also enables gradient-based grasp optimization. For both methods, we demonstrate generalized performance across multiple test sets, including on real-world experiments.

To my grandfather, who believed in me through and through.

Contents

List of Figures

List of Tables

Acknowledgments

The PhD has undoubtedly been the most formative, challenging, and motivating chapter of my academic career. I am truly fortunate to have been able to devote these past five years to chasing my academic interests, diving headfirst into new ideas, and tackling the daunting objective of "original research." Although the PhD has at times felt bleak and difficult, the privilege of being able to address some of the hardest engineering problems in the world (in collaboration with some of the brightest minds in the world), has always kept me afloat. As I celebrate the culmination of my PhD work in this thesis, I extend my deepest gratitude to the many people who journeyed with me along the way.

First and foremost, I thank my advisor Ruzena Bajcsy from the bottom of my heart, without whom all this would be impossible. She believed in me and gave me the once-in-a-lifetime opportunity to join her academic family– I consider this one my life's biggest blessings. A force to be reckoned with, Ruzena exudes an aura of staunchness that only decades of experience can draw out. How lucky I felt knowing that I could be under her wing. She would encourage us to "go the other way" and be bold with our ideas, even when I felt like I didn't even have any ideas to show for myself. As my academic mother, Ruzena continuously endowed me with renewed perspectives on my research projects, greater academia, and most importantly, life itself. I will forever be grateful and honored to be her final student.

I also thank my committee members Shankar Sastry, Hannah Stuart, and Ronald Fearing. They contributed significantly to my research vision, both through their honest criticisms and also through their deep scientific intuitions towards interdisciplinary directions I had never even imagined. From them, I learned to consider the big picture, critically question the purpose of a project, and even be willing to let go of ideas that just don't make sense, no matter how fond of them I may have become. I also thank professors Claire Tomlin and Anca Dragan, who are leaders in their respective fields of control theory and algorithmic human-robot interaction. It was under their instruction that I felt that first spark for research, and I have undoubtedly drawn a significant amount of my research inspiration from what I learned in their classrooms. The PhD would also not have been possible without Shirley Salanio, who is undeniably everyone's favorite person in the department. As my graduate student advisor, she kept me on track, helped me navigate the trickiest of administrative situations, and above all, deeply cared for her students as people. I am also grateful for Leslie Goldstein and Jessica Gamble, who helped support the Bajcsy group with finances and facilities and ensured that we had the most productive working environment possible.

None of this work would be possible without the support of the fellow members of the Bajcsy group, Robert Matthew, Sarah Seko, Michael Estrada, Laura Hallock, Zoe Cohen, and Carolyn Matl. Thank you for fostering a collaborative, chaotic, and fun lab environment. There was always the most going on in our lab with our all fun gadgets, sensors, and robot setups. I thank Carolyn in particular for being my closest friend who is wise and kind beyond her years. It was also an incredible opportunity to collaborate on projects with Ravi Pandya, Sandy Huang, Edson Araujo, Michal Gregor, Erickson Nascimento, Jingjun Liu, and Dylan Chow. Outside of research, I thank my fellow EECS 106A GSIs for the incredible teaching experience we shared together. In particular, I thank head GSIs Valmik Prabhu for his doggedness throughout any teaching challenge as well as Jaeyun Stella Seo for exemplifying

the deepest and most genuine care for her students. I thank my department mates with whom I shared a camaraderie– the wonderful women of WICSE, my prelim study group, and my dearest department friends with whom I could laugh all my worries away. Jeffrey Chan, Armin Askari, Ashish Kumar, Ellis Ratner, Kelvin Xu– thank you for being there every step of the way. I am extremely grateful to Jeffrey for his companionship in a program that had previously felt cold and isolating, as well as to Armin for being a living reminder that the good days are right here.

I consider my PhD incomplete without the extremely instructive and principled work we did at the NVIDIA Seattle Robotics Lab. My collaborators and mentors Yashraj Narang, Dieter Fox, Tucker Hermans, Balakumar Sundaralingam, Clemens Eppner, Miles Macklin, and Fabio Ramos are some of the brightest research minds I've ever had the honor of working with, and they remind me that this degree is just the beginning. I am indebted to Yashraj Narang, my closest collaborator and mentor with a heart of gold. Thank you for your patience, initiative, and excellence. I also thank Tucker for his invaluable robotics knowledge and for being such a role model in every aspect of life.

To my church friends who made Berkeley feel like home– Matthew Kim, Jessica Wang, Michael Zhao, Janie Lee, Cassie Chiang, Stephanie Kelly, Libby Kao– thank you for being the light of my time here and for reminding me of life's greatest joys. I thank Eleanor Siow for sharing a home with me for three years and for experiencing with me all the highs and lows of young adulthood. To my fellow Canadians who by also relocating to the USA made it feel like I never left home– Judy Shen, Kevin Zhu, Angela Hu, Rahul Chandan, Uma Rajpurkar, Bill Zhao, David Simons, and Barry Fung– I thank you for being my core support network and I am proud of everything you've accomplished in these last five years. To Judy– since the day we first learned how to code more than a decade ago, we have inexplicably stayed in sync; thank you for knowing my journey better than almost anyone else in the world, and I hope that never changes. I also thank my friends from home for their companionship and care that transcends all time zones: Gloria Wu, Truman Tai, Sitan Wang, Fion Cheung, Ryan Cheung, Nisha Kansal, Zoe Song, Rui Janson, Jenson Lam, and especially my oldest friends Ruth Chiu and Averley Huang.

Finally, I extend my deepest gratitude to my family for all of their love and support. I thank my parents Lixia Ma, Zhenfa Huang and grandmother Binglan Pan for their unending support and for providing me the privilege to do what I love. I will never be able to repay them for their sacrifices that enable me to be here today. I thank my sister Samantha Huang for being a constant source of joy and laughter– everywhere I am with you feels like home. I also thank my wonderful in-laws for their continued support, and to Joanne Lee for being the best sister-in-law I could ask for. All my appreciation is not enough to fully thank my incredible husband Ivan Lee for his devotion and love over all these years. Thank you for giving me the courage to challenge myself, embrace failure, and never give up. I thank my late grandfather for being the wind in my sails in all of my endeavors, and I carry his memory in everything I will ever do. I thank my kitty Maisie who during our time in Berkeley slept in the crook of my arm daily and sat on my desk with me through all of the quiet, long nights of work. I miss you and hope you are well. Above all, all glory and thanks be to God who sustained me all of these years and continues to lovingly shepherd me through life.

Chapter 1

Introduction

There has been a long-standing assumption in robotics that robots themselves, as well as the objects with which they interact, are rigid. This is a convenient assumption that ignores any shape change of these materials under applied forces, thereby drastically simplifying all associated modeling, perception, and control applications into 6 dimensions comprising 3D position and 3D orientation. For example, the absence of shape change simplifies the problem of rigid object perception into a 6D pose estimation. The control of rigid manipulators is also relatively simple, as each link maximally has only 6 degrees of freedom, wherein all elements of the link's body undergoes the same 6D transformation during control regardless of what joint forces are applied. Within situations where the assumption of rigidity holds, we have been able to see remarkable successes in applications such as industrial manufacturing, space exploration, and self-driving.

In the real-world and especially in the home, however, this assumption of rigidity is often violated. For example, in the home setting, we may want robots to be soft, as the inherent compliance of soft materials enables safe interactions with delicate environments and humans. In contrast, rigid robots are materially stiff and are prone to applying immense forces upon physical contact. As another example, many in-home objects such as food, textiles, and bodies are materially deformable and require specialized physical models that either extend or replace those for rigid bodies.

In this thesis, we aim to develop solutions and applications for which rigidity no longer holds true. We explore two classes of problems towards incorporating deformability in robotics– first, where deformation is introduced on the **robot side (Part I)**, and second, where deformation is introduced on the **object side (Part II)**. In both parts, we also discuss relevant applications towards real-world, in-home automation.

Part I introduces the design and applications of a soft tactile sensor for use in the home and with humans. We seek to answer the following question:

> *What applications towards in-home assistance can we achieve using soft tactile sensing on the robot?*

Chapter 2 discusses its design and capabilities, wherein the use of an embedded depth camera enables a simple design that simultaneously outputs rich contact signals that can be interpreted as physically salient geometry and force readings. The subsequent chapters study

several applications of this sensor in real-world tasks– physical human-robot interaction for upper-limb assistance (Chapter 3), contour following for wiping (Chapter 4), and generalized learning from demonstration with tactile signals (Chapter 5).

Part II focuses on handling deformation on the object side rather than on the robot side. Here, the question we are trying to answer is the following:

> *How can we create grasp planners for deformable objects, for which rigid models and rigid grasp planners fail?*

Unlike rigid objects, deformable objects are notorious hard to model and control. When grasped, deformable objects undergo shape change, which in turn elicits physical responses in the form of fields such as deformation and stress fields. These fields are important metrics for optimal grasp planning (e.g., we may want to minimize the stress fields when grasping a deformable fruit in order to avoid bruising), yet are simultaneously extremely difficult to model since they have infinite degrees of freedom. To plan grasps over these fields as optimization metrics, we first introduce a finite element method-based solver named DefGraspSim. In Chapter 6, we explore its use as a forward pass dynamics simulator to be used in offline grasp planning of deformable objects, and verify its real-world reliability with real experiments. In Chapter 7, we introduce a graph neural network-based simulator named DefGraspNets, which is a differentiable surrogate model for DefGraspSim. DefGraspNets is not only around 1500x faster than DefGraspSim in its forward pass, thereby enabling online grasp planning, but also differentiable so that candidate grasps can be refined and optimized in its backwards pass. While there exists a speed-accuracy tradeoff between DefGraspSim and DefGraspNets, we demonstrate that DefGraspNets can still be used to plan optimal grasps for real objects.

As both computational capabilities and our understanding of complex contact interactions continue to improve, it is the hope that working with deformable materials in robotics will eventually no longer be treated as a clunky, intractable problem for which simplifying assumptions are desperately needed. I believe that a robot that is truly capable and intelligent should find itself very comfortable in the world of deformables, and I hope this work can be a stepping stone towards that reality.

Part I

Soft Tactile Sensing for In-Home Applications

Chapter 2

Depth Camera-Based Soft Sensing

This chapter is adapted in part from "A Depth Camera-Based Soft Fingertip Device for Contact Region Estimation and Perception-Action Coupling" [61], written in collaboration with Jingjun Liu and Ruzena Bajcsy.

2.1 Introduction

It is widely believed that traditional rigid robots are functionally incompatible within unstructured, cluttered settings [107][66]. At the same time, many soft-bodied biological organisms equipped with rich, complex tactile feedback are adept at performing adaptive and compliant maneuvers in these environments [166]. Researchers are therefore striving to enable similar perception-actuation capabilities in engineered robots [75] to meet safety and flexibility criteria in settings such as exploration [119], surgery [20], and physical human-robot interaction [135].

Our work in particular is inspired by the human fingertip, a critical and sensitive body part endowed with around 241 mechanoreceptors per square centimeter area [24]. The dense sensory information received from the fingertip, coupled with general movement of the finger, allows humans to perform not only "perception for action" (eg. dexterous manipulation), but also "action for perception" (eg. object geometry recognition) [24]. This capability of coupled perception-action, as exemplified in the human fingertip, is the driving design principle behind our robotic device. Although the actuation of soft mechanisms is an area of active research, the contributions of our work here are more pertinent to the development of soft tactile sensing.

The objective of tactile sensing is to physically interact with the external environment and extract knowledge about it by interpreting salient signals. To this end, many existing soft tactile sensing designs in the past have relied on embedding photoreflective, piezoelectric [26], strain-sensitive [32], electromagnetic [161], and other electronic modules [97] within a compliant material. Such designs are easily arranged into a flat skin-like configuration that then covers a robot body to enable tactile sensing capabilities for interaction with humans or the environment. One main advantage of such skins is that the spatial resolution can be made to be as fine as around the millimeter scale, depending on the engineered density

of sensing elements [3]. However, these constituent sensors limit the material strain and compliance that make soft sensors desirable [35]. Moreover, even if the skin itself is soft, overlaying it on top of a hard robot chassis severely limits the achievable compliance during interaction, which poses a safety threat when humans are involved.

Since high-compliance interaction is of the utmost importance for safety, in this work we do not improve traditional tactile skins further, but rather develop a new design for a soft robot end effector interface. An example of existing work based on a similar design goal is the development of a pair of pneumatically pressurized fingers that can carefully pick up fragile objects by sensing an overall pressure change in the fingers upon contact [74]. However, this sensory information is low in resolution, and can neither sense the objects' shapes nor even localize the point of contact. Recently, the leading technique in tactile sensing that does not require embedded sensing elements is optical tactile sensing, in which devices image visual cues from a deformed compliant membrane to reconstruct the interaction effects between the environment and the sensor. These techniques involve reconstructing the 3D tactile membrane using images from a 2D camera by use of tracking dots [54][38][18][179], and/or by taking advantage of optical properties in different media [88][68][145]. Some well-known examples of soft optical tactile sensing devices are the TacTip sensors, which image the distortion of internal papillae of a soft gel-filled membrane, as well as the extremely high-resolution MIT GelSight sensor, which makes use of the superposition of different colored light waves in a gel medium to infer contact information [89].

Existing tactile sensor designs tend to bear a trade-off between accuracy and ease of manufacturing. However, in the advent of modern miniature depth-sensing cameras, which can directly and accurately image membrane geometries as plug-and-play solutions, the extremities of this trade-off are mitigated. Thus, we believe it is worth investigating the function of these small depth cameras as a perception tool for soft sensing devices. Moreover, since such a device requires only an embedded camera, we can explore the modulation of internal stiffness using pneumatics, which can modulate interaction impedances [165]. Furthermore, its few embedded elements implies that the design is easy to manufacture and also extremely scalable. While the lower bound on its geometry is restricted by the range of state-of-the-art depth camera technology and thus renders our design ineffective for very fine manipulation tasks, its larger geometry actually makes it well suited for human-level interaction. In this chapter, we discuss the design of such a soft sensor in Section 2.2, and demonstrate its capabilities for geometry and force sensing in Sections 2.3 and 2.4.

2.2 Sensor Design

Fig. 2.1 is a sketch of the components and dimensions of the first version of the proposed soft design. It is comprised of a CO_2-pressurized capsule mounted on a linear actuator. The capsule is composed of a 2 mm thick elastic hemispherical shell, which equips the with sensing capabilities, sealed onto the rim of a rigid cylindrical housing unit 3D printed from polylactic acid filament. The elastic membrane was molded with Ecoflex^{TM} 50 silicone, with a slightly elongated skirt extending from the hemispherical base to allow fixture onto the housing unit rim with cable ties. The unit also houses a PMDTec CamBoard pico flexx, presently the smallest commercially available depth-sensing camera with a quoted minimum

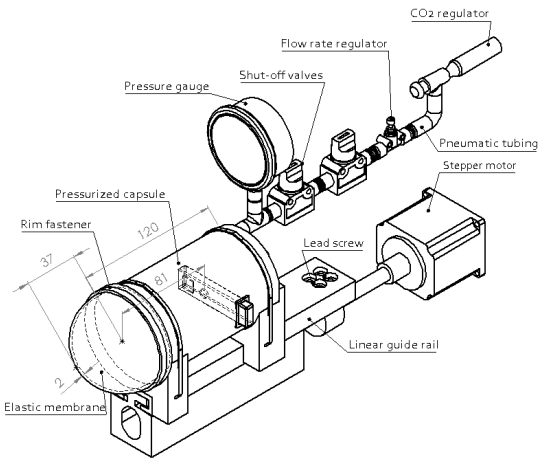


Figure 2.1: Sketch of the components and dimensions [mm] of the first version of the soft tactile sensor. ©2019 IEEE.

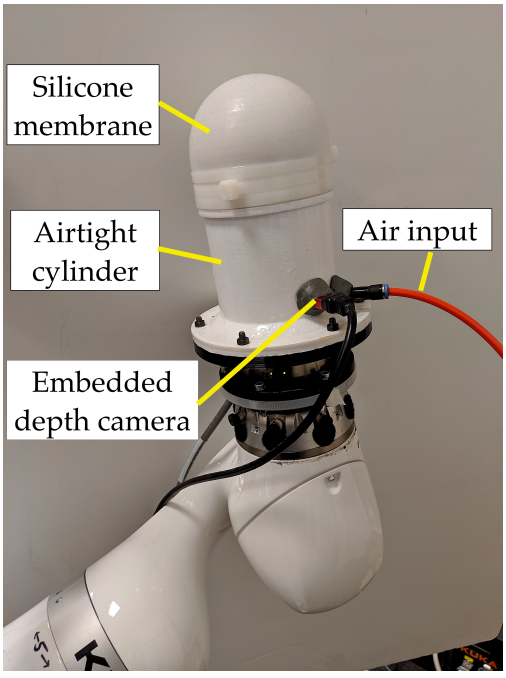


Figure 2.2: Labeled photograph of the sensor mounted as the end effector of a KUKA iiwa 14 manipulator. ©2022 IEEE.

range of 10 cm, and fixes it to be axially centered with the pole of the . The distance from the camera to the membrane was experimentally selected to optimize for visibility and field of view. Furthermore, the depth camera operates at 500 fps at an exposure time of 50 μs, as optimized for performance at small ranges. Connected to the capsule is a series of pneumatic components to manually control and monitor the pressurization of the capsule using a 16 g CO_2 cartridge. This pneumatic system enables the modulation of stiffness by altering the internal gaseous state of the capsule. Modelling clay and epoxy sealant was also used where necessary to eliminate air leakage. Originally, this design was actuated via non-soft techniques using a linear rail powered by a high-torque NEMA 23 stepper motor attached to a lead screw.

The second and final version of this sensor is designed to be mounted onto the end of a LBR iiwa 14 R820 robot arm from KUKA AG with its constituent components labeled in Fig. 2.2. It is larger in size than the initial version, with a membrane outer and inner diameter of 100 and 94 mm respectively with a uniform thickness of 3 mm. A 30 mm-long skirt extending from the hemisphere's base allows the silicone membrane to be securely fastened to the cylinder, with three external cable ties tightly enforcing a hermetic seal. The cylinder surface is fully covered with a coat of XTC-3D®resin to prevent air leakage, and the interior of the cylinder is painted with a matte black acrylic paint to absorb any stray reflected rays from the camera laser that lead to distorted camera readings. Ports on the cylinder that allow for the connection of the pico flexx USB cable as well as the pneunatic tubing are also sealed with the resin and additional modeling clay where necessary. An external electric air compressor enables the modulation of pressure state inside the sensor. Since the tactile membrane is designed to be thin so that it would be sensitive enough to contact-induced deformations, it inflates slightly upon internal pressurization. Thus, each internal pressure state is linked to a certain amount of inflation. In later applications of physical human-robot interaction for upper limb support, contour following, and learning from demonstration, all development utilized the sensor at one fixed pressure state, at which the membrane's inner radius was inflated to 50 mm, compared to its neutral 47 mm radius when free air exchange between the capsule and the atmosphere is permitted. All sensing and actuation is connected to Robot Operating System (ROS) nodes. Since the KUKA robot does not have native ROS support, an open source ROS stack is used to control the KUKA's joint positions [51] and the TRAC-IK library is used for computing inverse kinematics [12]. As used in later experiments for accessing ground truth force-torque readings at the wrist, an ATI Axia 6-dof transducer is mounted between the end of the robot arm and the tactile sensing module.

2.3 Geometry Sensing

We now explain how the sensor's sensory data, an organized point cloud representation of the elastic membrane's physical shape, allows for both the geometric reconstruction of its contact region with an obstacle as well as a recovery of the forces applied to the sensor.

Theory

Upon contact of the membrane with an external obstacle, the goal of contact region reconstruction is to estimate the geometry of the obstacle from the sensory data, which in this case is the point cloud stream from the camera. Since the imaged elastic membrane deforms to the shape of the obstacle it touches, the problem of reconstructing the contacted obstacle boils down to the identification of surface elements on the cloud at which contact with the obstacle is made. It is important to note that not all surface elements that are deformed post-contact are themselves in contact with the obstacle. This is because tensional forces can be induced in non-contact elements whilst the membrane settles at an equilibrium state of minimal elastic energy [38].

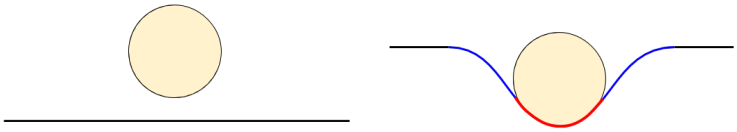
Therefore, each surface element on the membrane during physical interaction with an obstacle can be classified in exactly one of three categories:

- **Undeformed (Type A)**: Surface element maintains its geometry and location pre- and post-contact.
- **Deformed and Non-Contacted (Type B)**: Surface element is not in contact with the obstacle, but undergoes a non-rigid transformation post-contact.
- **Deformed and Contacted (Type C)**: Surface element is both in contact with the obstacle and undergoes a non-rigid transformation post-contact.

Fig. 2.3 illustrates an example of these surface types. In our configuration, it is straightforward to distinguish deformed surface area elements, since the known geometry of the hemispherical shell allows for a pose comparison between any surface area element pre- and post-contact. To further discern a Type B from a Type C surface element, we adopt the contact region estimation method presented by Ito et al. [68], who developed a fluid-filled hemispherical shape-sensing tactile shell similar in geometry to ours. They showed that for a soft membrane that is comprised entirely of convex surface elements when not in contact with an object, like in the case of our sensor's hemispherical shell, a change in the sign of surface convexity signifies that the region is subject to external forces from an obstacle. We thereby follow the computationally efficient method presented in [128] to estimate the local curvature signal κ of a curve between points at $\vec{x}$ and $\vec{y}$ with normal vectors $\hat{n}(\vec{x})$ and $\hat{n}(\vec{y})$ as the following:

$$\kappa(\vec{x}, \vec{y}) = \langle \vec{x} - \vec{y}, \hat{n}(\vec{x}) - \hat{n}(\vec{y}) \rangle \tag{2.1}$$

In order to calculate the normal vectors at each point in the cloud, we apply normal estimation for organized point clouds using integral images as implemented in the Point Cloud Library (PCL) [144]. Then, to reason about the curvature of the entire surface area element, we select two surface curves passing through its center along the two principal planar axes, and calculate the curvature from their endpoints using Eq. 2.1. This technique is illustrated in Fig. 2.4, and the pseudocode for classifying any point in software is presented in Algorithm 1. In this formulation, p defines the dimensions of the surface element. This parameter p should be chosen to be big enough to reduce false classifications on a pre-contact membrane, but small enough to be accurate during post-contact estimation.



(a) Pre-contact (b) Post-contact

Figure 2.3: Upon contact with a spherical obstacle, an elastic membrane can be partitioned into regions of Type A (black), Type B (blue), and Type C (red). ©2019 IEEE.

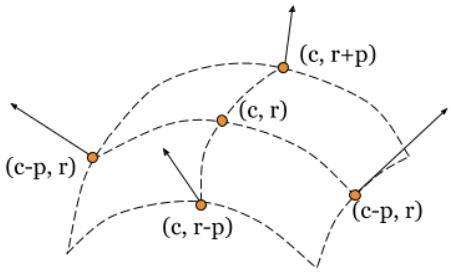


Figure 2.4: Two surface curves are sampled from each surface area element to characterize the concavity of the element. ©2019 IEEE.

Algorithm 1 Classify element centered at indices (c, r) of organized point cloud

$x\_neighbors \leftarrow (c-p, r), (c+p, r)$
$y\_neighbors \leftarrow (c, r-p), (c, r+p)$
if IsUndeformed(c, r) **then**
  return A
else if Convexity$(x_neighbors) \geq 0$ or Convexity$(y_neighbors) \geq 0$ **then**
  return C
else
  return B
end if

Furthermore, the success of this procedure hinges on the following reasonable assumptions:

1. Contact surfaces of an obstacle are themselves flat or convex. Otherwise, the membrane cannot be guaranteed to deform upon contact, which is a necessary condition for the sensor's optical perception mechanism.

2. The inner pressure of the membrane is larger than the outer pressure so that changes in the sign of concavity are undoubtedly from external obstacle normal forces rather then a balance of internal pressure and deformed tensional forces in the membrane.

3. The thickness of this particular membrane material is low enough such that the bending moment is negligible. This criteria is fulfilled by design, as the thickness is only around 5% of the shell radius.

4. The contact area of an obstacle is larger than a single surface element in the membrane. Otherwise, deflections in the membrane will not be adequately captured in the normal vector calculations.

Results

The goal of this experiment is to analyze the performance of the contact region reconstruction methods in the actual sensor device. In the experiment, the sensor contacts three different rigid obstacles mounted on the force sensor starting from a pre-contact state at $PV = 70$ Nm at a pole indendation depth of 10 mm. Two of these obstacles, the rectangular prism and the ring, has flat surfaces. The sphere, however, is a convex but non-flat obstacle. Following the method outlined above, each surface element as defined by the points in the organized point cloud are classified as one of Type A, B, or C. The results of the 2D projection of the surface classification is shown in Fig. 2.5. The surface area element size is chosen to minimize false positive classification as a contact area in the pre-contact case. Using the pico flexx camera, this corresponds to a $p = 10$ value as referred to in Algorithm 1. We see that there is good correspondence between the geometry of the estimated contact region with that of the real obstacle, especially for the flat rectangle and ring obstacles. However, the sensor underestimates the area of the convex spherical object as a result of our use of an internal compressible gas. Since volume is not preserved upon contact, the membrane is unable to fully hug the boundary of the contact sphere. Thus, though the flat obstacles yields a maximum boundary detection error of approximately 1 mm, this error is around 3 mm for the sphere. The results suggest that although the specific shape of an obstacle may constrain the amount of geometric information that can be gleaned from any singular contact state, the region of the membrane that actually is in contact with an obstacle can be measured very well. Thus, approaching an obstacle from only one angle is generally insufficient in characterizing its geometry. This is no different from our own biological fingertips, in that objects of interest usually have to be probed from many different directions before we can infer their structures. When the sensor probes an object in 3D, it is also possible to concatenate sequential readings to build up a fully 3D representation of the object (see Fig. 2.6 for example readings on everyday objects).

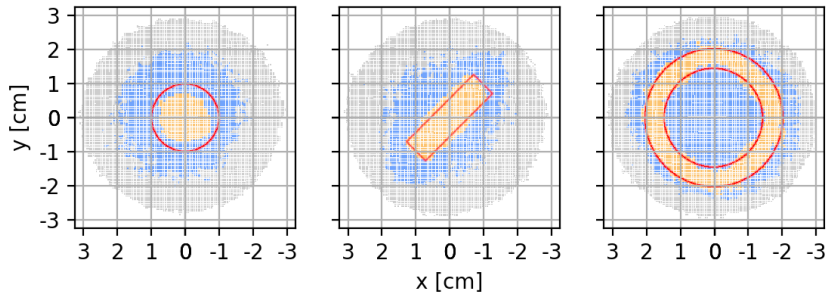
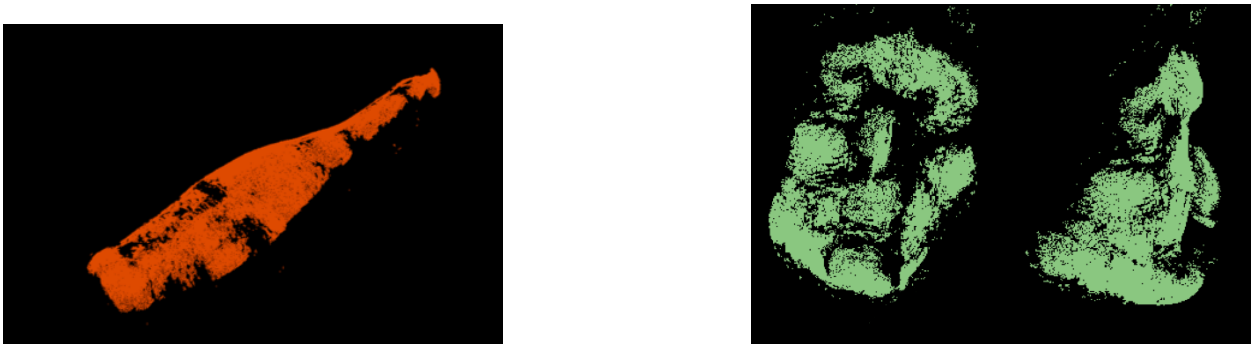


Figure 2.5: Classification of point cloud surface elements projected onto the xy-plane. Here, the surface element center, rather than the full surface element, is plotted as either Type A (gray), Type B (blue), or Type C (yellow) in order to depict the spacing between adjacent points in the cloud. The borders of the real obstacle are shown in red. The projections shown here are limited to a radius of 3 cm, since that was large enough to entirely contain the contact regions. ©2019 IEEE.



(a) Glass bottle

(b) Face (front and side view)

Figure 2.6: Examples of geometry reconstruction of two real-world 3D objects. ©2022 IEEE.

Sensitivity to Pressure

Though contact area estimation yields good results compared to the real geometry of the obstacle, the data was only collected for a fixed deformation and initial gas state. Since real world applications are seldom so constrained, we aim to quantify how sensor action (i.e., how it interacts with the environment by modulating PV states and actuating linearly) impacts perception (i.e., what sensory information can be obtained from the membrane via the camera). In this experiment, the 3D estimated contact surface area is plotted as a function of the indentation depth for the three spherical obstacles at different PV values, as shown in Fig. 2.7. From this data alone, there are no clear differences in estimated surface area for different initial gas states, as long as no free air exchange is allowed with the external environment. This is a useful result, as it implies that contact estimation can be performed with the same parameters regardless of the sensor's initial gas state. We also

see that the estimated surface area generally tends to increase with the indentation depth. This is expected due to the increasing cross-sectional area of each sphere encountered by the membrane as it sinks into the sensor. At the same time, we previously observed the membrane's inability to completely wrap itself around spherical obstacles. Thus, the fact that the estimated contact area never attains a value as high as half of the contact sphere's full surface area is justified. Though not explored in detail within this work, these trends are not directly applicable to obstacles with sharper boundaries (e.g., if the sensor were to contact the face of a rectangular prism).

The relationship between indentation depth and the estimated surface area of all deformed points (i.e., those of either Type B or C) is also analyzed. In Fig. 2.8, this estimated deformation surface area is plotted for the same indentation depths as in Fig. 2.7. We see that with lower initial pressure and expansion of the sensor, the bending moment of the membrane is not counterbalanced as strongly by the internal pressure, and therefore the deformation is greater upon contact with an obstacle. This implies that a lower density of the internal gas allows for the sensor to be more sensitive (i.e., undergo larger deformations) upon obstacle contact.

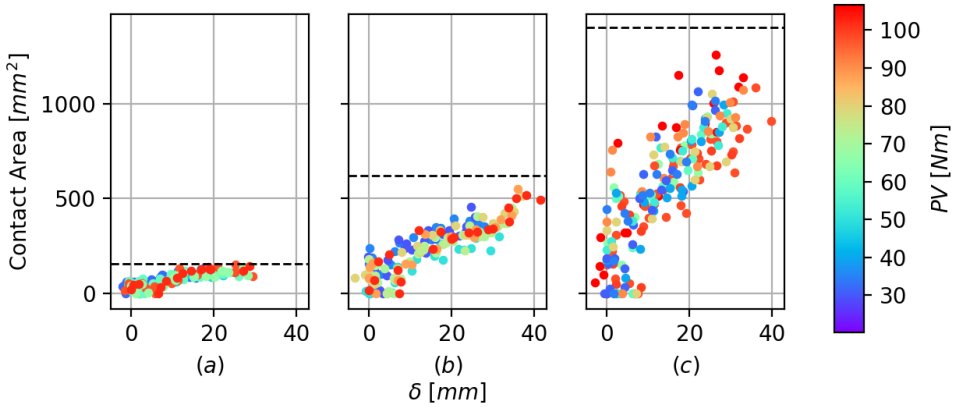


Figure 2.7: 3D contact surface area estimates upon contact with spheres of diameters a) 10, b) 20, and c) 30 mm at different indentation depths and pre-deformation PV states. The black dotted line denotes half of the total surface area of the actual sphere. ©2019 IEEE.

2.4 Force-Deformation Characteristics

Theory

In physics theory, the constitutive equations of continuum elastic materials relates the stress and strain tensors over the membrane body. In the case of the soft sensor, this intrinsic relationship between stress and strain means that the membrane configuration at any contact state encodes precise information about the applied and internal forces being applied to it.

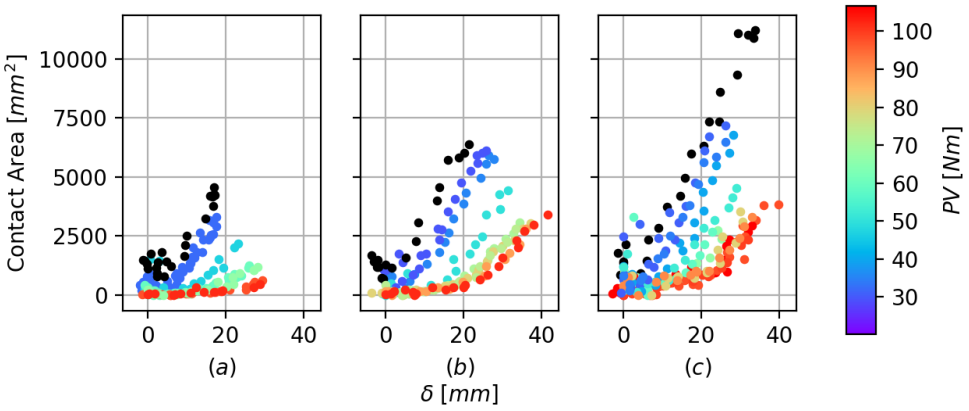


Figure 2.8: 3D deformation surface area estimates upon contact with spheres of diameters a) 10, b) 20, and c) 30 mm at different indentation depths and pre-deformation PV states. Results when the capsule is allowed free air exchange with the atmosphere are plotted in black. ©2019 IEEE.

In this section, we discuss which methods work best in practice for finding the applied forces on the membrane given its deformed configuration.

This is not a simple task, as recovering the membrane loads that transformed it from its initial body shape to a subsequent deformed one requires complex procedures [171] such as finite element modelling and other numerical optimization techniques [151]. Although elastic shell theory can be leveraged in solving this inverse elastostatic problem given our compatible sensor configuration [182], it is still too computationally demanding to be applied in robotic applications that require real-time feedback and control. For this reason, we follow the approach of similar devices and utilize approximate but much simpler models to characterize the force-deformation characteristics of our sensor. As shown by Vella et al [169], the indentation of pressurized elastic shells exhibit two near-linear force-deformation regimes at small and large indentation depths relative to the shell thickness respectively. Their setup was similar to ours in that upon contact, the module preserves neither volume nor internal pressure. In addition, a previous iteration of the sensor in our group similarly verified near-linear characteristics that matched up very well with the analytical Hertz contact model [5]. Therefore, we maintain an approximation of the tactile sensor at each internal gas state as an equivalent linear isotropic elastic sphere. We also restrict indentation geometries in characterization experiments to simple spherical geometries, such that the following Hertz contact relationship between the load force F and the indentation depth δ holds:

$$F = \frac{4}{3}\left(\frac{1-\nu_1^2}{E_1} + \frac{1-\nu_2^2}{E_2}\right)^{-1}\left(\frac{1}{R_1} + \frac{1}{R_2}\right)^{-1/2}\delta^{3/2} \tag{2.2}$$

where E_i, and R_i, and ν_i are the elastic modulus, radius, and Poisson's ratio of each equivalent elastic sphere. We note that these equivalent mechanical properties depend on the internal gaseous thermodynamic attributes as well as the induced strain in the membrane due

to expansion. In our setup, since pumping air into the capsule changes not only the internal pressure but also the volume of the capsule due to the expansion of the elastic membrane, we characterize each pre-contact internal gas state by the product of the pressure and the volume, denoted as PV.

At small indentation depths relative to the size of the sphere, we expect the Hertz relationship to be roughly linear, as was also demonstrated by [5]. The validity of this hypothesis will be verified in characterization experiments next.

Results

In order to bypass the need for solving complex elastic shell analysis problems in real time, we characterize the relationship between force and indentation depth of our device empirically. For contact obstacle spheres of different sizes (diameters of 10, 20, and 30 mm), the applied force vs. indentation depth is measured for different initial internal gas states. During data collection, the indentation depth is varied using the linear actuator at a quasistatic rate of change of 10 mm/s. At each time step, the indentation depth is measured directly by the depth camera, and the axial force by the load cell. These results are plotted in Fig. 2.9. As expected, the force-displacement curves are nearly linear. Moreover, for all three obstacles, the axial force F_z increases with PV at a fixed depth δ. Then, for each plotted curve given a fixed initial gas state characterized by PV, linear regression is performed and the fitted slopes are recorded. The relationship between this fitted slope vs. the initial gas state is plotted in Fig. 2.10. It is clear that as the diameter of the contact sphere increases, the slope of the force-indentation curve increases given a fixed initial PV. Though the slope increases monotonically with PV, the relationship is non-linear. We then fit a polynomial curve of order four to each of the plots in Fig. 2.10. As a result, for each of the three spherical geometries, a deterministic relationship between PV and its resulting force-indentation model is found.

Force-Matching Experiments

It is shown that the sensor's readings are impacted by its interaction with the environment. Similarly, its physical interactions can be informed by its sensory feedback. We demonstrate this notion of perception for action by utilizing the camera data to accurately apply a desired force on the environment. In this experiment, we leverage our experimental model built in Section 2.4 to map indentation depths to applied forces, with the size of the contact sphere as well as the initial PV state as parameters. Thus, given a desired axial force to be applied, the corresponding indentation depth that would result in this desired force can be estimated. Then, using the membrane point cloud as a control input (perception), the sensor can move to any desired indentation depth in order to apply a certain desired axial force on the contact obstacle (action).

We test the sensor's ability to perform this force matching on the 10 mm diameter contact sphere at three different pre-deformation gas states that are representative of the range of gas states characterized. Then, for each $(PV, F_{desired})$ pair, the sensor autonomously moves to a position such that the indentation depth is equal to its estimate of $\delta_{desired}$ based on the experimental model. Note that the set of desired forces is chosen to lie within the range of

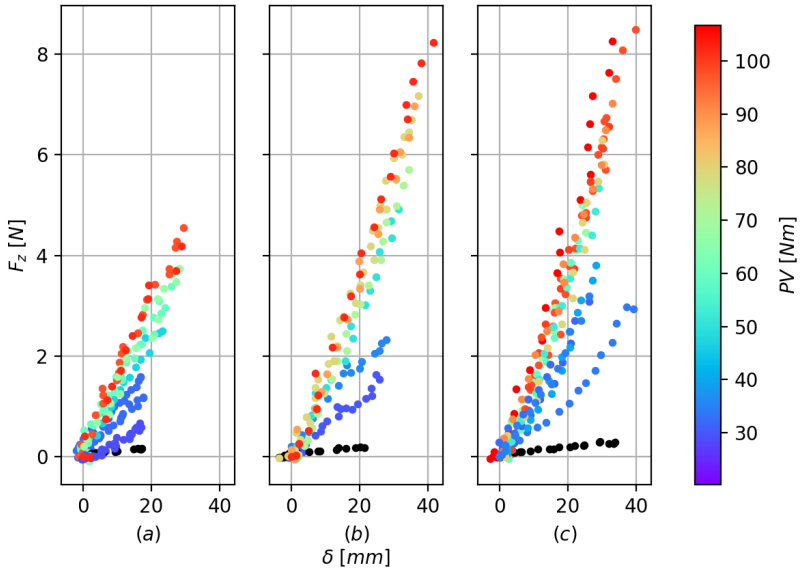


Figure 2.9: Force-indentation results upon contact with spheres of diameter a) 10, b) 20, and c) 30 mm at different pre-deformation PV states. Results when the capsule is allowed free air exchange with the atmosphere are plotted in black. ©2019 IEEE.

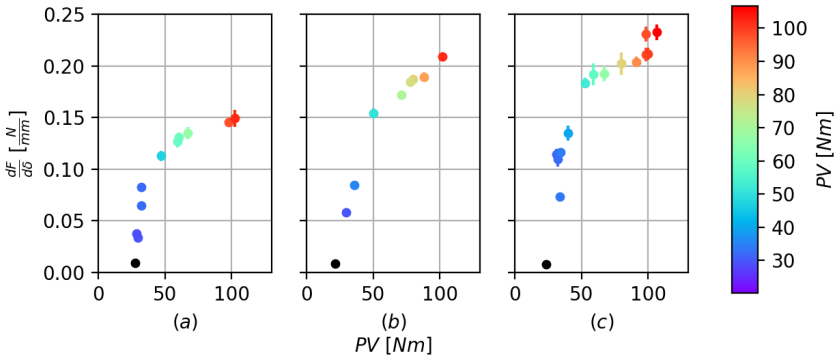


Figure 2.10: The slope of the force-indentation curve as calculated via linear regression plotted against the pre-deformation PV state. Results when the capsule is allowed free air exchange with the atmosphere are plotted in black. ©2019 IEEE.

possible applied forces at each of the three PV states. The sensor is actuated using a simple position-based proportional controller using the indentation depth as feedback. Three trials, each of which lasts less than 3 seconds, are performed for each $(PV, F_{desired})$ pair, and the results are shown in Fig. 2.11.

The average errors, defined as $F_{actual} - F_{desired}$, are tabulated for each PV state in Table 2.1. We remark that the error decreases for higher PV states. One possible explanation

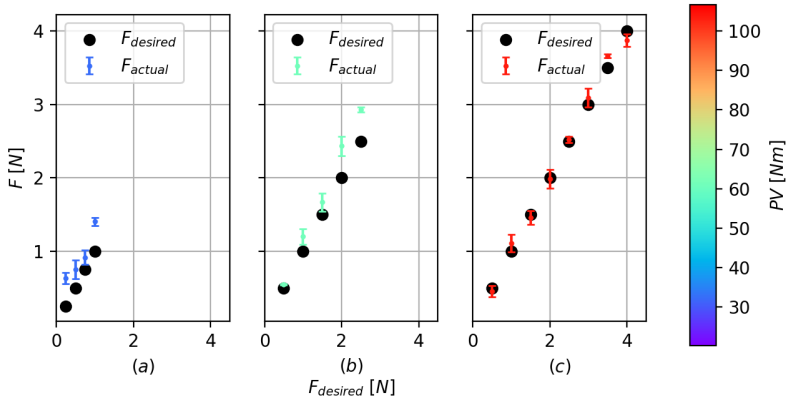


Figure 2.11: Force matching results for three different pre-deformation PV states a) 32.5, b) 61.2, and c) 103.4 Nm using the model for the 10 mm diameter contact sphere. The mean and standard error of the three trials are plotted along with $F_{desired}$. ©2019 IEEE.

Table 2.1: Average force error on a 10 mm diameter sphere.

PV **[Nm]**	32.5	61.2	103.4
Average Error [N]	0.30 ±0.08	0.25 ±0.08	0.02 ±0.07

for this observation is that perturbations about small PV values yield high variances in the resultant estimated slope of the force-indentation curve. This is evident in Fig. 2.10, where the rate of change of the curve is much higher at lower PV. For this reason, the model is extremely sensitive at low PV values, and the resulting force errors in practice are expected to be higher. All in all, the sensor's ability to autonomously apply desired external forces using only sensory information in real time supports the validity of the experimental model characterized in Section 2.4.

Chapter 3

Physical Human-Robot Interaction for Upper Limb Support

This chapter is adapted in part from "High Resolution Soft Tactile Interface for Physical Human-Robot Interaction" [58], written in collaboration with Ruzena Bajcsy.

3.1 Introduction

Physical interaction among humans is a natural and essential method by which we communicate with and assist each other. As a communication tool, physical interaction can be more clear and intuitive than providing auditory or visual cues. For example, one can swiftly grab a friend's arm to stop them from walking into incoming traffic, or manually adjust their form when teaching them a new task, such as how to swing a golf club properly. As a means of assistance for those who have limited mobility, physical interaction is even more vital. This is especially clear for caretakers who aid clinical patients or the elderly in everyday tasks such as helping them walk, sit up in bed, or change their clothes.

If robots are to coexist with and be useful to humans, endowing them with similar capabilities of physical interaction would be hugely advantageous and would facilitate a myriad of exciting applications with considerable societal impact. A necessary component of such a robot capable of physical interaction is a tactile interface, which should not only ensure a human's safety during interaction, but also enable the correct sensing of interaction forces. In this work, we propose a tactile interface that strives to address these two concerns of *safety* and *accurate sensing*.

Our tactile interface consists of a pneumatically pressurized domed contact membrane molded from silicone. The compliant properties of this soft device make it inherently safe for physical interaction with humans. Moreover, our design belongs to an up-and-coming class of optical tactile sensors that utilizes embedded depth-sensing cameras to extract interaction information from a soft contact region. This type of camera offers a richness in visual data that is unmatched by traditional 2D cameras and ultimately enables high sensing accuracy. Specifically, there are two main goals that are desired in tactile sensing. Firstly, it is useful to infer the physical geometry of the object in contact, and it has been shown in our previous

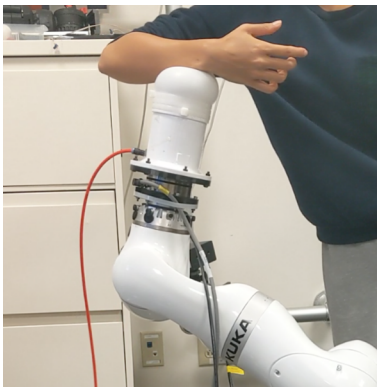


Figure 3.1: The proposed soft interface enables safe and precise tactile interaction between a human and a robot. ©2020 IEEE.

work that our design does so extremely efficiently [65]. Secondly, a more difficult goal is to infer the force that is being applied on the sensor. It is not possible to directly read force from vision, so some sort of mapping is required. This is challenging even for a sensor like ours that outputs a high-fidelity 3D membrane representation, as non-linear constituent elastic stress-strain equations require finite element methods that are computationally intensive and highly sensitive to manufacturing parameters. In this work, following the recent successes in the field, data-driven techniques will be employed in the modeling of the soft contact region.

The space of all contact interactions and resultant forces for which to build this mapping is extremely vast, but will be constrained in this work based on two proposed physical human-robot interaction (pHRI) applications that both involve interaction with the human upper limb. These applications are inspired by our motivating examples for physical interaction to be used in *communication* as well as in *assistance*. To demonstrate tactile communication, we show that a human can successfully alter a robot's end effector pose by indenting the tactile interface in the appropriate location with their fingertip. To exemplify an assistive task, we demonstrate that our soft interface can act as a soft forearm support that adapts to and follows the human's dynamic forearm motions by interpreting the shear and torque forces applied by the human forearm to the tactile interface.

3.2 Related Work

Methods for Inferring Interaction Forces Visually

It is challenging to interpret forces from visual data provided by optical tactile sensing methods. To this end, finite element modeling techniques have been applied to generate a mapping between visual data and contact forces, but this approach is computationally heavy and not suitable for real time control [156]. In the recent years, data-driven methods have proven themselves to be fast and accurate for use with modeling optical tactile sensors. Within the TacTip family, deep learning enabled the fast development of edge detection and contour following capabilities. For even more fine tasks, for example, the GelSight

has been a successful platform for learning physical models to detect object hardness [181], or to perform complicated manipulation tasks via reinforcement learning [47], or even to detect differences in fabric texture [90]. As a result of these successes, convolutional neural networks for processing visual data in tactile sensing has become a promising method in the community. In this work, we adopt a similar architecture when creating our own model.

Existing Applications in pHRI

The space of existing pHRI applications in general is limited with respect to purpose and precision. In rigid robots, force-torque sensors are the primary transducers for pHRI, enabling interactions for example with robot arms [6] and full-bodied partner ballroom dancing robots [78]. The sensory capabilities of these robots are very powerful, but their pHRI applications are limited due to risk of unsafe interactions with the hard chassis. The scope of existing soft pHRI robots is unsurprisingly even more narrow due to challenging constraints in mechanical design and sensing. Some noteworthy examples of existing soft pHRI systems include RI-MAN, a humanoid robot designed to help lift and carry humans [122], and Huggable, a soft robot pet surrogate [158] that responds to affective touch-based interactions. There is an apparent need in the field to develop robots that can perform useful and precise pHRI task while remaining compliant enough to assure human safety. Thus, this work presents two useful applications of the proposed robotic system that involve completely soft interactions and do not require bulky transducers apart from a small internally embedded camera.

3.3 Data Collection

Admissible Interactions

The space of admissible interactions studied in this work was limited by design based on the two different pHRI applications we wished to achieve. Both applications involve specific interactions with the human upper limb, and illustrate the usage of physical interaction as a means of human-robot *communication* and *assistance* respectively.

Robot End Effector Pose Correction

The first application utilizes the tactile interface as a communication tool from the human to the robot. In this regime, the human physically corrects the end effector pose of the robot by prodding the robot towards the desired configuration through interaction with the tactile interface. Here, interactions for this application are limited to single finger poking so that the contact geometry can be kept simple. Thus, we avoid more complicated interactions such as those from multiple fingers and/or the palm.

Dynamic Support of the Human Forearm.

The second application enables the robot to act as an assistive device for the human. Specifically, it assists with upper limb motion, a common feature in ergonomic design and exoskeletons that helps minimize the effort required to support the weight of one's arm [138]. In this

application, the human's forearm rests on top of and is fully supported by the soft tactile interface. However, such a support system is only useful if the human has autonomy over their own movements. Thus, the device follows and dynamically supports the forearm as it translates in space, as well as rotates with it when a torque about the sensor axis is applied.

Datasets

Both applications can only be implemented if the robot is aware of the forces and torques being applied to it during interaction. In its most useful form, the robot should only have access to the depth camera readings, since being outfitted with a force-torque sensor renders the overall system less compliant. Thus, mappings that predict the force-torque responsible for a measured membrane point cloud need to be available to the robot. To this end, we systematically collect a dataset of admissible interactions and their resulting induced force, torque, and point clouds in order to train a data-driven model that provides this mapping. There are two datasets, one for each application, that are collected. Note that the coordinate system is defined to be fixed to the ATI force-torque sensor. The principal camera axes also share the same orientation as the force-torque coordinate frame.

Finger Poking Dataset

Fingertip apparatus. A rigid sphere 1 cm in diameter printed from PLA is used in the data collecting experiments as an approximate duplicate of an average human fingertip. Since minimal slip and shear is expected for the poking interactions, it is reasonable to neglect the frictional effects of skin versus PLA.

Pre-indentation contact points. The sphere is then used to indent the membrane at a variety of locations at different angles and depths, and the resultant membrane point cloud and force-torque measurements at each state are recorded. The interaction states are discretized systematically. First, a set of initial contact points on the membrane surface are selected. As illustrated in Fig. 3.2, this is achieved by designating 40 equally spaced quarter-circle arcs that run from the pole to the base of the hemisphere (parameterized by $\phi = \frac{\pi j}{20}, j \in [0..39]$). Along each quarter-circle arc, five points are selected with projections on the xy-plane that are equally spaced with radii 0 to 4 cm from the pole, leaving enough of a distance from the base of the hemisphere so that boundary contact conditions can be avoided. These initial contact points indicate the location at which the spherical obstacle first makes contact with the membrane to begin indentation.

Indentation locations Then, the sphere indents the membrane at varying depths and angles from the z-axis. As illustrated in Fig. 3.3, each pre-indentation contact point is the starting point for 10 different indentation directions ($\theta = \frac{\pi k}{18}, k \in [0..9]$). Along each indentation direction, five different indentation depths δ equally spaced from 0.5 to 2.5 cm are visited. All indentation locations that stemmed from any one pre-indentation contact point are constrained to one plane, and interactions are actuated by the KUKA arm pushing into the fixed sphere at the appropriate angles and depths. In total, there are 10,000 unique states for which data is collected.

Visualization of example data. Figs. 3.4a, 3.4d, and 3.4g are radial plots of ground truth τ_x at a particular indentation state parameterized by θ and δ starting from every pre-

indentation contact state, where the size of each marker is scaled according to the magnitude of τ_x. Between Figs. 3.4a and 3.4d, the magnitude of the measured τ_x is greater in 3.4d, due to its indentation depth being 1 cm deeper. Fig. 3.4g depicts the torque value when the θ is large enough that torque applied is in the opposite direction compared to a smaller θ such as in Fig. 3.4d. The magnitude is also larger because the moment arm to the x-axis attached to the force-torque sensor becomes longer.

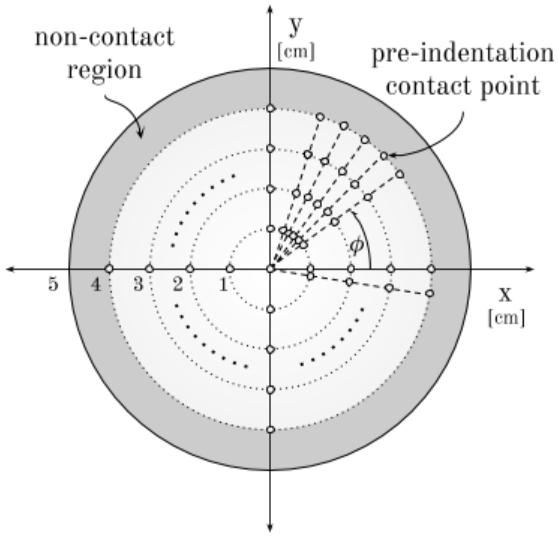


Figure 3.2: Polar view of the pre-indentation fingertip contact points that lie on 40 evenly-spaced quarter-circle arcs. ©2020 IEEE.

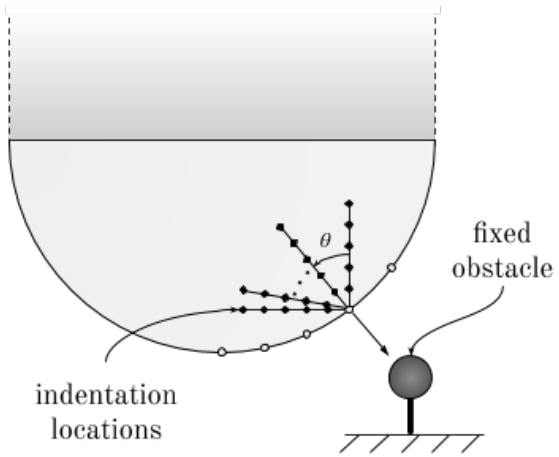


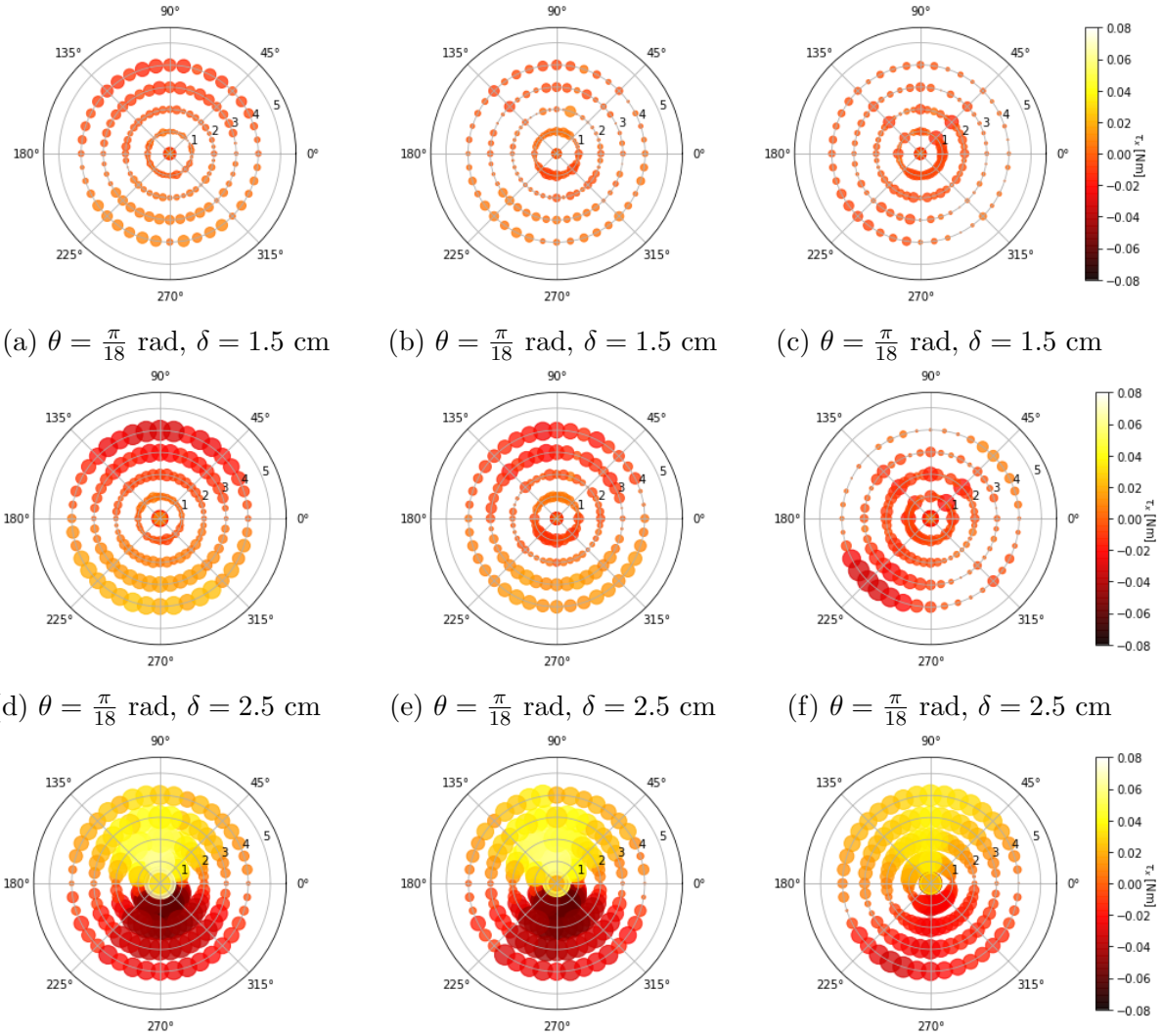
Figure 3.3: Side view of the pre-indentation contact points and the indentation locations associated with each. ©2020 IEEE.

Forearm Shearing and Torquing Dataset

Forearm apparatus. An attempt to replicate a generic forearm geometry using a simple 3D-printed ellipsoid is unsuccessful due to the different material properties between PLA and human skin yielding different frictional forces during interaction. Thus, data is collected using the researcher's real forearm. The forearm is placed directly underneath the sensor during data collection such that its axis is centered on the membrane. A fixed brace is used to keep the forearm consistently in the same place between data collection sessions.

Shearing. The device pushes into the forearm until an indentation depth δ is achieved, as measured from the z-displacement of the pole. Fig. 3.5 illustrates the indentation locations visited by the forearm contact point. Then, the device moves in a specific direction parameterized by ϕ in the xy-plane, defined in the same way as in Fig. 3.2 of the fingertip experiment, and stops at 5 different distances from 0.3 to 1.5 cm away. This shearing distance was limited to avoid material slip. This process is then repeated for $\phi = \frac{2\pi l}{40}, l \in [0..40]$ rad and also for $\delta = \frac{m}{4}, m \in [0..8]$ cm.

Torquing. In the same manner as the shearing experiments, the device pushes into the forearm until an indentation depth δ is achieved. Then, the device rotates about its z-axis



(a) $\theta = \frac{\pi}{18}$ rad, $\delta = 1.5$ cm (b) $\theta = \frac{\pi}{18}$ rad, $\delta = 1.5$ cm (c) $\theta = \frac{\pi}{18}$ rad, $\delta = 1.5$ cm

(d) $\theta = \frac{\pi}{18}$ rad, $\delta = 2.5$ cm (e) $\theta = \frac{\pi}{18}$ rad, $\delta = 2.5$ cm (f) $\theta = \frac{\pi}{18}$ rad, $\delta = 2.5$ cm

(g) $\theta = \frac{7\pi}{18}$ rad, $\delta = 2.5$ cm (h) $\theta = \frac{7\pi}{18}$ rad, $\delta = 2.5$ cm (i) $\theta = \frac{7\pi}{18}$ rad, $\delta = 2.5$ cm

Figure 3.4: Polar plot of τ_x in the fingertip dataset starting at all pre-indentation contact points for selected values of θ and δ. Columns separate ground truth readings (left), prediction learned from poking data (middle), and prediction learned from both poking and forearm data (right).

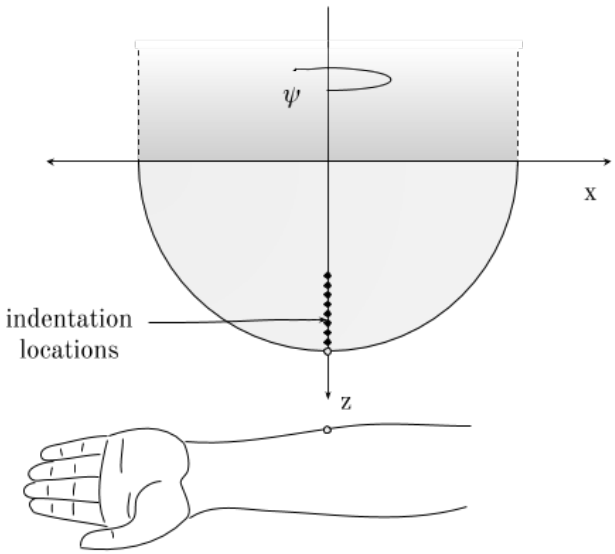


Figure 3.5: Side view of the forearm indentation states. Arm is not drawn to scale. ©2020 IEEE.

in one direction, stopping at angles $\psi = \frac{\pi n}{24}, n \in [0..8]$. for data recording. The maximum rotation angle achieved, $\frac{\pi}{3}$, is limited to avoid material slip from excessive torquing. This is repeated for rotation in the opposite direction, and then also for $\delta = \frac{p}{4}, p \in [0..8]$ cm. For both the shearing and torquing motions, data is collected for 1726 interaction states. In order to achieve a better distribution of samples, the forearm experiments are repeated 4 more times so that a total of 8640 datapoints are collected.

3.4 Wrench Prediction Model

Architecture

The neural network architecture used to map the input of a 171×224 organized point cloud into a 6-dimensional force-torque vector is illustrated in Fig. 3.6. Each point cloud input, an array of depth measurements, is passed through a series of two convolutional and max-pooling layers, and then through a fully-connected regression network to yield the predicted resultant force-torque measurement. We train three different models with the same architecture. The first model combines the data from the two datasets, and the other two are trained on either only the poking or only the forearm dataset. Since the two interaction regimes are quite different, we want to explore whether the combined dataset network would be able to generalize and perform as well as each of the individial dataset models. The networks are trained with the SGD optimizer and a learning rate of 0.01 with no decay, and a dropout of 0.2 is implemented in the fully-connected layers to prevent overfitting. An 80:20 train/test split is selected for hyperparameter tuning and validation. The finger poking dataset, with 10,000 total datapoints, leveled out to a validation MSE of 0.03 after training with a batch size of 500. The forearm dataset, with a total of 8640 datapoints, yielded a final validation

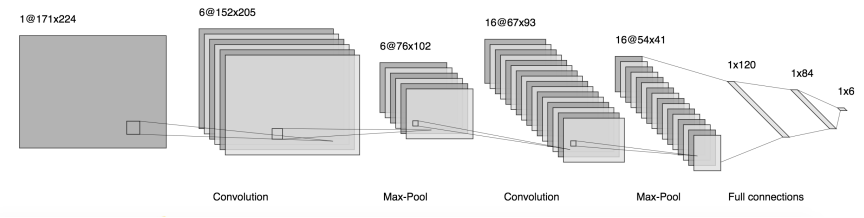


Figure 3.6: Architecture of the force-torque-deformation model. ©2020 IEEE.

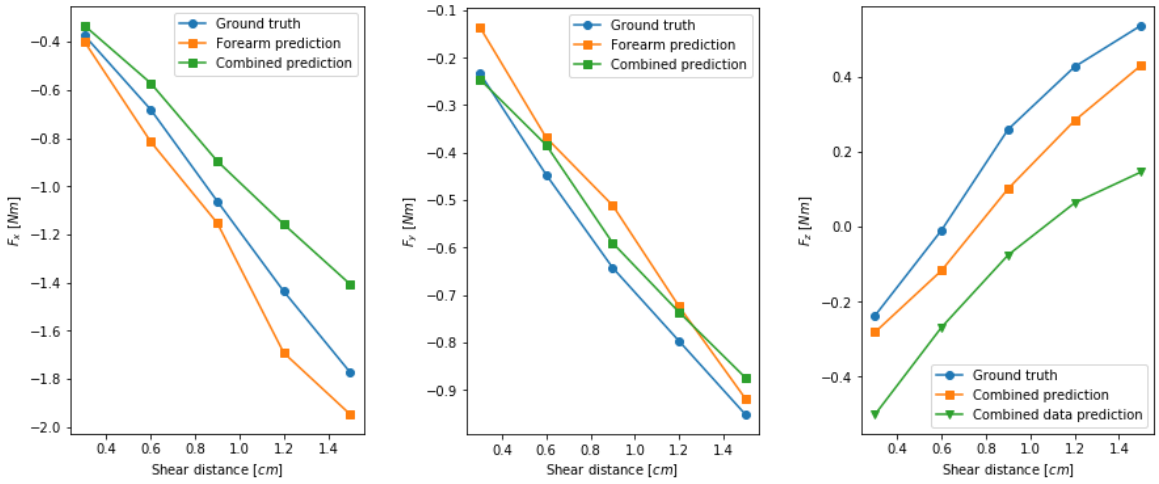
MSE of 0.057 after training with a batch size of 400. The combined dataset yielded a final validation MSE of 0.1 with a training batch size of 500. All force-torque values are recorded in SI units.

Prediction Results

We find that in general, the two models trained on the individual datasets perform very well in validation. For example, in Fig. 3.4 we can compare the performance of the poking dataset model (middle columm) with the ground truth (left column) and note that the prediction is qualitatively accurate. However, it is apparent that the combined model (right column) yields less accurate predictions, which can be vastly different from the ground truth. For example, in Fig. 3.4f, the sign on the τ_x reading in the third quadrant is evidently wrong. The performance of the forearm model is also very good in predicting forearm shear and torquing forces. Fig. 3.7 visualizes example validation data for both the forearm and the combined models on a set of ground truth forearm shearing readings at $\phi = \frac{3\pi}{4}$ at a depth of $\delta = 0.75$ cm. Qualitatively, both models perform well. However, when predicting forearm torques, with example prediction on validation data in Fig. 3.8, the combined model fails to output good prediction. One explanation is that while shearing and poking interactions involve pushing, the twisting involved in torquing yields more complicated membrane structures, and there is not enough training data for the combined model to generalize well.

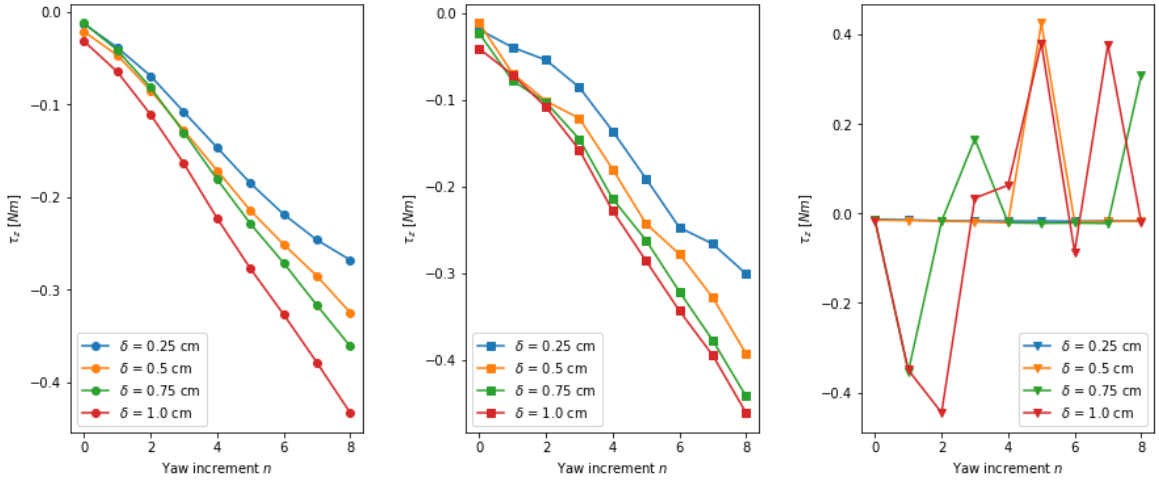
3.5 Real-World Applications

We perform a proof-of-concept demonstration of the two applications in which the robot uses its learned force-deformation models. For each application, we utilize the individually trained models rather than the combined one due to their higher accuracy. Example snapshots are depicted in Figs. 3.9 and 3.10. For the two cases, we implement the same simple control loop at a slow 1 Hz rate for safety purposes, at which the robot changes its configuration based on its the predicted force and torque from the point cloud stream. At each update step, where values are measured in SI units, the robot position in the world frame $p = [p_x \;\; p_y \;\; p_z]^T$ is updated according to $p_i \leftarrow p_i + 0.1F_i, i \in \{x, y, z\}$. Additionally, a rotation about each of the principal axes in the body frame attached to the base of the cylinder (i.e. at the force sensor), is applied according to $R_i(2\tau_i)$ where R_i i,s the rotation operator about the i^{th} principal axis. These constants are chosen empirically such that they yield a large enough



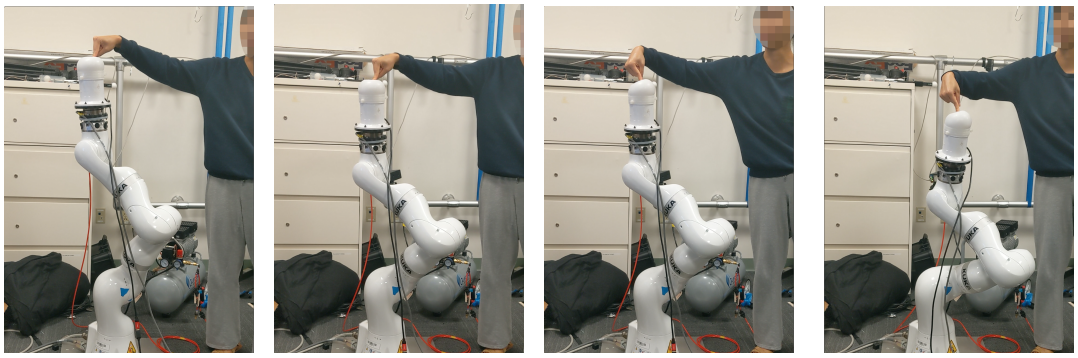
(a) Predicted F_x (b) Predicted F_y (c) Predicted F_z

Figure 3.7: Forearm shearing example validation data at $\phi = 3\pi 4$ and $\delta = 0.75$ cm comparing the predictions of the forearm and combined models with the ground truth. ©2020 IEEE.



(a) Ground truth data. (b) Forearm model prediction. (c) Combined model prediction.

Figure 3.8: Predicted τ_z on example forearm torquing validation data as a function of the yaw angle for different indentation depths δ. ©2020 IEEE.



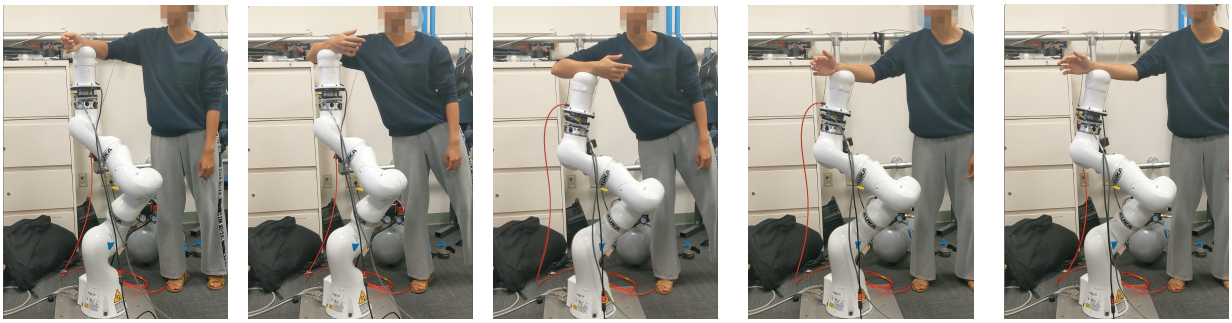
(a) User uses their finger to poke along and opposite to the z-axis of the robot.

(b) Robot changes position proportional to its sensed contact force.

(c) User pokes at the interface at an angle to the z-axis.

(d) Robot changes position and orientation according to the user's force-torque input.

Figure 3.9: Real-world validation of the robot end effector pose correction application. ©2020 IEEE.



(a) User begins by resting their forearm on the soft interface.

(b) User applies a combined shear and torsional force to the robot.

(c) Robot reacts by changing both its position and orientation.

(d) User attempts to exert a pure torque on the interface.

(e) Robot reacts by twisting in the same direction as the applied torque.

Figure 3.10: Real world validation of the dynamic forearm support. ©2020 IEEE.

change in robot joint configuration at each step without being too sensitive to sensor noise. In the forearm application, the force is initially zeroed at the resting position (i.e., when the forearm's weight is fully supported by the robot). As expected from the good validation results in the models, the system is reliable and consistently responded as expected. However, an in-depth user study is not performed, so the usability and comfort of the system is not confirmed here.

3.6 Discussion

We present a method to infer contact forces and torques during interaction with our soft tactile interface. Rather than implement complicated, intractable analytical solutions, we turn to numerical solutions. We train a neural network to resolve contact wrenches for a limited set of upper limb interactions, and show that inference can be performed in-the-loop. We finally leverage these predictions on several pilot experiments with a real human.

However, our results are limited compared to a general physics solver, either analytically or using 3D FEM. Our network only predicts net wrench signals, rather than entire force distributions over the interface body. While this may be enough for many applications, it could be useful to expand inference to the entire contact distribution itself, which may enable more complicated interactions such as multi fingered pinching. As evidenced from the combined model's performing unreliably compared to the individually trained models, we are far from having achieved generalization with our technique. In the future, using networks that leverage known physical relationships in elastic shell theory, such as those with strong relational inductive biases such as graph networks, would likely outperform the CNN-based technique in our work. Finally, running the system at a higher control rate would be more compelling for real world applications, and it would be way to test and develop real-time HRI algorithms for human-robot teaming and teaching.

Chapter 4

Contour Following

This chapter is adapted in part from "Soft Tactile Contour Following for Robot-Assisted Wiping and Bathing" [60], written in collaboration with Dylan Chow and Ruzena Bajcsy.

4.1 Introduction

Robots have the potential to greatly reduce the time and effort humans spend daily on domestic and personal care. One such major task is cleaning the home, for which the average American spends almost 24 hours per month. Over the past couple of decades, floor cleaning robots have become a popular consumer product [160], in which the home environment is simplified into essentially a two-dimensional problem due to the planar nature of the floor. However, robotic cleaning in the three-dimensional domain is more complicated, and has yet to see mainstream adoption in the consumer market. For example, such tasks include the cleaning of furniture like windows, kitchen counters, railings, and sinks. In a similar vein, the bathing of humans is also a three-dimensional problem. For bathing, the development of assistive robots is not only a matter of convenience, but also of necessity. In fact, the inability to bathe independently is the leading cause for moving into full-time care facilities [40]. Meanwhile, the current shortage of nurses [48] continues to be stressed by the rapidly growing elderly population [117].

We group the cleaning of household furniture and bathing of bodies as one unified problem. Both involve the process of *contour following*, in which a robot performs wiping motions along the edges and surfaces of what is being cleaned. In addition, a certain level of contact between the robot and the object or body is maintained in order to apply the required forces to scrub off grime or dirt. Based on how humans themselves would perform these cleaning tasks, we follow three major design principles when building our system.

First, tactile sensing is the dominant and most informative sense in humans for performing contact-rich tasks such as wiping for cleaning [84]. Therefore, our system also utilizes tactile sensing, though external vision-based sensing can add complementary scene comprehension in future work. Similarly to a human hand, our tactile sensing relays signals that are informative of the target's geometry as well as the contact forces being exerted. Second, humans are able to safely perform wiping for cleaning without damaging the target, due to the

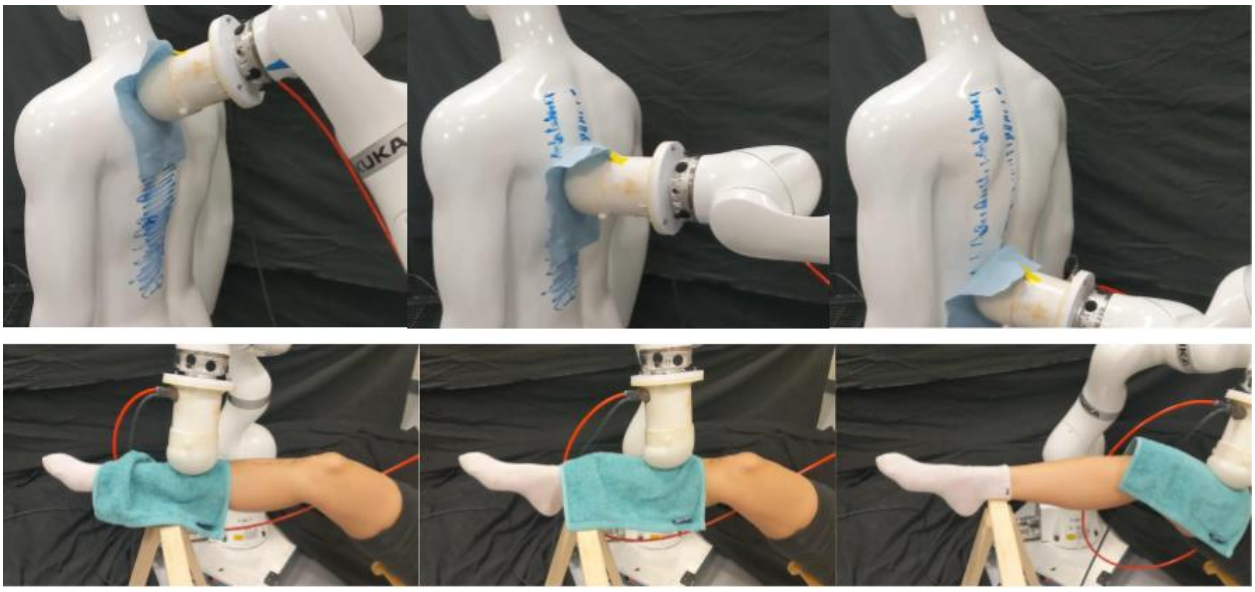


Figure 4.1: We use a soft tactile sensor to perform cleaning tasks on objects and body parts via contour following and force regulation. This general controller is successful on all unseen tasks and requires no learned models. ©2022 IEEE.

softness of their palms as well as the compliance in their arm joints. Introducing compliance into the robotic system is crucial for safety, so we propose the use of an soft tactile interface that is inherently soft due to material compliance. Third, humans can easily adapt to wiping unseen geometries. Thus, we also design our system to function for *general* contours, rather than impose strict constraints on object geometry like most previous works have done. We also employ a physically-interpretable control scheme that requires no data-driven learning. Finally, we demonstrate our system on wiping tasks for a variety of objects such as pipes, shelves, and parts of the human body. To our knowledge, this is the widest set of general contour-following experiments for objects on a human-sized scale.

Existing work in the field utilizes diverse techniques and designs to perform robotic wiping. Saito et al. applied deep learning and kinesthetic demonstration to teach a robot holding a rigidly-connected mop to wipe the exterior surface of lampshades [147], and found that the tactile signal corresponding to four pressure sensors attached to the mop handle were more important features for successful wiping than were the raw camera images of the scene. Gao et al. represented objects using task-oriented keypoints for force control tasks such as wiping a whiteboard with soft erasers [41]. However, the whiteboard surface was known to be flat with fixed and predetermined edges. In another work, when the geometry of the object of interest was perfectly known in an industrial setting, wiping occurred by performing force control while tracking a predefined position trajectory [124].

Bathing also involves a very similar method of wiping that is instead performed on the surface of a body. King et al. used a robot end effector wrapped in a washcloth to perform back and force wiping motions on a user-specified region of a human arm. The wiping trajectory was purely horizontal as enforced by a constraint that the arm was laid flat on a bed, and was initiated upon contact detected by a force-torque sensor at the wrist [76]. To

adapt to a wider set of limb poses, Erickson et al. used a six-electrode capacitive sensor at the end effector to systematically probe the surrounding areas of human arms and legs bent at different angles, and trained a neural network to predict the relative pose between the end effector and the limb at test time. With real-time pose estimation, the sensor was able to successfully traverse the length of human legs and arms, even when bent at the knees or elbows. However, only position targets were used, and applied forces during wiping were not controlled. A vision-based system washing system used cameras to dynamically detect the surface of a human back and fit a back-and-forth wiping dynamic motion primitive to the relevant region [28], but for safety did not actually perform the action on a human due to a lack of tactile sensing. While vision and tactile sensing are complementary, in this work we develop a tactile-only system due to its dominance in contact-rich tasks such as contour following [84].

A major shortcoming in almost all of these works is that the controllers were designed only for one or two specific objects. However, we believe a useful controller for executing wiping actions on any surface, be it on objects or on humans, should be *general*. For example, a robot that is able to wipe down a railing should also be able to wipe down a person's back. Thus, we cannot make assumptions about object geometry (e.g., that it is flat [41]) a priori. However, inference of local object geometry information is difficult to achieve from low-dimensional implicit sensor signals, which requires the use of neural networks and adequate training data. Yet, these models are non-interpretable and functional only on seen objects (e.g. lampshades [147] or limbs [34]). In this work, we propose the use of an optical tactile sensor that provides explicit geometric signals of contact surface geometry via imaging the deformation of a soft membrane. As a result, we can apply the same general controller to wipe any unseen object surface without the use of deep learning.

4.2 Contour-Following Controller

Intuitively, successful contour following needs to fulfill several specifications. First, the shape of the object should determine how the robot moves in space so that contact is maintained. Second, the amount of force applied on the object should be regulated. Third, contact should be constrained to the soft part of the robot only. To achieve these specifications, we propose the following three-term discrete-time linear controller to govern the end effector pose with translational and rotation components $\vec{p}$ and R. Please refer to Fig. 4.2 to visualize the controller's state variables.

$$\vec{p}(t+1) = \underbrace{K_v \vec{v}_o(t)}_{\text{tracking term}} - \underbrace{K_f(F_{des} - F(t))\vec{n}_o}_{\text{force term}}(t) + p(t) \tag{4.1}$$

$$R(t+1) = \underbrace{R(\vec{\omega}(t), K_\theta \theta)}_{\text{axis alignment term}} R(t) \tag{4.2}$$

We now explain the effect of each term, while the next subsection details how these state quantities (e.g. $\vec{v}_o$, $\vec{\omega}$) are computed from sensor point clouds. The tracking and force terms govern position change of the end effector, while the axis alignment term governs orientation change.

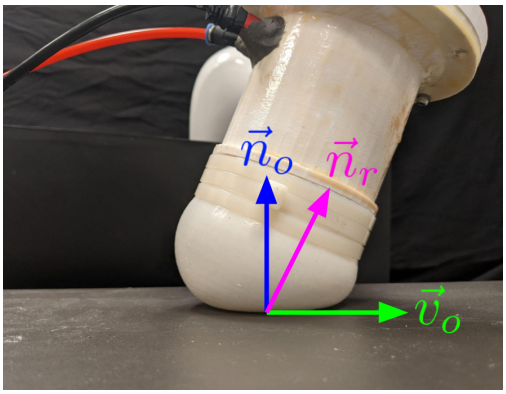


Figure 4.2: Visualization of the object tangent $\vec{v}_o$, object normal $\vec{n}_o$, and the sensor central axis vector $\vec{n}_r$. ©2022 IEEE.

Tracking term. The tangent vector to the object $\vec{v}_o$ determines the local change in geometry of the object. By moving the end effector in exactly this direction, we are able to traverse along the target's contour. $K_v > 0$ acts as the step size and should be small to ensure that the sensor will not lose contact with the object between each timestep.
Force term. This term controls for the total applied force exerted onto the object. For example, wiping tasks often require high enough forces to scrub away grime or dirt, but not high enough to cause abrasion on human skin. F_{des} and $F(t)$ are the desired and current force measures on the object respectively. This force term of the controller drives the end effector toward exerting a desired F_{des} on the object by moving along $\vec{n}_o$, which is perpendicular to the object tangent $\vec{v}_o$. The force value $F(t)$ can be measured with a force sensor, e.g. with a force-torque sensor placed at the based of the tactile sensing module [59]. However, in the absence of this additional force sensor, we extract a force-correlated measure $\tilde{F}(t)$ purely from the point cloud data that can replace the true force $F(t)$ in the control term.
Axis alignment term. Finally, the axis alignment term ensures that contact stays on the soft membrane and does not move into the rigid cylinder. Contact with the rigid part of the sensor is not only unsafe, but is also unable to be registered by the camera. Thus, we ensure that the central axis vector of the sensor, $\vec{n}_r$ remains perpendicular to the surface tangent $\vec{v}_o$. The change in orientation between each step is quantified by a rotation of θ about a vector $\vec{\omega}$, and a scaling factor $K_\theta \in (0, 1]$ limits the amount of correctional rotation applied. Fig. 4.3 shows the difference in movement when the controller omits the axis alignment term. In the absence of the axis alignment term, the object eventually makes contact with the rigid part of the sensor, and the cleaning cloth subsequently gets caught.

Computation of state quantities

Object tangent $\vec{v}_o$. The computation of $\vec{v}_o$ depends on whether the sensor is in contact with an object edge or with an object surface. When an edge is contacted (Fig. 4.5a), we want the robot to follow the edge itself (e.g., along the curved pipe with a distinct edge in Fig. 4.3). Thus, $\vec{v}_o$ is the direction of the contacted edge, which we calculate as the normalized principal component of the deformed point cloud. However, when the sensor

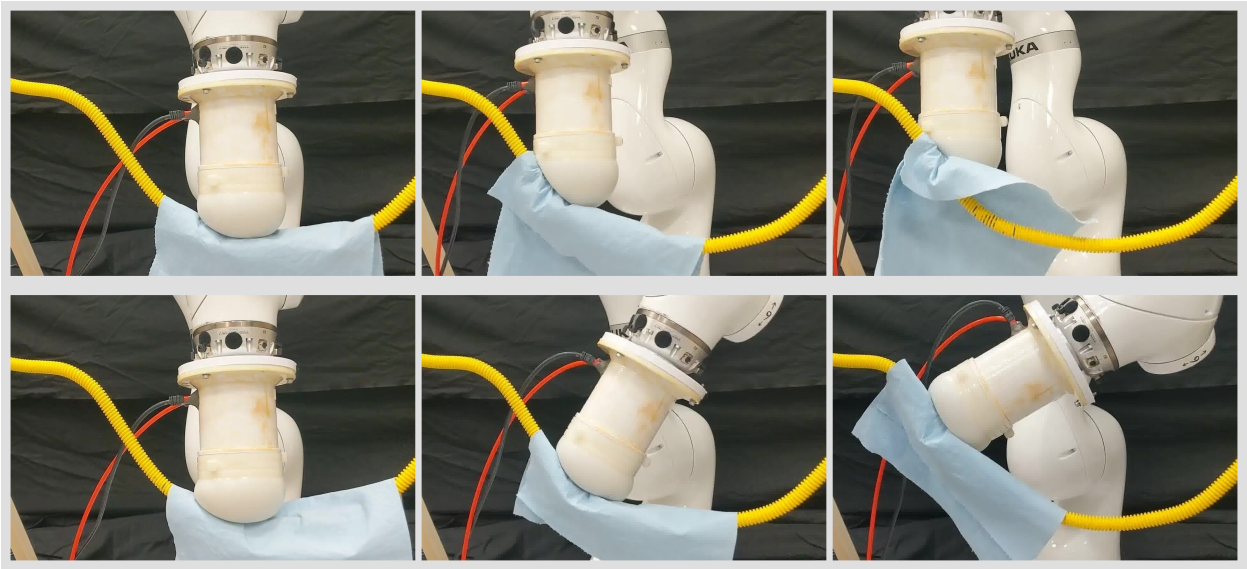


Figure 4.3: Following the contour of a curved pipe when the axis alignment controller term is omitted (top), compared to when all three terms are present (bottom). The absence of the axis alignment term can lead to undesirable contact between the rigid part of the sensor and the object itself.

contacts a surface rather than an edge (Fig. 4.5), there is no longer a dominant direction of variance. In this case, we assume that the position trajectory of the end effector should be constrained along a user-defined plane with normal $\vec{n}_p$. We define a plane that passes through the centroid of the deformed point cloud with normal $\vec{n}_p$, and select all points on the deformed cloud that are within 0.1 cm away from the plane. We then calculate $\vec{v}_o$ as the normalized first principal component of these selected points. Note that both $\vec{v}_o$ and $-\vec{v}_o$ are valid object tangent vectors, so the sign should be selected based on the convention of the practitioner.

Object normal $\vec{n}_o$. The object normal vector $\vec{n}_o$ is the normalized projection of the unit sensor axis vector $\vec{n}_r$ onto a plane with normal $\vec{v}_o$. Thus, $\vec{n}_o$ is perpendicular to $\vec{v}_o$.

$$\vec{n}_o = \frac{\vec{n}_r - \vec{n}_r \cdot \vec{v}_o}{||\vec{n}_r - \vec{n}_r \cdot \vec{v}_o||} \tag{4.3}$$

Rotation parameters $\vec{\omega}, \theta$. Finally, $\vec{\omega}$ and θ are the axis and angle of rotation that transform $\vec{n}_r$ into $\vec{n}_o$.

$$\vec{\omega} = \vec{n}_r \times \vec{n}_o \tag{4.4}$$

$$\theta = cos^{-1}(\vec{n}_r \cdot \vec{n}_o) \tag{4.5}$$

Force-correlated $\tilde{F}$. This quantity is extracted solely from point cloud data, and is correlated with the true applied force onto the sensor. Note that in order to compute the real applied force on the membrane from the deformation field of the point cloud, the most accurate method is via the finite element method, which has not been implemented for this system. We thus estimate the applied force by modeling each point in the cloud p as an

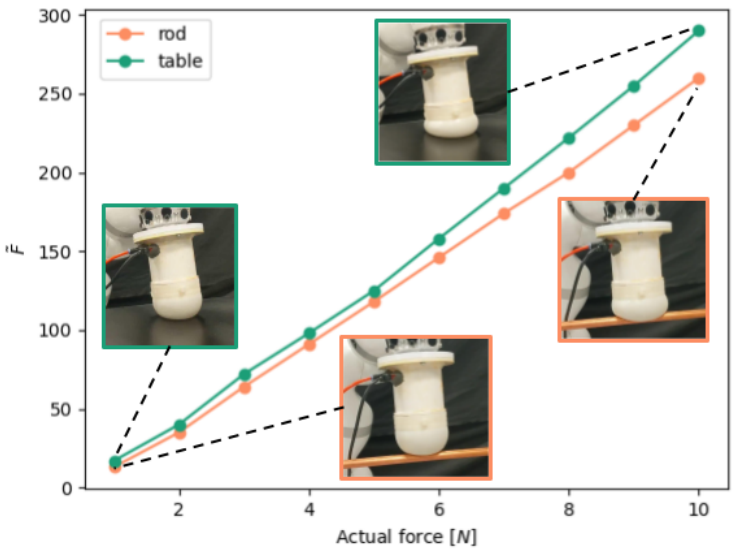


Figure 4.4: $\tilde{F}$ compared to the actual applied force as measured by an external force sensor. The correlation is extremely strong even when in contact with different objects. ©2022 IEEE.

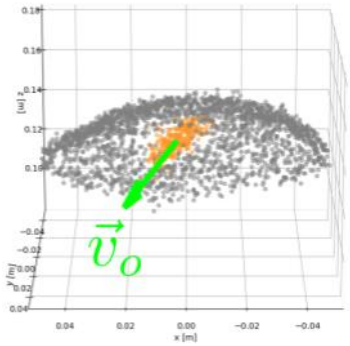
independent linear elastic element, such that the total $\tilde{F}$ is the sum of the total deformation over the N_p points in the cloud,

$$\tilde{F}(t) = \sum_{i=1}^{N_p} p_i(0) - p_i(t) \tag{4.6}$$

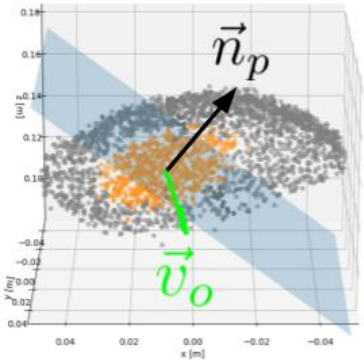
where $p_i(t)$ is the depth of point i at time t, and $p_i(0)$ is the depth of point i when the membrane is undeformed. Fig. 4.4 shows that there is an extremely strong correlation between $\tilde{F}$ and the actual force.

4.3 Applications

We demonstrate the generalizability of our system by using the same controller to perform a set of various contour-following tasks on different objects and body parts. We publish position commands at 1 Hz with $K_v = 1$ cm such that the robot's end effector translational velocity is approximately 1 cm/s. This speed can be increased in future work, where the objects and body parts may be more dynamic and thus require faster control rates. We classify each task as either an edge or surface following task based on prior knowledge of the object geometry. In all tasks, before initial contact is made, the robot is commanded to move in a fixed direction based on prior knowledge of the task (e.g., the robot will initially move downwards when it starts above a table). Once initial contact is made, it should be maintained for the rest of the trajectory. Various cloths are placed in front of the tactile interface to clean the object, and the cleaning target is stationary in all tasks.



(a) An object edge is contacted.



(b) An object surface is contacted.

Figure 4.5

Figure 4.6: The object tangent $\vec{v}_o$ is calculated either differently for edge contacts (left) and surface contacts (right) using the geometry of the deformed point cloud shown in orange. ©2022 IEEE.

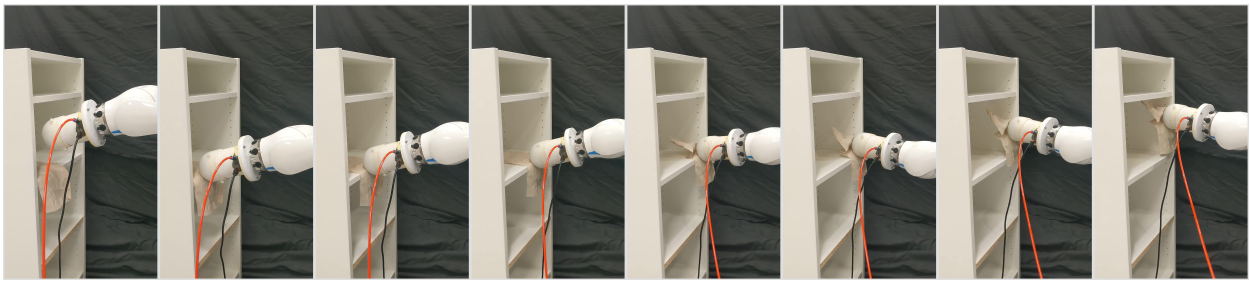


Figure 4.7: The robot wipes the inside edges of a bookcase shelf, while successfully detecting the change in edge direction. ©2022 IEEE.

Edge following

We demonstrate the robot's ability to follow and wipe an object when there is a prominent edge in the contact geometry. In Fig. 4.7, the robot is able to clean the inner edges of a shelf on a bookcase with a thin napkin. It lowers onto the shelf then senses contact and moves horizontally along the shelf edge. Once it reaches the corner of the shelf, it senses a new vertical edge of the bookcase and follows it upward. Next, we use a damp microfiber cloth to clean a trail of black pepper off a human arm and leg (figs/CF. 4.8, 4.9). Though human arms and legs do not feature geometric edges as sharp as those on the bookcase, there is still a principal direction of geometric variance along the length of the limb that the controller utilizes successfully for contour following. The robot is able to maintain contact throughout the dip in the elbow bend of the arm as well as over the bump of the knee on the leg. In both cases, the robot is able to clean off all of the pepper (Fig. 4.10).

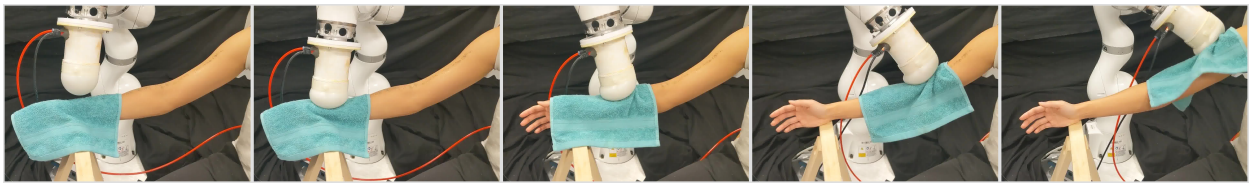


Figure 4.8: The robot wipes along the length of an arm with a damp washcloth, cleaning away a trail of black pepper. ©2022 IEEE.

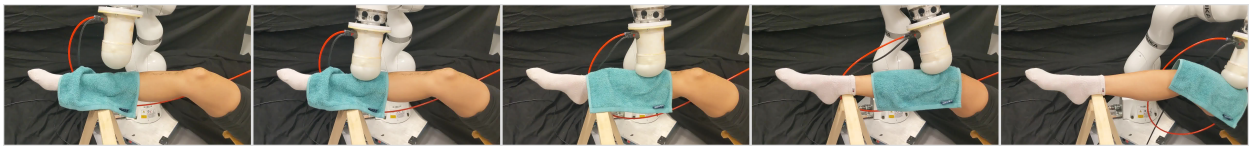


Figure 4.9: The robot cleans the the length of a leg with a damp washcloth and maintains constant contact even over the knee. ©2022 IEEE.

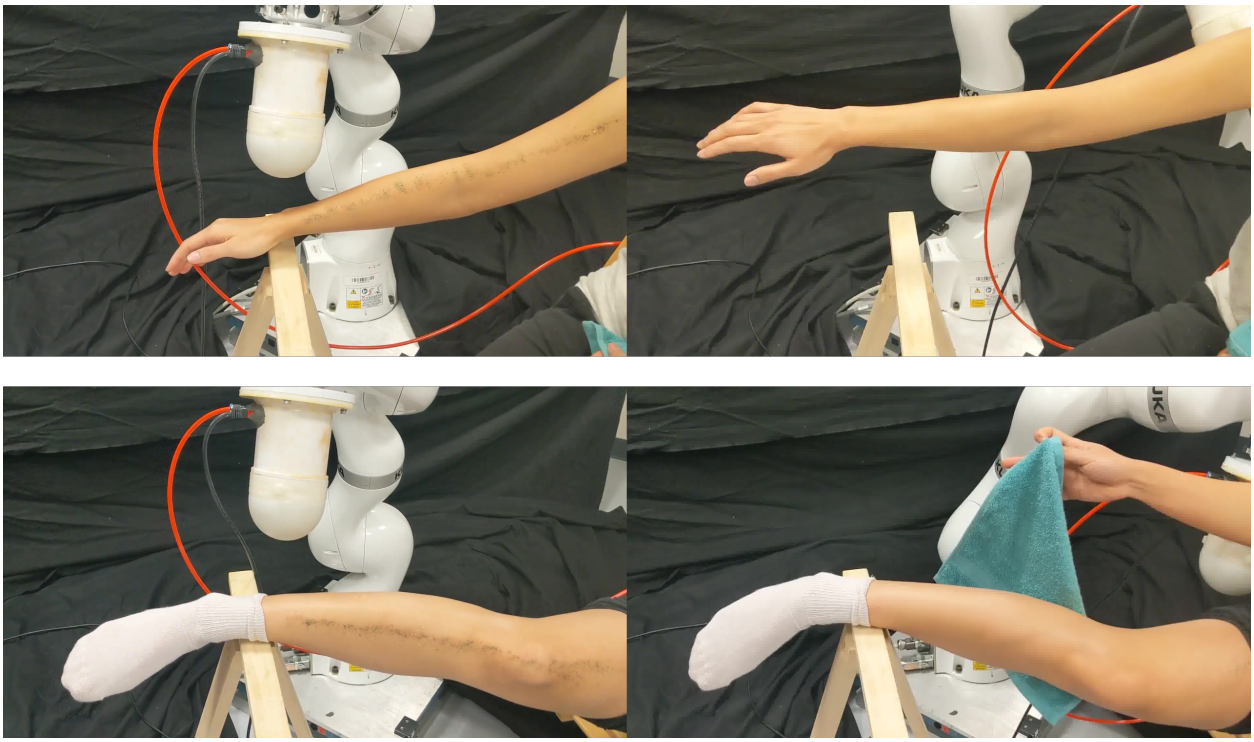


Figure 4.10: Before and after the wiping of a human arm (top) and leg (bottom) covered in black pepper. ©2022 IEEE.

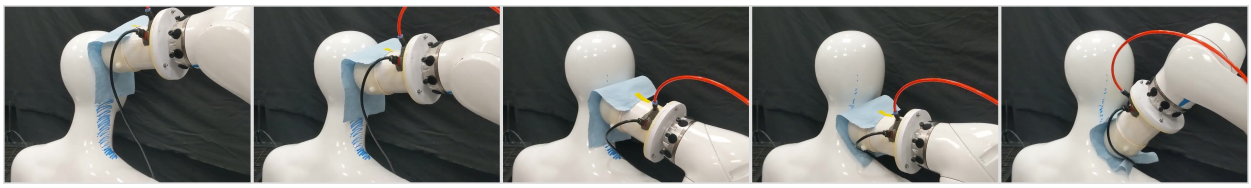


Figure 4.11: The robot cleans the side of a mannequin's head and neck. ©2022 IEEE.

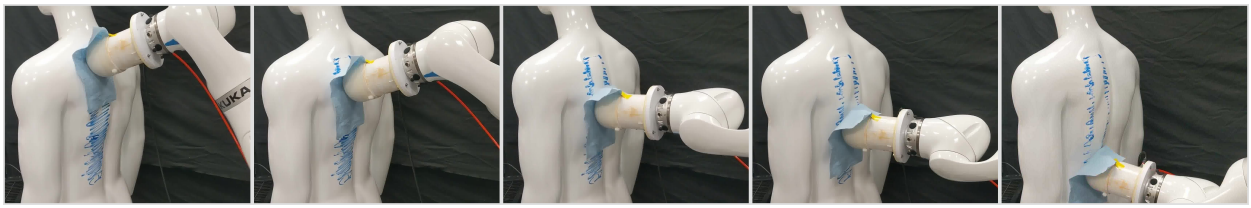


Figure 4.12: The robot cleans down the length of a mannequin's back. ©2022 IEEE.

Surface following

Next, we demonstrate wiping on the neck and back regions of a mannequin with a dry wipe cloth, where there is no longer a prominent edge to follow. In our current system, the practitioner must first set a plane of motion to define the end effector trajectory. In Fig. 4.11, the robot cleans the side of the mannequin's head and neck with the coronal plane as the motion constraint. In Fig. 4.12, the robot wipes down the mannequin's back with the motion constrained to the sagittal plane. Constant contact is maintained and localized to the soft membrane, and the dry erase markings are cleanly removed as a result (Fig. 4.13).

4.4 Discussion

We demonstrate the utility of a general robotic system that successfully performs contour-following tasks on a set of unseen objects and body parts. Throughout all trajectories, the robot is able to maintain contact under high enough forces to scrub off impurities such as black pepper and dry erase markings. These interactions are inherently safe due to all contact being restricted to the compliant silicone membrane of the sensor.

There are several significant avenues for future work. In this iteration, all interactions have been quasi-static with stationary objects and bodies. However, we want to incorporate human motion and use faster control rates to better adapt to the true nature of bathing tasks. Furthermore, the addition of an external camera system would provide beneficial global scene information to remove all need for a priori specifications. For example, cameras can determine in which direction and with what orientation the robot should move in order to make initial contact with the target. The camera can also segment an object or body into regions and automatically determine the plane normals $\vec{n}_p$ that would allow the robot to cover the most area when performing surface following. Currently, the robot is commanded to stop the wiping trajectory when the practitioner detects that it has reached the end of

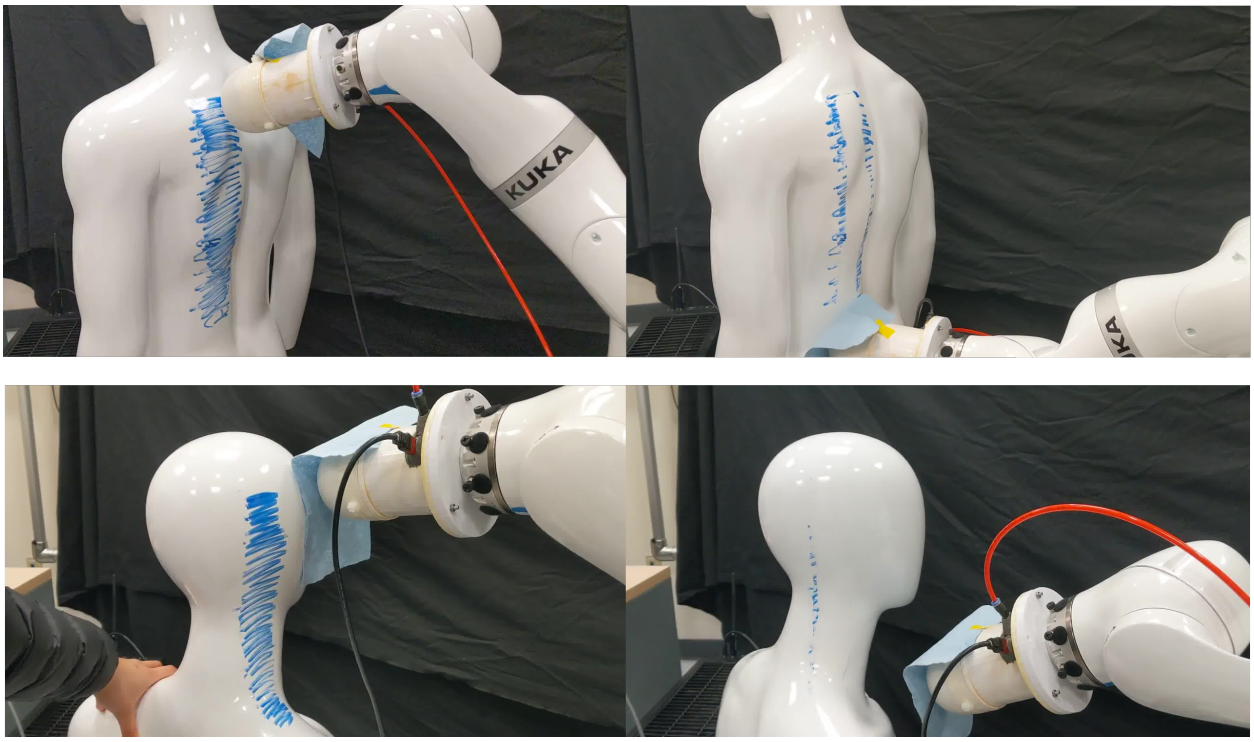


Figure 4.13: Before and after the wiping of dry erase marker off the back (top) and the head and neck (bottom) of a mannequin. ©2022 IEEE.

the object or body. An external vision system can not only automate this detection, but also aid in path planning to cover larger regions (e.g., the entire back rather than a single strip down one side).

Chapter 5

Learning from Demonstration

This chapter is adapted in part from "Robot Learning from Demonstration with Tactile Signals for Geometry-Dependent Tasks" [59], written in collaboration with Ruzena Bajcsy.

5.1 Introduction

For robots to be useful in the real world, such as in the unconstrained environments of our homes, they should reliably meet the vast needs of users. Because these needs are ever-changing and cannot be fully anticipated, these robots should be able to *adapt* to different requirements. Furthermore, there should be an intuitive and simple interface through which robots can *learn* these desired behaviors. Robot learning from demonstration (LfD), also known as programming by demonstration and imitation learning, allows users avoid directly programming these behaviors onto the robot [2]. Rather, robots by means of LfD can observe human-provided demonstrations, learn to generalize the task to new situations, and then execute the proper behavior itself.

LfD has historically been successful in the encoding and reproduction of robot motion. However, accounting purely for propioception is not sufficient for many tasks, which often require physical interaction with objects and/or other agents. Thus, end effector wrench data has also been used in an LfD framework to capture salient interaction signals and enable task reproduction [1]. However, tasks cannot be fully represented with just one time-dependent wrench reading without a priori assumptions. One such assumption is that the *geometry* of the object with which the robot interacts is known and unchanging. In general, simple wrench signals cannot encode the rich tactile profiles that arise during robotic contact. Humans are equipped for dextrous manipulation through sensitive shape and force-sensing in our fingertips— for robots to perform similarly, their own tactile sensing must contribute to their learning. In this work, we explore how to perform LfD with manipulation tasks that are dependent on the *geometries* of the objects being manipulated. We employ the use of a custom tactile sensor and discuss how to effectively featurize its readings to best enable successful LfD with a low number of demonstrations. In addition, we validate the framework applied to two exemplary geometry-dependent tasks. The first is an edge-following task, and the second is a mixed manipulation task with objects of different shapes.

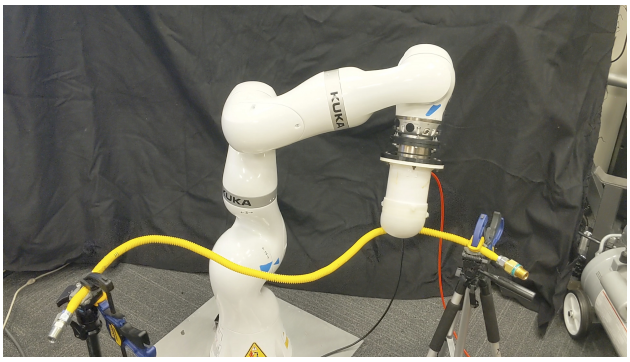


Figure 5.1: Geometric tactile features for interaction-based tasks, eg. edge-following, cannot be encoded by wrench signals. ©2020 IEEE.

Robot learning from demonstration is comprised of three building blocks— observation, encoding, and execution [7]. Though LfD is useful on a higher level of abstraction, such as in decomposing a complicated task into a sequence of subtasks [45], much work including ours performs learning on the lower sensory and motor levels. Over time, LfD works have advanced from motion replication to robust frameworks that use vision and force-torque sensing for interaction with the environment. Learning force-control tasks in particular has been successful for tasks such as ironing a shirt or pushing open a door [1]. While force sensing is imperative for the proper learning and execution of any interactive task, all prior LfD works have only recorded end effector wrenches, either via an attached force torque sensor or by reconstruction using joint torques. However, wrench readings yield no insight about contact geometry, which is a crucial element for interaction in general.

There are various interfaces through which human teachers can provide demonstrations to the robot. A natural method is for the teacher to perform the task themselves and have their actions recorded by a motion-capture system [79]. When additional sensing is required of the task, force and pose sensing have been combined through a force-sensing glove worn by the demonstrator [31]. However, there is a correspondence problem in mapping the sensing-action spaces between the teacher and robot [118]. This correspondence problem is removed when demonstrations are performed on the robot itself via teleoperation or kinesthetic teaching. Haptic devices have been used to successfully teach force-based tasks through teleoperation while allowing the demonstrator to feel the forces sensed by the robot [142]. On the other hand, while the demonstrator physically guides the robot in kinesthetic teaching, the contact forces acting on the robot are physically relayed to the teacher as well. However, when end effector wrenches are inferred using joint torques, torques applied during kinesthetic teaching can confound this inference. This is not an issue for our particular setup, and thus all our demonstrations were performed kinesthetically.

Compared to deep learning techniques, where high-dimensional data is fed as input at the cost of requiring much of it, simpler LfD models achieve generalization with less data with low dimensionality. The dimensionality of the demonstration data can be reduced by either dropping sensor readings that contain the least mutual information with the action outputs [142], or by projecting them into a latent space [14]. Dynamic motion primitives (DMP) is an LfD approach first proposed in [67] that, after having broken down a complex

task into a sequence of motion primitives, models them as a set of mass-spring-damper systems. It is a general framework for all motor and sensor trajectories, time-invariant, and guaranteed to converge. Meanwhile, the hidden Markov model (HMM) provides a unified approach that can encode multiple motion alternatives in one model and can be updated with partial demonstrations [15]. Though HMM does not have a convergence guarantee, it is advantageous in that it does not require the a priori representation of the task structure as with DMP. Thus, we chose to adopt the HMM approach for its flexibility.

5.2 Hidden Markov Model Framework

Our LfD framework closely follows the proposed approach by Calinon et al [15], which models the task as an HMM and predicts best actions during execution with a modified version of Gaussian mixture regression (GMR) [159] that accounts for hidden state transitions.

Hidden Markov Model Setup

The crux of the HMM approach in a LfD setting is to estimate the joint distributions of the sensor input and action outputs observed in a demonstration. Each teacher demonstration is comprised of of a sequence of physically observable sensor input-action output pairs $O = (O_1 = o_1, ..., O_T = o_T)$ of length T. Under a hidden Markov model, we assume that the system at each step t is in one of N unobservable, or hidden states $Q_t \in \{1...N\}$. We assume that the probability of an observation o_t at time t given that the hidden state is j, denoted as $b_j(o_t)$, is normally distributed according to parameters associated to j:

$$b_j(o_t)p(O_t = o_t|Q_t = j) = \mathcal{N}(o_t|\mu_j, \Sigma_j) \tag{5.1}$$

We denote the entire set of observation distribution parameters by $B = \{\mu_j, \Sigma_j\}$. Furthermore, the chain of hidden states is assumed to be both time-homogeneous and Markovian, so that the state transitions can be modeled by a time-independent stochastic transition matrix $A = \{a_{i,j}\}p(Q_t = j|Q_{t-1} = i)$. At the initial state of $t = 1$, the hidden state distribution is described by $\pi_1 = p(Q_1 = i)$. Thus, every hidden Markov model is defined by a set of parameters $\Lambda = (A, B, \pi)$. These parameters are found by employing the Baum-Welch algorithm to find the maximum likelihood estimate Λ^* given the observation sequence O:

$$\Lambda^* = \underset{\Lambda}{\operatorname{argmax}}\, p(O|\Lambda) \tag{5.2}$$

Baum-Welch Algorithm

The HMM forward variable $\alpha_i(t)$ is the probability at time t of having observed the partial starting sequence $\{O_1 = o_1, ..., O_t = o_t\}$ and being in state i, and is defined recursively as such:

$$\alpha_i(t) = p(O_1 = o_1, ..., O_t = o_t, Q_t = i|\Lambda) \tag{5.3}$$

$$\alpha_i(1) = \pi_i b_i(o_1) \tag{5.4}$$

$$\alpha_j(t+1) = b_j(o_{t+1}) \sum_{i=1}^{N} \alpha_i(t) a_{ij} \tag{5.5}$$

The backward variable $\beta_i(t)$ is the probability of observing the partial ending sequence $\{O_{t+1} = o_{t+1}, ..., O_T = o_T\}$ having been in state i at time t, and is defined recursively:

$$\beta_i(t) = p(O_{t+1} = o_{t+1}, ..., O_T = o_T | Q_t = i, \Lambda) \tag{5.6}$$

$$\beta_i(T) = 1 \tag{5.7}$$

$$\beta_i(t) = \sum_{j=1}^{N} a_{ij} b_j(o_{t+1}) \beta_j(t+1) \tag{5.8}$$

We further define $\gamma_i(t)$ as the probability of being in state i at time t given observation sequence O:

$$\gamma_i(t) p(Q_t = i | O, \Lambda) = \frac{\alpha_i(t)\beta_i(t)}{\sum_{j=1}^{N} \alpha_j(t)\beta_j(t)} \tag{5.9}$$

Also, $\xi_{ij}(t)$ is defined as $p(Q_t = i, Q_{t+1} = j | O, \Lambda)$, the probability of being in state i at time t and then j at $t+1$:

$$\xi_{ij}(t) = \frac{\gamma_i(t) a_{ij} b_j(o_{t+1}) \beta_j(t+1)}{\beta_i(t)} \tag{5.10}$$

Similarly to typical expectation-maximization algorithm for Gaussian mixtures, the HMM parameter update equations are as follows, where m indexes the M demonstrations $\{O_1, ...O_M\}$:

$$\pi_i = \frac{\sum_{m=1}^{M} \gamma_i^m(1)}{M} \tag{5.11}$$

$$\mu_i = \frac{\sum_{m=1}^{M} \sum_{t=1}^{T_m} \gamma_i^m(t) o_t^m}{\sum_{m=1}^{M} \sum_{t=1}^{T_m} \gamma_i^m(t)} \tag{5.12}$$

$$\Sigma_i = \frac{\sum_{m=1}^{M} \sum_{t=1}^{T_m} \gamma_i^m(t) (o_t^m - \mu_i)(o_t^m - \mu_i)^T}{\sum_{m=1}^{M} \sum_{t=1}^{T_m} \gamma_i^m(t)} \tag{5.13}$$

$$a_{ij} = \frac{\sum_{m=1}^{M} \sum_{t=1}^{T_m - 1} \xi_{ij}^m(t)}{\sum_{m=1}^{M} \sum_{t=1}^{T_m - 1} \gamma_i^m(t)} \tag{5.14}$$

In our work, we initialize π to be uniform and μ_i to be the i^{th} mean of N-means clustering on the demonstration data. Covariances Σ_i are initialized randomly.

Action Prediction during Task Replication

Each observation from demonstration m at elapsed time t is comprised of a sensor input-action output pair. That is, $o^m(t) = \begin{bmatrix} S^m(t) & A^m(t) \end{bmatrix}^T$. Thus, parameters in B can be broken into components corresponding to the input and output spaces of the observations, such that:

$$\mu_i = \begin{bmatrix} \mu_i^S & \mu_i^A \end{bmatrix}^T \tag{5.15}$$

$$\Sigma_i = \begin{bmatrix} \Sigma_i^{SS} & \Sigma_i^{SA} \\ \Sigma_i^{AS} & \Sigma_i^{AA} \end{bmatrix} \tag{5.16}$$

One method by which to predict the most likely action output $\hat{A}$ from a sensor input S is to treat the hidden states as constituents of a Gaussian mixture model and apply Gaussian mixture regression [159]. However, this regression does not leverage the sequential encoding of the learned HMM. Calinon et al [15] proposed an extension to GMR, called GMRa, which instead considers not only the state vector itself, but also the sequential information captured through the forward variable in the HMM. Under this formulation, the mixing weight for each state becomes the relative likelihood of the corresponding forward variable component in the sensor input space. That is,

$$h_i(S_t) = \frac{\mathcal{N}(S_t|\mu_i^S, \Sigma_i^{SS}) \sum_{j=1}^{K} h_j(S_{t-1}) a_{ji}}{\sum_{k=1}^{K} \left[\mathcal{N}(S_t|\mu_k^S, \Sigma_k^{SS}) \sum_{j=1}^{K} h_j(S_{t-1}) a_{jk} \right]} \tag{5.17}$$

$$h_i(S_1) = \frac{\pi_i \mathcal{N}(S_1|\mu_i^S, \Sigma_i^{SS})}{\sum_{k=1}^{K} \left[\pi_k \mathcal{N}(S_1|\mu_k^S, \Sigma_k^{SS}) \right]} \tag{5.18}$$

5.3 Experimental Setup

System Design

Our robotic platform comprises a soft tactile sensor mounted as the end effector of a KUKA LBR iiwa 14 R820 (Fig. 2.2). This sensor contacts the environment with a hemispherical palm-like 3 mm thick membrane molded from Ecoflex$^{\text{TM}}$ 50 silicone. It is sealed onto an air-pressurized cylindrical capsule, and inflates slightly upon internal pressurization. The sensor is fixed at one pressure state, at which the membrane's inner radius is inflated to 50 mm compared to its neutral 47 mm radius when free air exchange is permitted. Contact states are measured by imaging the inside of the membrane using an embedded miniature time-of-flight Camboard pico-flexx depth-sensing camera from PMD Technologies. Demonstrations are performed kinesthetically by the researcher and collected at 5 Hz.

Input and Output Featurization

The inputs of the learning model are derived from the high-dimensional 224×171 sized point cloud readings of the tactile sensor. As shown in the example point cloud reading in Fig. 5.2, we can isolate the portion of the point cloud that is deformed upon contact through comparison with the undeformed reading. From this deformed point cloud, signals correlated with the physical contact state related to forces and geometry can be obtained. We extract a a set of signals including the 3D centroid $\vec{c}$, the three variance values $\vec{\lambda}$ along its three principal component vectors, as well as the three principal component vectors $\vec{e_1}$, $\vec{e_2}$, and $\vec{e_3}$. The subset of these signals to be used as features in the HMM are handpicked based on domain knowledge, rather than through quantitative methods such as mutual information analysis. Consistency in the direction of the component vectors is maintained by setting

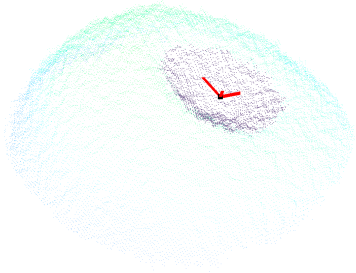


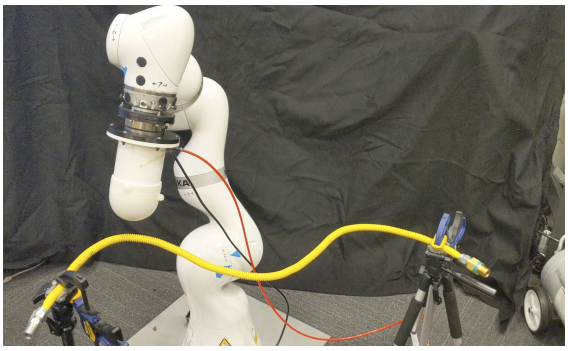
Figure 5.2: Example point cloud reading of the tactile sensor, with black deformed portion and red principal component vectors. ©2020 IEEE.

their signs so their largest element is positive. All sensor input quantities are expressed in the world frame rather than in the camera frame. When the deformed point cloud is empty, as is the case when nothing contacts the sensor, by default the centroid is the origin of the camera frame, and the principal component vectors are the camera frame coordinate axes. When transformed to the world frame, these default quantities during non-contact contain propioceptive information regarding the pose of the end effector. By not including explicit robot position and orientation signals in the input, we ensure that during contact, the robot responds solely to tactile signals rather than overfits to the demonstrated trajectories. The action output of the model is a 6-dimensional instantaneous velocity of the end effector with respect to the world frame with both linear $\vec{v}$ and angular $\vec{\omega}$ components approximated by the rate of change of time-adjacent readings. After collecting demonstrations, all datapoints for both inputs and outputs are first linearly normalized from 0 to 1 before learning the HMM. During task execution, outputs are predicted for normalized inputs, then sent to the robot as velocity commands at 5 Hz.

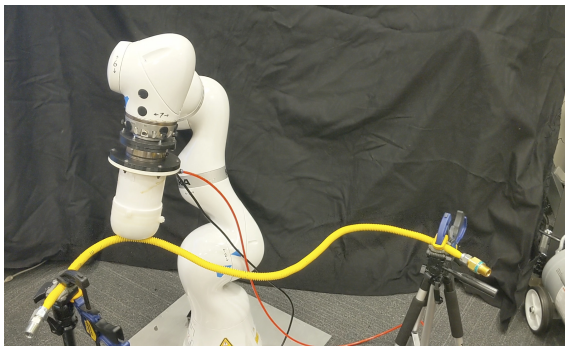
5.4 Application: Edge-Following

Task Demonstrations

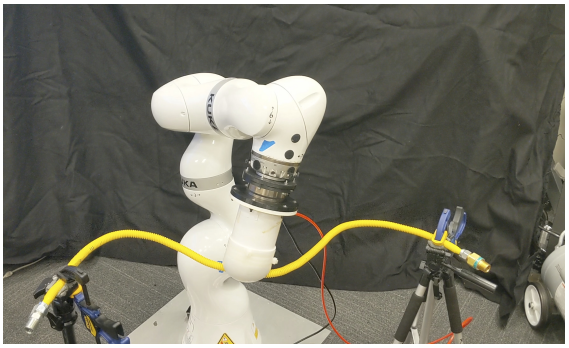
The first task taught to the robot is to follow an edge. During the demonstration phase, we lower the robot end effector until the palm sensor contacts an edge positioned directly below the starting point, and then we slide the sensor along that edge. The physical edge used in the demonstration is the length of a straight copper pipe with a 0.5" diameter. We perform 6 different demonstrations with the pipe fixed at different orientations and positions in space. By convention, the traversal direction is such that the y component in the world frame is positive. We consider the edge-following task successful as long as the sensor consistently makes contact with the edge, meaning that the end effector orientation is not critical to the task. The six demonstration end effector position trajectories are plotted in Fig. 5.4, with arrows along the trajectory showing the $\vec{e_1}$ reading at that position. The demonstration consists of two intuitive phases. The first is the lowering phase, during which no contact



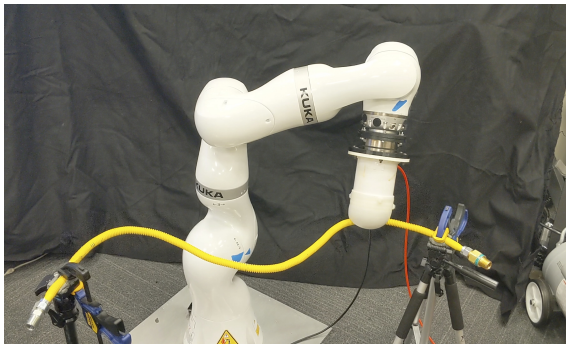
(a) Robot begins its policy rollout above the obstacle.



(b) Robot lowers itself onto the pipe and follows the edge.



(c) Contact is maintained during a dip in the $-z$ direction



(d) Robot continues to contact the pipe along a lateral bend.

Figure 5.3: Policy rollout of edge following HMM on a curved pipe. ©2020 IEEE.

with the edge is made. The next is the phase during which the sensor contacts the edge and traverses along it.

Learned HMM

For both tasks in this work, sensor input features are handpicked to minimize dimensionality but maximize relevance to the action outputs. In edge-following, it is important to detect only whether the sensor is in contact with something, and if so, along which direction the longest edge is "pointing." Thus, we choose the inputs to be the principal variances and the first principal component vector, such that $S(t) = \begin{bmatrix}\lambda_1(t) & \lambda_2(t) & \lambda_3(t) & e_{1x}(t) & e_{1y}(t) & e_{1z}(t)\end{bmatrix}^T$. The centroid is excluded from the feature vector because the location of contact is not important to the task. Although the demonstrations are performed on a straight pipe, we want the robot to be able to follow an edge with arbitrary curvature. To this end, we learn an HMM with $N = 2$ states— one that corresponds to the robot lowering onto the edge, and one to perform edge following. We try to avoid overfitting to the demonstration by limiting the number of hidden states to be learned. We plot the same demonstrations in $\vec{\lambda}$ space in Fig. 5.5 with each reading colored according to the most likely hidden state it is in according to Eq. 5.17. Matching up to our intuition, the two HMM states do correspond to the

physical pre-contact (Q_2) and edge traversal conditions (Q_1), where the yellow star marks the reading in the absence of contact, i.e., $\vec{\lambda} = \vec{0}$.

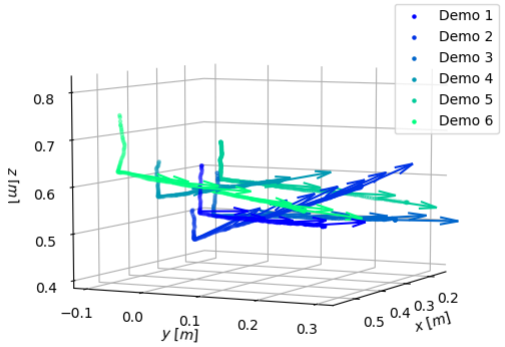


Figure 5.4: Demonstration trajectories for the edge-following experiment along a straight pipe. Round markers are end effector positions, and are $\vec{e_1}$. ©2020 IEEE.

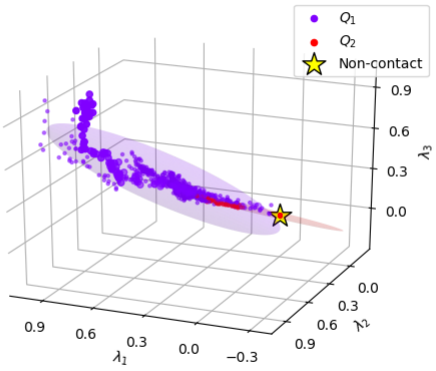


Figure 5.5: Demonstration data alongside HMM states in normalized $\vec{\lambda}$ space. Ellipsoids depict the 95% confidence region of the corresponding Σ_i. ©2020 IEEE.

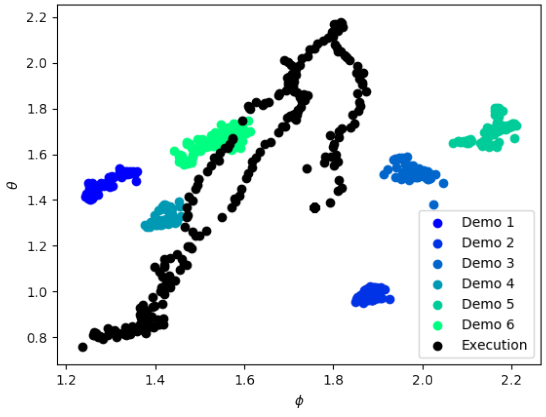


Figure 5.6: Principal component $\vec{e_1}$ observed in demonstration and execution phases, mapped onto spherical coordinates. ©2020 IEEE.

Task Execution

To verify generalizability of the edge-following HMM, we perform the task execution on a flexible gas connector also with a 0.5" diameter that is bent to be curved in 3D space. As

pictured in the rollout in Fig. 5.3, the robot is successful in initiating and maintaining contact throughout the length of the pipe. We can visualize how the HMM is able to generalize to new observations in Fig. 5.6, where the spherical coordinates of $\vec{e_1}$ observed in both the demonstration and rollout phases have little overlap. Note that in the teaching phase, each demonstration performs edge traversal in only one direction. Thus, during the execution phase, the robot is unable to follow an edge that goes back in an opposite direction it has experienced before, such as along a closed path. Furthermore, since no stopping condition is observed in the demonstrations, as long as the robot senses an edge, it will move. If contact is broken, it believes it is again in Q_2 and lowers itself until a new edge is found.

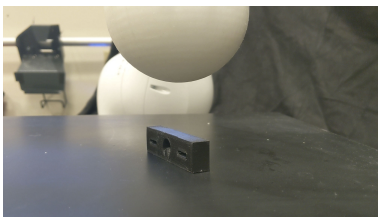
5.5 Application: Mixed Object Manipulation

Task Demonstrations

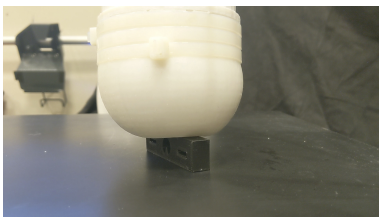
The robot is also taught to manipulate objects with different geometries with the following desired behavior:

- Starting from a point above a predefined quadrant of the table, lower the robot palm and make contact with the object directly underneath.
- If the object is capsule-shaped, *push* it through a predefined corridor and off the table into a bin.
- If the object is a rectangular prism, *rotate* it in place until it its length is parallel to the table edge closest to the robot.

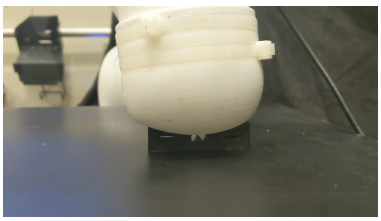
Example demonstrations for both subtasks are in Fig. 5.8 and 5.7 for the capsule and prism subtasks respectively. Nine different objects pictured in Fig. 5.9 are used in demonstrating this task. Of the nine, eight are capsules (with diameters and heights ranging from [1...5] and [1.5...1.7] cm), and one is a rectangular prism of dimensions $7.5 \times 1 \times 2$ cm. Ten demonstrations of each type (pushing and rotating) are performed on the robot, and a 4-state HMM is learned from the data. The sensor input used here is the 9-dimensional feature vector containing $\vec{c}$, $\vec{\lambda}$, and $\vec{e_1}$.



(a) Palm begins above prism.

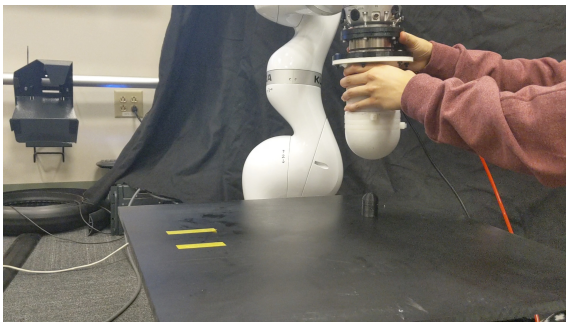


(b) Palm lowers and contacts prism.

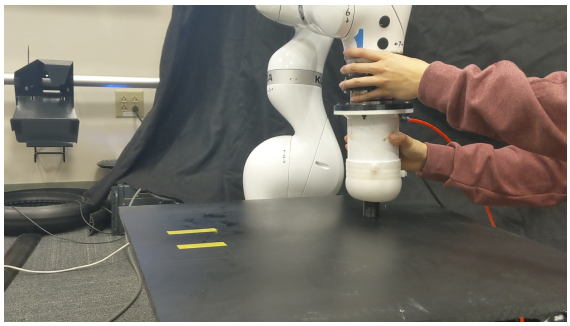


(c) Palm rotates until desired orientation is achieved.

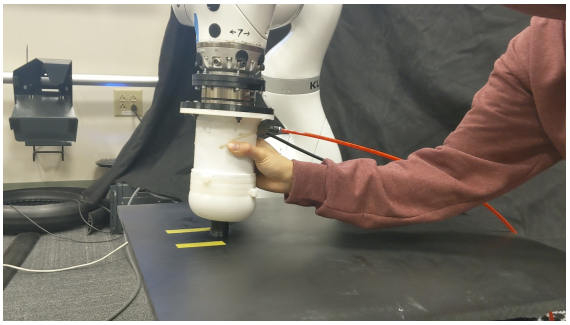
Figure 5.7: Kinesthetic demonstration showing how to rotate a prism to its desired orientation.



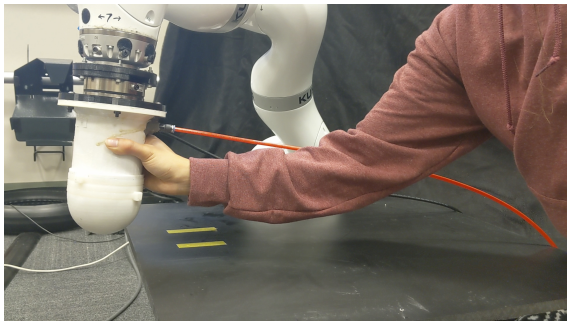
(a) Palm begins above the capsule.



(b) Palm is lowered onto the capsule.



(c) Palm pushes the capsule through a corridor.



(d) Palm pushes capsule off of table.

Figure 5.8: Kinesthetic demonstration showing how to push a capsule through a corridor and off the table. ©2020 IEEE.

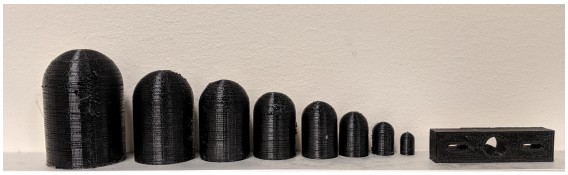


Figure 5.9: Set of objects used in the mixed manipulation task. ©2020 IEEE.

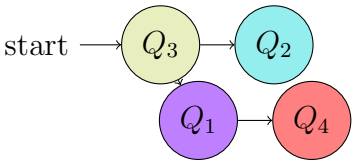


Figure 5.10: 4-state HMM with the most likely exit transitions depicted as as edges. ©2020 IEEE.

Learned HMM

A 4-state HMM is learned from the collected demonstrations, and the most likely exit transitions are depicted in the finite state machine in Fig. 5.10. Figs. 5.11 and 5.12 plot the centroids observed in the demonstrated manipulations with the capsules and prism respectively, colored according to the most likely hidden state. Both subtasks start off in the same state, Q_3, which corresponds to the pre-contact lowering phase. Upon contact, the system moves into either the pushing phase Q_1 or the rotating phase Q_2 depending on the shape of the object. Fig. 5.13 depicts the demonstrations in normalized $\vec{\lambda}$ space and shows that there is a clear distinction between $\vec{\lambda}$s when contacting the capsules versus the prism that the HMM was able to differentiate. Once the robot pushes the object off the table so that

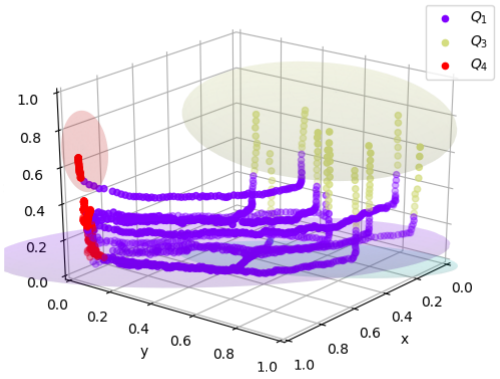


Figure 5.11: Capsule demonstrations alongside HMM states in normalized $\vec{c}$ space. Ellipsoids depict 95% confidence region. ©2020 IEEE.

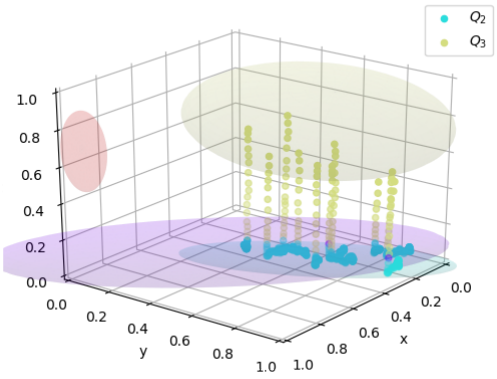


Figure 5.12: Prism demonstrations alongside HMM states in normalized $\vec{c}$ space. Ellipsoids depict 95% confidence region. ©2020 IEEE.

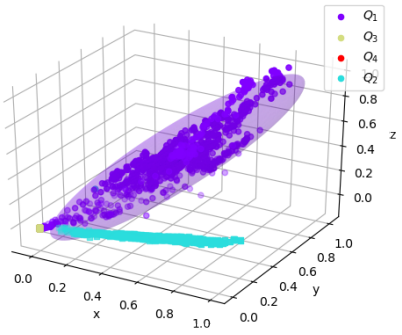


Figure 5.13: Demonstrations alongside HMM states in normalized $\vec{\lambda}$ space, with ellipsoids at the 95% confidence region. The two most significant visible states Q_1 and Q_2 clearly separate the two object types. ©2020 IEEE.

it is no longer sensed by the palm, it immediately transitions from Q_1 to the post-pushing phase Q_4.

Task Execution

In the task execution phase, we give the robot 10 unseen scenes (5 for each subtask type) from randomly sampled starting poses in the predefined table quadrant, and find that it is successful in completing its task in all 10 trials. For the capsule subtasks, the executed trajectory successfully eases the capsule through the corridor and off the table just like in the

demonstrations (Fig. 5.14). During the prism subtask executions, the rectangle's orientation also converges correctly (Fig. 5.15), where the final orientations from the execution matches up well with that which is observed in demonstration. Granted, the starting states in the execution phase are not vastly different from what is observed in the demonstration phase. For example, if the task is initialized at the opposite side of the table, around which the robot never recorded any demonstration, we find that it either does not manage to squeeze through the corridor properly, or even veers off in a wrong direction. Because the $\vec{c}$ centroid readings are highly correlated with the actual position of the end effector, we would have to ensure a higher variance in positions during the demonstration phase to encourage generalization.

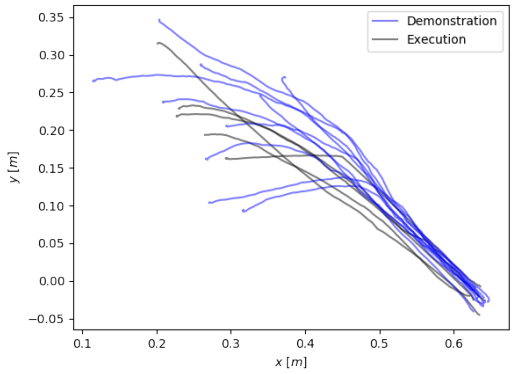


Figure 5.14: Centroid trajectories recorded in demonstration and execution phases during contact with the capsule mapped onto the normalized xy space. ©2020 IEEE.

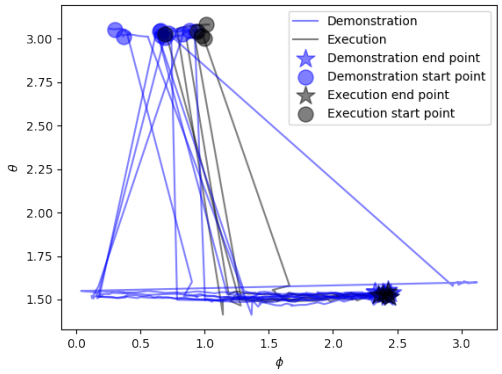


Figure 5.15: Principal components $\vec{e_1}$ recorded in demonstration and execution phases during contact with the prism, mapped onto spherical coordinates. ©2020 IEEE.

5.6 Discussion

In this work, we successfully incorporate geometry-based tactile sensing to learn the demonstrated tasks of mixed object manipulation and edge following. Rather than program these behaviors explicitly, practitioners can kinesthetically teach these behaviors to the robot. This is a very promising proof of concept for teaching general tactile-related tasks to robots, but both tasks in this work are very simple by design. The sensed geometries are either static, as with the capsules and prism, or slow-evolving, as with the curved tube. It would be compelling to investigate dynamic tasks where the contacts evolve rapidly. In addition, our method of featurizing tactile signals using PCA of the deformed membrane works well for simple shapes, but likely is not sufficient for more complicated shapes or for distributed contacts. Finally, we use just one tactile sensing palm that can only push and twist. A robot hand with multiple tactile sensing fingers would pose a more interesting learning problem that requires higher dimensional coordination that is not explored here.

Part II

Deformable Object Grasping

Chapter 6

DefGraspSim: Physics-based Simulation of Grasp Outcomes

This chapter is adapted in part from "DefGraspSim: Physics-based Simulation of Grasp Outcomes" [63], written in collaboration with Yashraj Narang, Clemens Eppner, Balakumar Sundaralingam, Miles Macklin, Ruzena Bajcsy, Tucker Hermans, and Dieter Fox.

6.1 Introduction

From clothing, to plastic bottles, to humans, deformable objects are omnipresent in our world. A large subset of these are *3D deformable objects* (e.g., fruits, internal organs, and flexible containers), for which dimensions along all 3 spatial axes are of similar magnitude, and significant deformations can occur along any of them. Robotic grasping of 3D deformables is underexplored relative to rope and cloth, but remains critical for applications like food handling [43], robotic surgery [153], and domestic tasks [148]. Compared to rigid objects, grasping 3D deformable objects faces 4 major challenges to which we respond with 4 key contributions.

First, classical analytical metrics for grasping rigid objects (e.g., force/form closure) do not typically consider deformation of the object during the grasp [148]. Yet, deformations significantly impact the contact surface and object dynamics. For example, one can grasp a soft toy haphazardly; however, if the toy were rigid, it would no longer conform to one's hands, and many grasps would become unstable. Conversely, one can grasp a rigid container haphazardly; however, if the container were flexible, grasps along its faces would crush its contents. Unlike for rigid objects, the success of a 3D deformable grasp depends on properties such as compliance. We thus propose a set of diverse *performance metrics* that quantify deformable grasp outcomes (Sec. 6.5), such as stability, deformation, and stress. Performance metrics may also compete (e.g., a stable grasp may induce high deformation).

Second, performance metrics may be partially or fully unobservable (e.g., volumetric stress fields), requiring estimators in the real world. Previous works have addressed this by formulating quality metrics, which we refer to as *grasp features*: simple quantities a robot can measure before pickup that can predict performance metrics. Whereas grasp features have

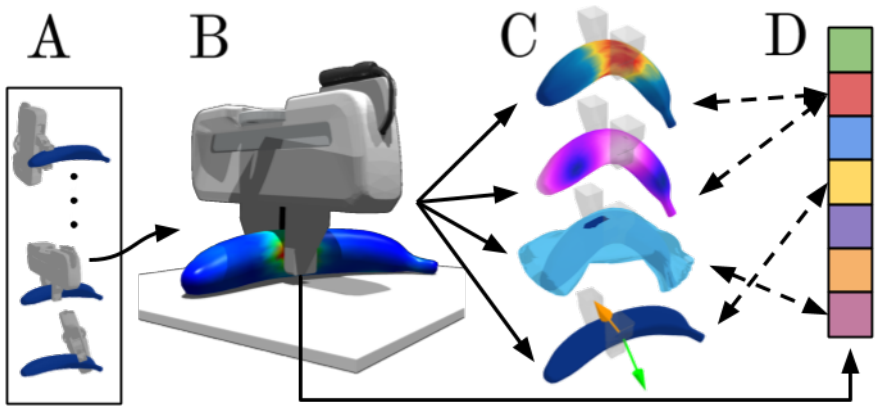


Figure 6.1: (A) For a broad set of candidate grasps on a deformable object, (B) we simulate the object's response with FEM, (C) measure performance metrics (e.g., stress, deformation, controllability, instability), and (D) identify pre-pickup grasp features that are correlated with the metrics. Our simulated dataset contains 34 objects, 6800 grasp experiments, and $1.1M$ unique measurements. ©2022 IEEE.

predominantly been designed for rigid objects, we propose a set of grasp features compatible with deformables (Sec. 6.6).

Third, there exists neither a general framework to evaluate arbitrary deformable grasps (i.e., via performance metrics and grasp features), nor an exhaustive dataset of deformable grasp experiments. We thus release DefGraspSim, a codebase that allows users to automatically perform an exhaustive set of FEM-based grasp evaluations on arbitrary 3D objects.[1] Simulation offers multiple advantages: it extends classical analytical methods through accurate modeling of object deformation, enables safe execution of experiments, and provides full observability of performance metrics. We also conduct a large-scale simulation-based study of 3D deformable object grasping on 3D deformables varying in geometry and elasticity (Fig. 6.1), and provide this live dataset of 34 objects, 6800 grasp evaluations, and $1.1M$ corresponding measurements. This is the largest deformable object grasping dataset in existence.

Fourth, simulation studies do not necessarily correspond to real-world behavior. To address this gap, we perform a pilot sim-to-real study on results generated by DefGraspSim, and demonstrate that simulated results show reliable correspondence with real-world experiments (Sec. 6.8).

We believe these 4 main contributions are an important milestone towards developing a complete learning and planning framework for grasping 3D deformables.

6.2 Related Work

Modeling techniques. With over three decades of development, methods in rigid-object grasp planning range from model-based approaches using exact geometries [93, 37, 115] to data-driven approaches without full models [85, 25, 104, 77, 120, 98]. Rigid-body grasping

[1] `https://sites.google.com/nvidia.com/defgraspsim`

Table 6.1: Comparisons between Isaac Gym and other robotics simulators that support both 3D deformable bodies and actuator interactions.

Simulator	Interactions	3D geometries	Materials	Underlying model	Observable states	Processor
MuJoCo [164]	soft-rigid, rigid-rigid	Boxes, cylinders, ellipsoids	Homogeneous isotropic elastic	Mass-spring with surface nodes	Nodal positions	CPU
PyBullet 3 [23]	soft-rigid, soft-soft, rigid-rigid	Arbitrary geometries	Homogeneous isotropic elastic/hyperelastic	Mass-spring or Neo-Hookean volumetric FEM	Nodal positions, contact points & forces	CPU
IPC-GraspSim [73]	soft-soft	Arbitrary geometries	Homogeneous isotropic elastic	Incremental potential contact model	Nodal positions, velocities, and accelerations	CPU
Isaac Gym [111]	soft-rigid, rigid-rigid	Arbitrary geometries	Homogeneous isotropic elastic	Co-rotational linear volumetric FEM	Nodal positions & velocities, contact points & forces, element stress tensors	GPU

simulators such as GraspIt! [114] and OpenGRASP [87] have been used to develop many of these algorithms. For 3D deformable objects, rigid-body approximations can lead to efficient simulations [136]; however, continuum models are preferred, as they can represent large deformations and allow consistent material parameters without an explicit model-fitting stage [30]. 3D continuum models include Kelvin-Voigt elements governed by nonlinear PDEs [53], mass-spring models [82], 2D FEM for planar and ring-like objects [71], and gold-standard 3D FEM [95]. However, many powerful FEM simulators used in engineering and graphics (e.g., Vega [9]) do not feature infrastructure for robotic control, such as built-in joint control. For comprehensive reviews of 3D deformable modeling techniques, please refer to [4, 180]. In this work, we use the GPU-accelerated Isaac Gym simulator to analyze grasp interactions with deformable objects. In Table 6.1, we compare Isaac Gym to other robotics simulators, including MuJoCo [164] and PyBullet [23], which have successfully modeled deformable ropes and cloths using rigid-body networks with compliant constraints [172, 106, 91, 22], but have recently started to support 3D deformable objects as well.

Performance metrics. Prior works have evaluated 3D deformable-object grasps using performance metrics based on pickup success, strain energy, deformation, and stress. Success-based metrics include the minimum force required by a particular grasp, which is calculated via real-world iterative search [53] and FEM [82, 95]. Success depends on both object geometry and stiffness (e.g., a cone can be picked up only when it can deform to the gripper) [95]. Metrics based on strain energy (i.e., elastic potential energy in the object) have served as proxies for an object's stability against external wrenches. In 2D the deform closure metric generalizes rigid form closure [13] and quantifies the positive work required to release an object from a grasp [44]. It is optimized by maximizing strain energy without inducing plastic deformation. Similarly, for thin and planar 2.5D objects, grasps have been selected to maximize strain energy under a fixed squeezing distance [71]. Deformation-based metrics have also been proposed for cups and bottles to detect whether contents are dislodged during lifting and rotation [176]. Finally, stress-based metrics have been proposed to avoid material fracture, but were evaluated only on rigid objects [132].

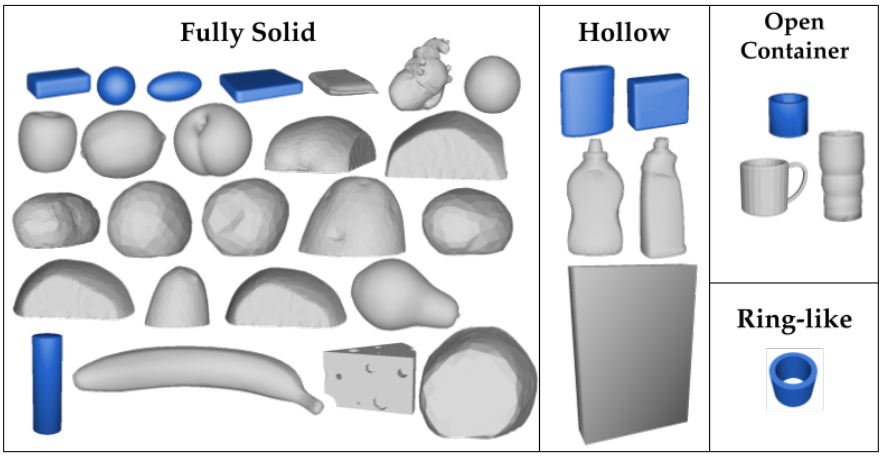


Figure 6.2: The 34 evaluated objects grouped by geometry and dimension (shown to scale). Objects in blue are self-designed primitives; those in gray are scaled models from open datasets [16, 175, 109, 69]. ©2022 IEEE.

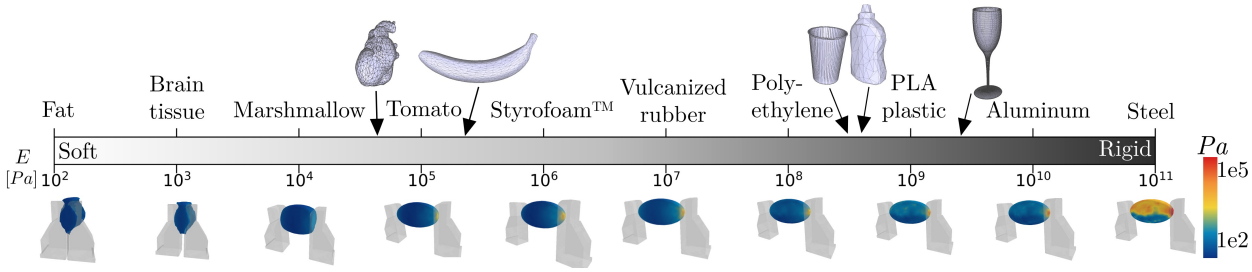


Figure 6.3: Young's modulus E for various materials (adapted from [108]). (Top): real-world objects and their typical E. (Bottom): Stress distributions of an ellipsoid under 1 N of grasp force. Soft ellipsoids undergo large deformations; rigid ones have high-stress regions. ©2022 IEEE.

Grasp features. Many grasp features to predict grasp performance have been previously investigated on rigid objects. Features include force and form closure [37] and grasp polygon area [116], and their predictive accuracy has been tested under different classification models [143]. A thorough survey on rigid grasping features can be found in [141]. However, grasp features for deformables have only been explored in one study, which measured the work performed on containers during grasping to predict whether its liquid contents would be displaced [176].

6.3 Grasp Simulator

We use the FEM-based simulator Isaac Gym [111] to simulate grasps on 3D deformable objects. FEM is a variational numerical technique that divides complex geometrical domains into simple subregions and solves the weak form of the governing partial differential equations over each region. In FEM simulation, a deformable object is represented by a

volumetric mesh of *elements*; the object's configuration is described by the element vertices, known as *nodes*. We use Isaac Gym's [111] co-rotational linear constitutive model of the object's internal dynamics coupled to a rigid-body representation of the robotic gripper via an isotropic Coulomb contact model [157]. A GPU-based Newton method performs implicit integration by solving a nonlinear complementarity problem [102]. At each timestep, the simulator returns element stress tensors and nodal positions, which are used to calculate grasp metrics. With sufficiently small timesteps and high mesh density, FEM predictions for deformable solids can be extremely accurate [140, 125]. We simulate at a frequency of $1500Hz$. The simulator executes at 5-10 fps, and each grasp experiment (Sec. 6.4) requires around 2 to 7 minutes to run.

We evaluate a set of 34 3D deformable objects comprising both simple object primitives and complex real-world models, categorized by geometry and dimension (Fig. 6.2). We process object surface meshes in Blender to smooth sharp edges to avoid stress singularities, and reduce node count where possible to optimize speed. We then convert these into tetrahedral meshes using fTetWild [55].

For all experiments our objects have density $\rho = 1000\frac{kg}{m^3}$, Poisson's ratio $\nu = 0.3$, coefficient of friction $\mu = 0.7$, and Young's modulus $E \in \mathcal{E} = \{2e4, 2e5, 2e6, 2e9\}Pa$. $\mathcal{E}$ covers a wide range of real materials, from human skin ($\sim 10^4 Pa$) to ABS plastic ($\sim 10^9 Pa$) (Fig. 6.3). The target squeezing force on an object is $F_p = 1.3 \times \frac{mg}{\mu}$ (where m is mass and g is gravity), which is the force required to support the object's weight with a factor of safety. For a fixed E, increasing μ decreases F_p as well as the induced deformation. This effect is essentially the same as if μ is fixed while E is increased, since an elastically stiffer object will also deform less for the same F_p applied. Thus, we fix μ and vary E.

6.4 Grasp Experiments

We perform simulated grasping experiments within Isaac Gym on 34 objects using the widely used Franka parallel-jaw gripper. To generalize to other parallel-jaw grippers we remove all specialized gripper features. For each object, we generate a diverse set of 50 candidate grasps with an antipodal sampler [33]. Each object initially rests atop a horizontal plane; we disable gripper collisions with the plane to test the full spatial distribution of grasps by allowing grasps to come from underneath. Prior to grasping, the pre-contact nodal positions and element stresses of the object are recorded. The gripper is initialized at a candidate grasp pose, then squeezes using a force-based torque controller to achieve the target grasp force, F_p. Once F_p converges, the grasp features are measured. Then, one of the following experiments (Fig. 6.4) is executed: *pickup, reorientation, linear acceleration, and angular acceleration.*

1) Pickup. The platform lowers to apply incremental gravitational loading to the object. Pickup is a success if the gripper maintains contact with the object for 5 seconds. If so, stress and deformation fields are recorded, and stress, deformation, and strain energy performance metrics are computed.

2) Reorientation. The grasp force is increased from F_p to F_{slip}, the minimum force required to counteract rotational slip. The platform is lowered until the object is picked up. The gripper rotates the object to 64 unique reorientation states. We record stress and deformation

fields at each state, and compute deformation controllability as the maximum deformation over all states. F_{slip} is estimated by approximating each gripper contact patch as 2 point-contacts that oppose the gravitational moment. The gripper rotates the object about each of 16 vectors regularly spaced in a unit 2-sphere at angles $k\pi/4, k \in [1..4]$ for a total of 64 unique reorientation states.

3) Linear acceleration. The gripper linearly accelerates along each of 16 unique direction vectors as in the reorientation experiment. Each vector has a complement pointing in the opposite direction; thus, this method generalizes the cyclic shaking tests from previous works [33]. The acceleration is recorded at which at least one finger loses contact with the object. Linear instability is computed as the average loss-of-contact acceleration over all directions. The robot moves at $1000\frac{m}{s}^3$ jerk in a gravity-free environment, corresponding to a linearly increasing acceleration. We impose a realistic upper acceleration limit of $50\frac{m}{s}^2$ ($\approx 5g$).

4) Angular acceleration. The gripper rotationally accelerates about 16 unique axes. Angular instability is computed as the average loss-of-contact acceleration over all axes. The robot accelerates at $2500\frac{rad}{s}^3$ jerk; to mitigate undesired linear acceleration, the midpoint between the fingers is the center of rotation. The angular loss-of-contact threshold is limited to $1000\frac{rad}{s}^2$ (i.e., the linear acceleration limit, scaled by the $0.04m$ max. finger displacement, a reference moment arm).

Controller Details. A contact force-based torque controller is used to achieve the desired grasp forces. A low-pass filter is first applied to the contact force signals due to high frequency noise that prevails from small numerical fluctuations in position, especially at higher moduli. For the three experiments involving post-pickup manipulation, the finger joints are frozen immediately after pickup to maintain the gripper separation.

Codebase. We release the code to replicate our experiments on arbitrary objects,[2] and a full software flow diagram is shown in Fig. 6.5. Candidate grasps can either be user-defined or generated using our grasp sampling module. We also encourage the use of external software such as Blender to preprocess object models (e.g., mesh simplification and edge filleting) to improve simulation speed and convergence.

6.5 Grasp Performance Metrics

During the preceding experiments, we measure the following 7 performance metrics to comprehensively evaluate grasp outcomes. These metrics include high-dimensional field quantities (e.g., stress fields), which are unexplored in rigid object modeling. Note that linear and angular instability are separate metrics. During each grasp experiment, we also measure a set of grasp features (Sec. 6.6), which capture low-dimensional state information about the grasp and have potential to be used as predictors of the final performance metrics.

Pickup success: A binary metric measuring whether an object is lifted from a support plane.

Stress: The element-wise stress field of an object when picked up. Exceeding material thresholds (e.g., yield stress, ultimate stress) leads to permanent deformation, damage, or

[2] `https://github.com/NVlabs/DefGraspSim`

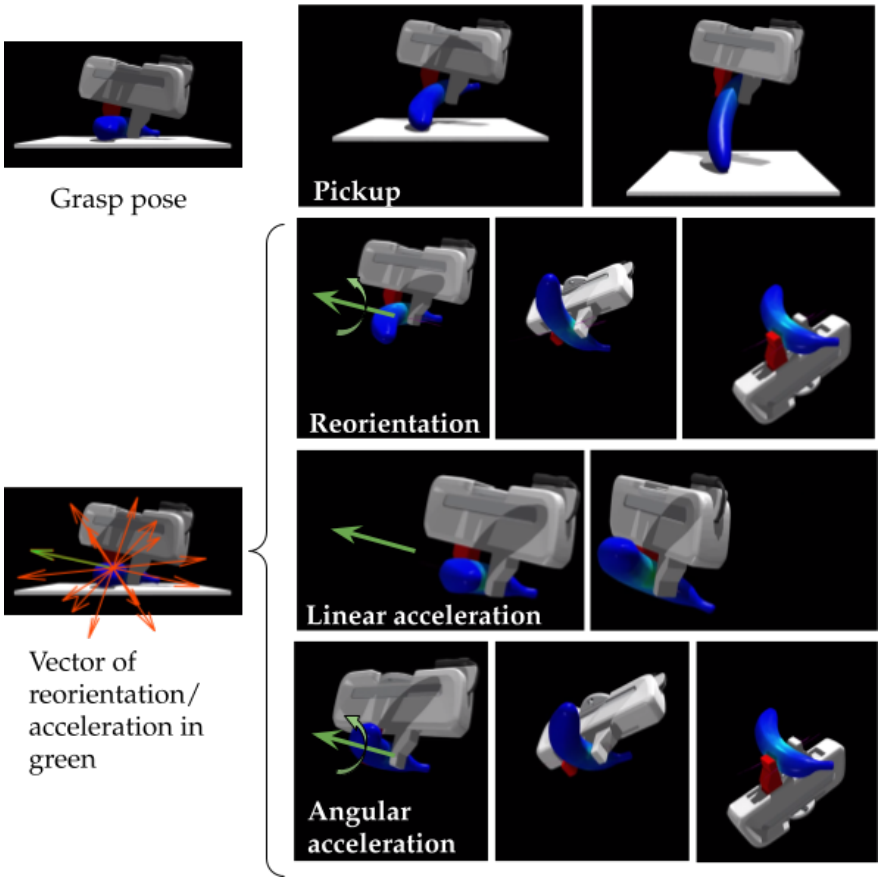


Figure 6.4: Example frames from the execution of four different experiments per grasp on a banana: pickup, reorient, twist (angular acceleration), and shake (linear acceleration). ©2022 IEEE.

fracture; examples include creasing of boxes, bruising of fruit, and perforation of organs. We convert each element's stress tensor into von Mises stress, a scalar quantity that quantifies whether an element has exceeded its yield threshold. We then measure the maximum stress over all elements, since real-world applications typically aim to avoid damage at any point.

Deformation: The node-wise displacement field of the object from pre- to post-pickup, neglecting rigid-body transformations. Deformation must often be minimized (e.g., on flexible containers with contents that can be damaged or dislodged). To compute this field, the difference between the pre- and post-pickup nodal positions is calculated, the closest rigid transform is determined [155], and the transform is subtracted. We compute the ℓ^2 norm of each node's displacement and measure the maximum value over all nodes.

Strain energy: The elastic potential energy stored in the object (analogous to a Hookean spring). Conveniently, this metric penalizes both stress and deformation. The strain energy is given by $U_e = \int_V \sigma^T \epsilon dV$, where σ, ϵ, and V are the stress tensor, strain tensor, and volume, respectively.

Linear and angular instability: We define instability as the minimum acceleration applied to the gripper (*along* or *about* a vector for linear and angular instability, respectively) at

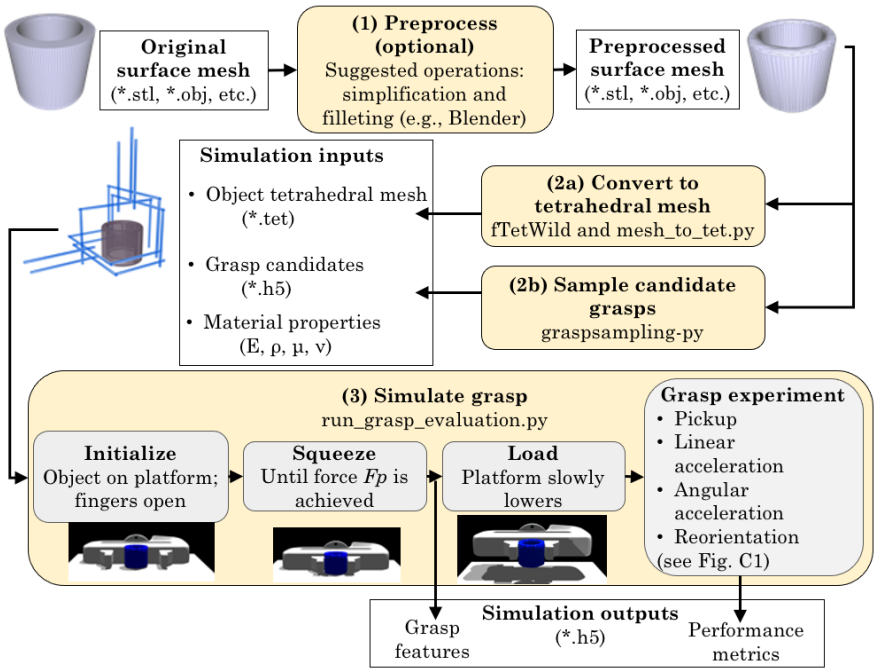


Figure 6.5: Software flow diagram of the DefGraspSim codebase. ©2022 IEEE.

which the object loses contact (i.e., separates along the gripper normal, or slides out of the gripper). This measures how easily an object is displaced from the grasp under external forces.

Deformation controllability: We define deformation controllability as the maximum deformation when the object is reoriented under gravity. (An example of shape change induced during reorientation is shown in Fig. 6.6.) Depending on the task, it may be useful to either minimize or maximize deformation controllability. For example, to reduce the effects of post-grasp reorientation on deformation, minimizing this metric allows the object to behave rigidly after pickup. Alternatively, to augment the effects of post-grasp reorientation (e.g., during insertion of endoscopes), we may maximize it instead. Our notion of deformation controllability is different from the classical notion (i.e., the ability to achieve any robot state in finite time). Here, we are not modifying robot controllability by changing actuation, but modifying object controllability by changing the number of possible deformation states.

6.6 Grasp Features

The 7 grasp features are recorded after applying the grasping force F_p, but before loading (Fig. 6.5). All can be measured by common real-world sensors (e.g., encoders, cameras, and tactile arrays) and are summarized in Table 6.2 and Fig. 6.7, along with references to existing works from which they are derived. See [141] for a full review of grasp features on rigid objects.

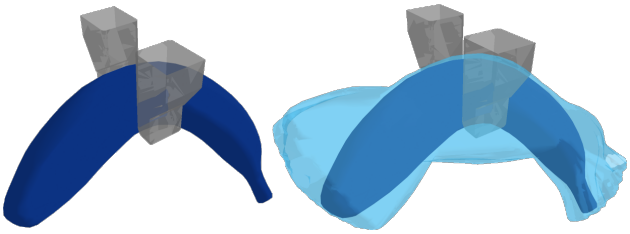


Figure 6.6: Illustration of deformation controllability. A soft banana-shaped object under pickup (left); the union of all shape configurations achieved under reorientation, superimposed in light blue (right). ©2022 IEEE.

Table 6.2: Grasp features, their descriptions, and existing works from which they are derived.

Feature	**Abbr.**	**Definition and Relevance**	**Usage in Literature**
Contact patch distance to centroid	*pure_dist*	Distance from the center of each finger's contact patch to the object's center of mass (COM) (Fig. 6.7a), averaged over the two fingers.	[143, 27]
Contact patch perpendicular distance to centroid	*perp_dist*	Perpendicular distance from the center of each finger's contact patch to the object's COM (Fig. 6.7a), averaged over the two fingers; quantifies distance from lines of action of squeezing force.	[8]
Number of contact points	*num_contacts*	Number of contact points on each finger, averaged over the fingers; quantifies amount of contact made.	[143, 27]
Contact patch distance to finger edge	*edge_dist*	Distance from each finger's distal edge to the center of its contact patch (Fig. 6.7b), averaged over the two fingers.	[36]
Gripper squeezing distance	*squeeze_dist*	Change in finger separation from initial contact to the point at which F_p is achieved; quantifies local deformation applied to the object.	[176]
Gripper separation	*gripper_sep*	Finger separation upon achieving F_p; quantifies the thickness of material between the fingers at grasp.	[143]
Alignment with gravity	*grav_align*	Angle between the finger normal and the global vertical; grounds the grasp pose to a fixed frame (Fig. 6.7b).	[170]

(a) (b)

Figure 6.7: Four grasp features illustrated on a Franka gripper. ©2022 IEEE.

6.7 Example Simulation Results

To demonstrate the utility of DefGraspSim, we simulate grasps on 5 real object models with realistic material parameters (Fig. 6.3, top) and visualize the resulting performance metrics of stress, deformation, and linear stability (Fig. 6.8). The results are well-aligned with established mechanics principles. On the heart and wine glass (Fig. 6.8a), regions of high stress arise with reduced contact areas (e.g., at heart nodules and the stem of the glass) and curvature discontinuities (e.g., at the lip of the glass). On a mustard bottle and plastic cup (Fig. 6.8b), high deformations occur when contacting regions of low geometric stiffness (e.g., at the main face of the bottle and lip of the cup) and vice versa (e.g. at the base of the bottle and cup). On a banana (Fig. 6.8c), stability increases with friction μ under the same grasp pose and force. When μ is fixed, grasps closest to the ends of the fruit are least stable.

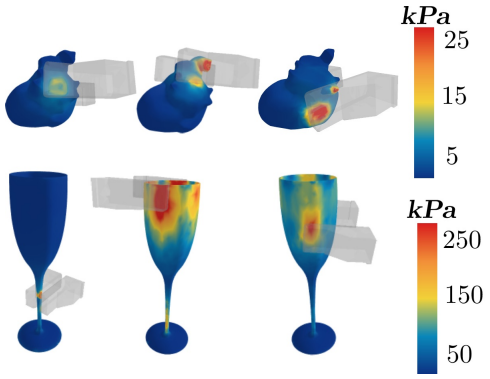
6.8 Sim-to-Real Accuracy

Although simulation is the primary focus of this work, we also investigate whether grasp outcomes predicted by DefGraspSim are faithful to the real world.

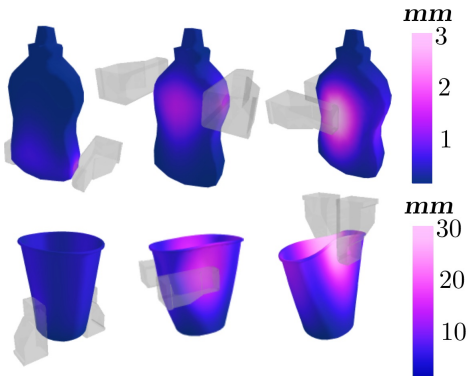
Tofu blocks. First, we test 3 grasps on real blocks of tofu under 1 and 2 N of applied force (Fig. 6.9). Simulated and real-world deformations exhibit strong similarities, and grasps achieve anticipated performance (e.g., grasps A and C, respectively, minimize and maximize sagging under the 2 force conditions). Also, permanent damage on the real-world tofu occurs under 2 N of applied force, with fracture occurring in grasps B and C. Although standard FEM cannot simulate fracture, simulated stresses for these grasps lie within the literature-reported range of breaking stress for tofu, around 3 kPa [163]. Moreover, fracture lines on the real tofu coincide with regions of stress higher than this threshold (i.e., along gripper edge under Grasp B; at tofu ends under Grasp C).

Latex tubes. Next, we perform 3 grasps on latex tubes of different geometry (Fig. 6.10). Again, simulated and real-world deformations are highly similar, including indentations and bulges localized to regions of contact; moreover, the vertical distance between the highest and lowest points of the tubes closely match. We also test the deformation controllability between grasp D (a middle grasp) and grasp F (an end grasp) on the thin tube (Fig. 6.11) by rotating the gripper by 90 degrees under each grasp. Deformation controllability is higher in grasp F, as the resulting angle swept out by the tube tip is only 47° (compared to 83° under grasp D), which indicates that more shape change is induced. These angle values also closely match those predicted by simulation.

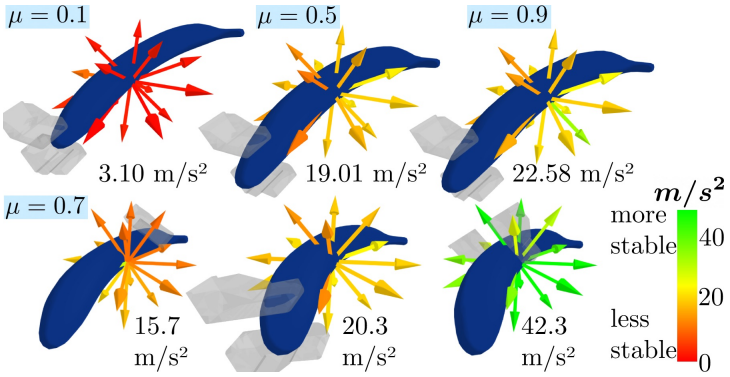
Bleach bottle. We also evaluate five grasps on a real bleach bottle (Fig. 6.12), which are ordered from G to K in ascending order of deformation imparted on the simulated version of the bleach bottle. Since deformation fields on the bottle are not readily accessible in the real world, we instead measure the volume of the bottle. After applying a fixed grasp force, the bottle is filled with rice, and the weight is recorded and divided by density. The resulting volume change of the real-world grasps is similar to simulated results, except that simulation incorrectly predicts that grasp H would impart more deformation than grasp G (Fig. 6.13). Cutting open the bottle reveals that material at the neck of the bottle is thicker (1 mm)



(a) Simulated stress fields after pickup for various grasps on a (top) heart and (bottom) wine glass. Objects are colored by the von Mises stress field.



(b) Simulated deformation fields after pickup for various grasps on a (top) mustard bottle and (bottom) plastic cup. Objects are colored by the l^2-norm of the deformation field.



(c) Linear stability of grasps on a banana at 4 N of grasp force under (top) the same grasp but variable friction μ, and (bottom) the same μ but variable grasps. Arrows are colored by the maximum acceleration in that direction before loss of contact. Number indicates the average acceleration at failure over all 16 directions.

Figure 6.8: Examples of simulated grasp outcomes on 5 objects, with visualizations of (a) stress, (b) deformation, and (c) linear stability.

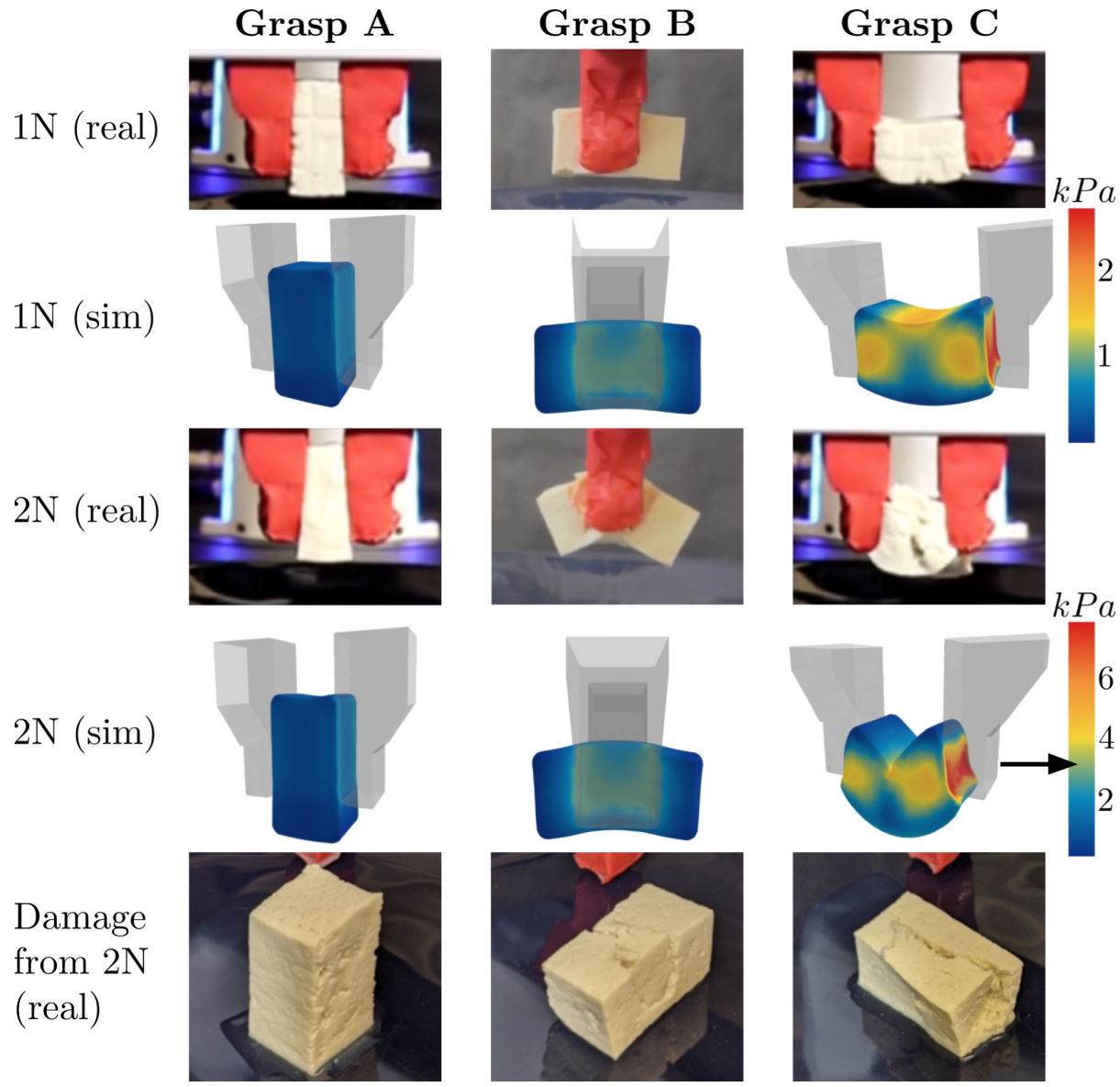


Figure 6.9: Three grasps tested on blocks of tofu (1 and 2 N of squeezing force) show similar outcomes in simulation and the real world. Real areas of fracture correspond to simulated stress greater than 3 kPa, the estimated breaking stress (denoted on color bar by black arrow).

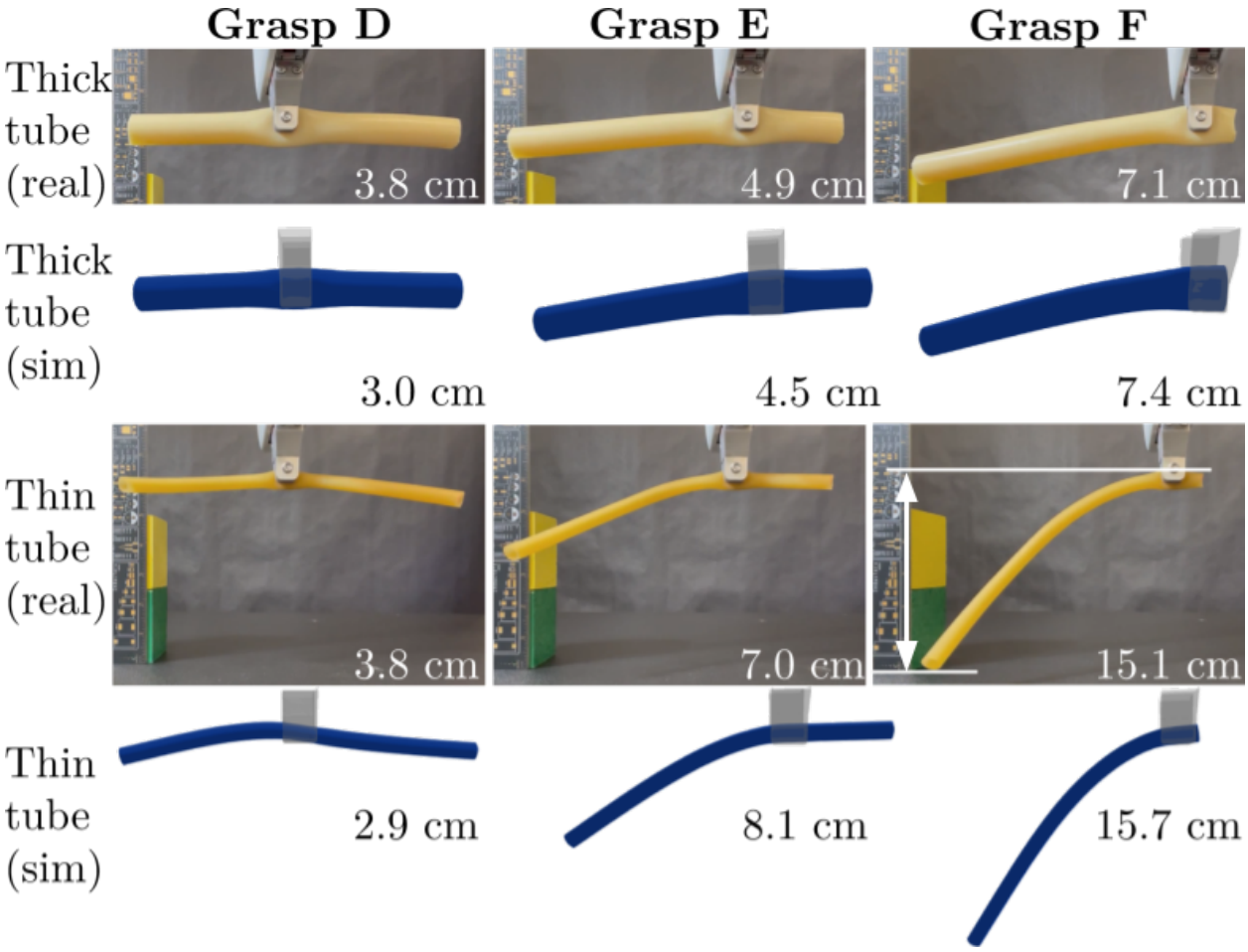


Figure 6.10: Three grasps tested on 2 real and simulated latex tubes under 15 N of gripper force. The vertical distance between the highest and lowest points of the tube is annotated. Localized deformation due to compression at the grippers is replicated in simulation. ©2022 IEEE.

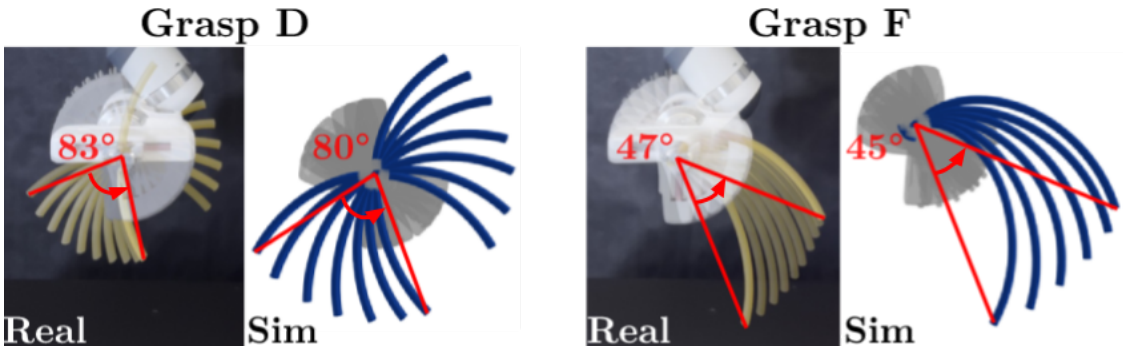


Figure 6.11: A middle grasp (grasp D) and end grasp (grasp F) under a counterclockwise 90° rotation of the gripper in the real world and in simulation. The angles swept out by the tube tip are marked in red. ©2022 IEEE.

Figure 6.12: Five tested grasps on a real bleach bottle. Grasps are also repeated in simulation. ©2022 IEEE.

than at the bottom (0.85 mm), whereas a uniform wall thickness is assumed in simulation. Thus, local stiffness higher on the bottle (including the contact region under grasp H) may be underestimated. This discrepancy in wall thickness also explains why the simulated grasps J and K predicted significantly more change in volume than in real life, as the geometric stiffness was underestimated within simulation.

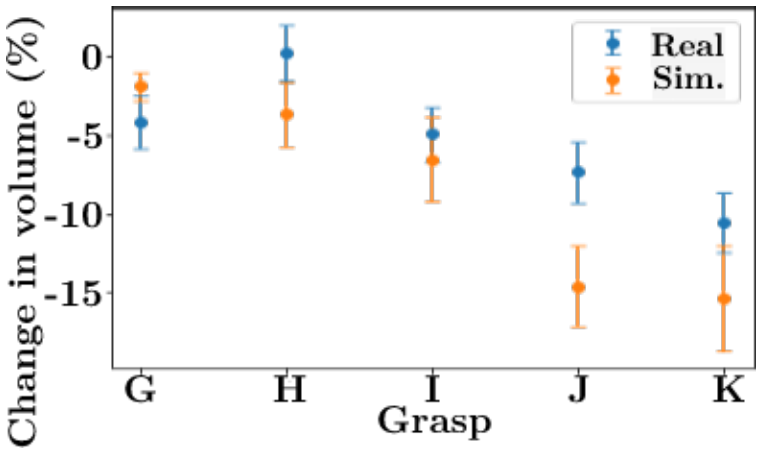


Figure 6.13: Percent volume change pre- and post-grasp for the real and simulated bleach bottles under grasps G to K. ©2022 IEEE.

Plastic cup. We test 4 grasps on a plastic cup for stability. In simulation, we run the linear acceleration experiment in the upward direction; in the real world, the cup is gradually filled with metal balls until contact is lost. Since the real Franka has no precise force controller, the actual gripper forces on the cup are unknown (unlike the prismatic tofu, the cup has complex geometry that makes force hard to analytically estimate from position inputs). Thus, while the ordering of grasps with respect to stability is consistent between real world and simulation, the weight values at failure are not (Fig. 6.14).

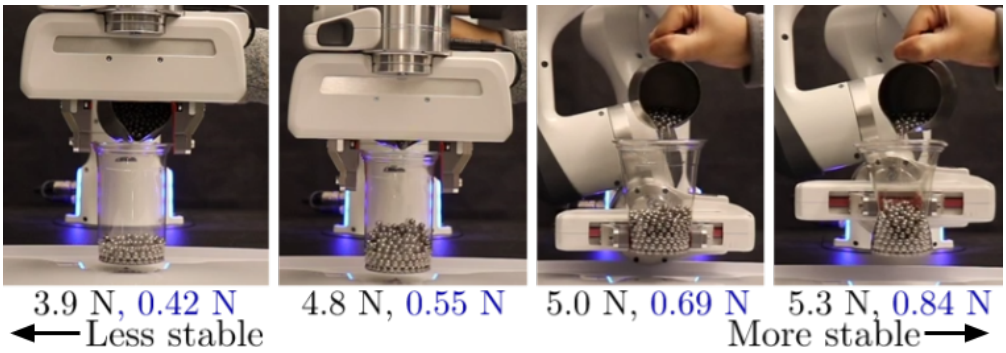


Figure 6.14: Under four grasps on a plastic cup, the maximum weight withstood before loss of contact is annotated for both the real world (black) and in simulation (blue). ©2022 IEEE.

6.9 Discussion

We propose and motivate performance metrics to describe deformable grasping outcomes as well as grasp features to serve as potential predictors of metrics. We then measure these quantities by conducting a battery of grasp simulations on 3D deformable objects and release our dataset of 6800 grasps and $1.1M$ measurements for further study, along with software that executes our experiments on arbitrary objects and material parameters. Finally, our simulated results are also shown to have good correspondence with real-world grasp outcomes. We envision DefGraspSim to be a useful research tool for the community working towards grasping deformable objects, with direct applications to a variety of compelling problems. For example, DefGraspSim can act as a data generator for learning representations of new high-dimensional features and metrics (e.g., for object and contact geometry, field quantities, etc.) for memory-efficient grasp planning. It can also be used as a testbed for customizing grasp experiments to create task-oriented planners (e.g., to minimize food deformation). To extend our sim-to-real pilot studies, we can use DefGraspSim to further perform rigorous, direct comparisons between simulation and reality on custom deformables of interest (e.g., on organs for robotic surgery). A wider variety of grasp conditions than presented here can also be simulated to improve grasp planning robustness to uncertainty in object material properties (e.g., via domain randomization). Finally, it can be used to generate training data for real-world system identification (e.g., tactile probing on unknown materials) and neural network-based grasp simulation for real-time grasp planning.

Chapter 7

DefGraspNets: Grasp Planning with Graph Neural Networks

This chapter is adapted in part from "DefGraspNets: Grasp Planning on 3D Fields with Graph Neural Nets" [62], written in collaboration with Yashraj Narang, Ruzena Bajcsy, Fabio Ramos, Tucker Hermans, and Dieter Fox.

7.1 Introduction

Deformable objects are omnipresent in our world, and grasping them is critical for food handling [43], robotic surgery [153], and domestic tasks [148, 183]. However, their physical complexities pose challenges for key aspects of grasp planning, including modeling, simulation, learning, and optimization. Deformable objects have infinite degrees of freedom and require continuum mechanics models to accurately predict their responses to body forces (e.g., gravity) and surface tractions (e.g., contacts). For deformable solids, continuum models can predict two field quantities critical for robot grasping, *stress* tensors and *deformation* vectors defined at every point in the object [162]. In general, low-stress grasps are desirable to reduce material fatigue from repeated grasping, or to avoid exceeding the yield stress of the object, at which point permanent deformation or failure occurs. Predicting deformation is also critical, especially when grasping containers. One may want to minimize the deformation on a box of crackers to avoid crushing the contents, or maximize the deformation on a bottle of ketchup to efficiently squeeze out the contents.

Although knowledge of stress and deformation fields is useful, deriving closed-form solutions is intractable for general cases. Moreover, direct real-world measurement is extremely difficult without cumbersome instrumentation. Consequently, robotic grasping has historically leveraged rigid-body models, for which deformation is ignored and object state can be simply described by 6D pose and velocity [123, 113].

On the other hand, we can use deformable-object *simulators* to access these quantities and plan grasps accordingly. However, such simulators rely on complex numerical models like the gold-standard 3D finite element method (FEM) [180, 4]. Although FEM can simulate the result of any grasp on a deformable object [63], each evaluation can take minutes on a

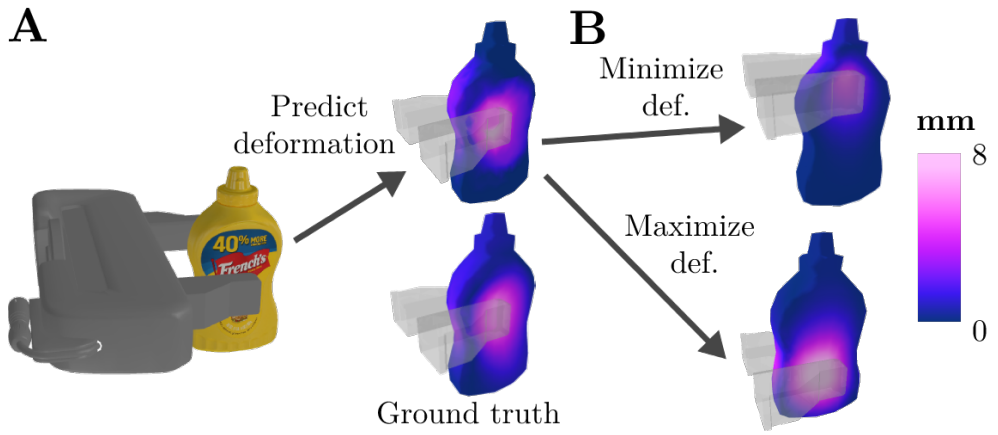


Figure 7.1: **(A)** DefGraspNets predicts the stress and deformation fields from grasping an unseen object 1500x faster than FEM, and **(B)** enables gradient-based grasp refinement to optimize these fields. ©2023 IEEE.

CPU-based industry-standard simulator [127] and seconds on a GPU-based robotics simulator [126], which is prohibitively slow for online grasp planning. Moreover, while differentiable simulators enable gradient-based optimization for parameter estimation [49] and control optimization [177, 49], few studies have explored their application to robotic grasping, and they are typically slower than standard FEM.

We propose *DefGraspNets*, a graph neural network that can enable grasp planning by predicting the stress and deformation fields resulting from grasps and allowing efficient optimization (Fig. 7.1). We demonstrate that this network is 1) **fast**, with a ∼1500x speed-up compared to a GPU-based FEM simulator, 2) **accurate**, with stress and deformation fields consistent with ground-truth FEM, 3) **generalizable**, with reliable rankings of grasp candidates over unseen poses, elastic moduli, in-category objects, and out-of-category objects, and 4) **differentiable**, enabling gradient-based optimization for grasp refinement. Finally, we conduct pilot studies that verify agreement of DefGraspNets, trained purely in simulation, with real-world outcomes. Data and code can be found on our website[1].

7.2 Related Work

Grasp planning has received significant attention in robotics [46, 123, 146, 21, 129]. Recent works leverage learning-based approaches to enable fast planning and generalization to novel objects [86, 121, 105, 99, 100]. We focus on grasp planning for 3D deformable objects. Unlike rope or cloth, 3D deformables have dimensions of a similar magnitude along all 3 spatial axes and can undergo significant deformations along any of them [63]. We review grasp planning for 3D deformables, as well as methods for predicting stress and deformation fields via graph neural networks and differentiable simulation.

[1] `https://sites.google.com/view/defgraspnets`

Grasp planning for deformable objects

Early works in grasp planning for deformable objects focused on finding *stable* grasps of planar objects, under which the object's strain energy would be maximized without inducing plastic deformation [44, 71]. This has since been extended to the 3D case, where novel time-dependent grasp quality metrics have been proposed to capture the evolution of contact states under deformation [154, 83].

Grasp planning for *deformation* of thin-walled containers (e.g., boxes, bottles) has also been explored. Given a 3D geometric stiffness map of the object, a minimal deformation grasp can be planned by localizing contact at high-stiffness regions. This map can be generated in simulation, via real-world probing [176], or from 2D images of the object via generative adversarial networks [110]. Grasp planning for *stress* has also been demonstrated via simulation on quasi-rigid objects using the boundary element method [132]. Finally, grasp planning for additional metrics can be performed with DefGraspSim, a 3D FEM-based grasp simulation framework [63]. For every grasp, it evaluates success, stability, stress, deformation, strain energy, and controllability.

These methods vary not only in the planning metric, but also in the type of computation required. Some require FEM simulation of the beginning of the interaction (e.g., just past initial contact) [71, 154, 83, 176] or the full interaction [132, 63], whereas others use neural networks [110]. Yet, all of these planners can only evaluate or predict the outcome of a candidate grasp, and cannot optimize grasps through gradient-based methods.

Graph neural networks for deformable-object interaction

Graph neural networks (GNNs) have been used to efficiently learn dynamics models for granular solids, deformable solids, and fluids [92, 149, 168, 133, 152]. Inspiring our work, MeshGraphNets [133] used GNNs to learn accurate dynamics for deformable solids using mesh-based representations, training from an industry-standard FEM solver. It predicted deformation and stress on a 3D deformable plate with kinematically-actuated colliding shapes and achieved evaluation speeds up to two orders of magnitude faster than the solver. RoboCraft [152] used GNNs to learn how plasticine-like objects with particle representations deform under interaction with a robotic gripper, training from visual input. Whereas MeshGraphNets used forward passes through the networks to predict dynamics, RoboCraft also used backwards passes to perform gradient-based trajectory optimization, molding the plasticine into a desired shape.

Our work also utilizes a GNN as a surrogate simulator for dynamics predictions. Unlike MeshGraphNets, which uses N-step rollouts to predict a final state via intermediate steps, DefGraspNets performs direct, one-step predictions of the final state. One-step prediction ensures that gradients are only propagated once through the network rather than over tens or hundreds of steps, mitigating vanishing or exploding gradients [94]. Furthermore, we focus on quasistatic rather than dynamic grasping; the ability of multi-step rollouts to predict object and controller dynamics offers limited advantage. Our ablation study verifies that using single-step predictions in our setting performs better than multi-step predictions (c.f. Sec. 7.8).

Like RoboCraft, we design our network to include gripper actions in order to perform gradient-based optimization for grasp planning. Unlike RoboCraft and MeshGraphNets, we use force rather than position commands for our actuators, as force commands are implemented in notable industrial grippers [81, 131, 150] and are preferable for grasping (as opposed to applications like shape control). Gripper force determines whether the grasp will overcome the object's gravity, and gripper position cannot indicate force without additional knowledge (e.g., contact area, object stiffness). In addition, when grasping stiffer objects, position commands can induce high torques that can damage both the object and gripper. We also generalize our network to different elastic moduli, which was not explored in prior works.

Differentiable simulators

Differentiable simulators for rigid and deformable bodies allow gradients of output variables (e.g., poses, velocities, or deformation fields of objects) to be computed with respect to input variables (e.g., control inputs or material parameters) [101, 39, 57, 50, 174, 70, 56]. Such simulators enable gradient-based optimization for control optimization [177, 70, 64, 56, 42], parameter estimation [49, 42, 56], and inverse design [178, 56].

There are 4 main strategies to realize a differentiable simulator or equivalent model: 1) finite-differencing a non-differentiable simulator, which has unfavorable $\mathcal{O}(n)$ scaling to an n-dimensional input space [11, 112], 2) analytically or automatically differentiating a simulator that smoothly approximates spatial or kinetic discontinuities (e.g., penalty-based contact forces and smooth friction models [49, 42], which may introduce inaccuracies or require tuning), 3) training a deep network with physically-based loss functions [139, 72], which has seen limited use for contact dynamics [134], and 4) training a deep network on datasets from a non-differentiable simulator, primarily with graph-based inductive biases [10, 92, 149, 133].

For our application, we aim to simulate robotic grasping of 3D deformable objects. Thus, we focus on gold-standard 3D FEM simulation of deformable objects with contact. For this application, strategy 2 has been explored in a handful of recent works [49], including differentiable projective dynamics [29, 137]. However, such simulators typically execute substantially slower than real-time (especially including a backward pass), and only one has realized differentiable FEM and contact modeled via the full nonlinear complementarity problem (NCP) [52] with both static and dynamic friction [137].

In this work, we explore strategy 4, training for the first time on a GPU-accelerated robotics FEM simulator [111] that addresses the full NCP [103] and has been experimentally validated across multiple studies [63, 127, 126]. To our knowledge, this effort also comprises the first application of such methods to robotic grasping of 3D deformable objects.

Strategy 2 has often been favored over strategy 4 due to the former's potential for generalizing to arbitrary physics [29, 137]. Nevertheless, we show for the first time that strategy 4, through judicious selection and scaling of training data, can indeed generalize to novel grasps, elastic moduli, in-category objects, and out-of-category objects. Furthermore, the trained networks can execute 2 to 3 orders of magnitude faster than the reference simulator (i.e., faster than real-time).

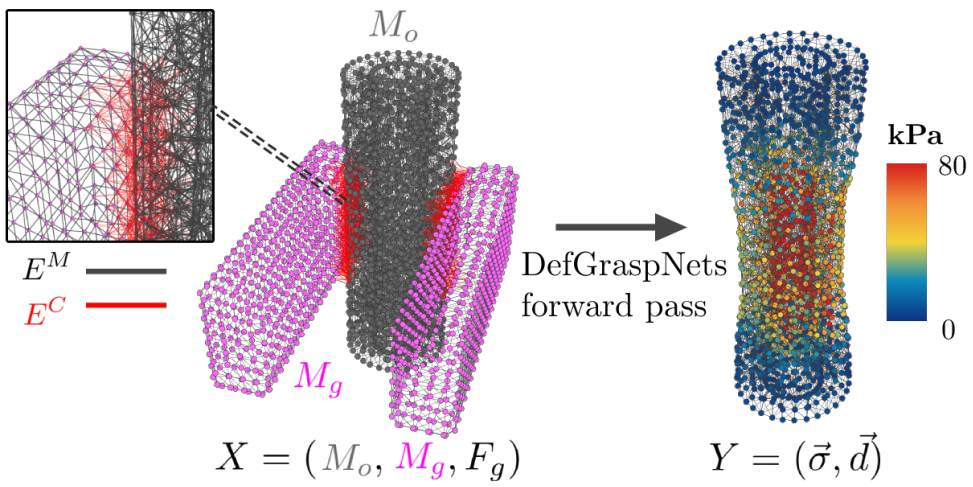


Figure 7.2: Given a candidate grasp state X consisting of an object mesh M_o, gripper mesh M_g, and grasp force F_g, DefGraspNets generates contact edges E^C and predicts output Y consisting of a stress field $\vec{\sigma}$ and deformation field $\vec{d}$ defined at each node of the object mesh. ©2023 IEEE

7.3 The DefGraspNets Model

Here, we explain the GNN structure of DefGraspNets, including the input and output representations. We detail training data generation in Sec. 7.4 and explain how we use DefGraspNets within a grasp planning algorithm in Sec. 7.5.

Summary of inputs and outputs

DefGraspNets takes as input a *candidate grasp state* $X = (M_o, M_g, F_g)$ comprising a mesh M_o of a deformable object in its pre-contact state, a mesh M_g of the gripper fingers upon initial contact[2], and a total normal grasp force scalar F_g. A mesh is a collection of vertices and undirected edges that connect them. For M_o, these vertices and edges form tetrahedral elements that define the volumetric geometry of the object. For M_g, the vertices and edges form triangular elements that define the surface geometries of the fingers.

DefGraspNets converts the candidate grasp state X into a multigraph G (Sec. 7.3), mapping the gripper-object contact interactions onto a graph structure. The multigraph is fed into an *Encode-Process-Decode* sequence [133, 149, 10](Sec. 7.3). DefGraspNets predicts the stress and deformation $Y = (\vec{\sigma}, \vec{d})$ at steady state, at all vertices of M_o (Fig. 7.2). Please refer to Sec. 7.4 for the formal definitions of these fields.

[2]Although M_g comprises two unconnected parts, we refer to it collectively as the "gripper mesh."

Multigraph representation

The multigraph representation $G = (V, E^M, E^C)$ has nodes V and undirected edge sets E^M and E^C; each edge stores the indices of its two connected nodes. We list their features, and mark those that differ from [133] with a $\star$ bullet.

Nodes. The nodes V correspond to the vertices of M_o and M_g. Each node v_i has a feature vector consisting of

- A 3-element one-hot vector for node type (i.e., part of M_g, M_o surface, or M_o interior)
- The 3D Cartesian position of the node
- $\star$ A 3D unit vector in the gripper closing direction. This is nonzero only for gripper nodes and informs the network which direction the grippers are closing.

Mesh edges. The mesh edges E^M correspond to the edges of M_o and M_g. Each mesh edge e_{ij}^M connects nodes v_i and v_j of the same type. Its feature vector consists of

- The 3D Cartesian displacement vector from v_i to v_j
- The scalar Euclidean distance between v_i and v_j
- $\star$ The scalar elastic modulus E of the deformable object. This is nonzero only for edges belonging to the object.

Contact edges. The contact edges E^C connect object and gripper nodes and are computed based on proximity at initial contact. Each edge e_{ij}^C is formed between a pair of nodes v_i and v_j that have different node types and are closer than hyperparameter ϵ. The edge's feature vector comprises

- The 3D Cartesian displacement vector from v_i to v_j
- The scalar Euclidean distance between v_i and v_j
- $\star$ The normalized grasp force F_g^C, which is the total grasp force F_g divided by the number of contact edges $|E^C|$.

Encoder, processor, & decoder architectures

First, all feature vectors associated with the nodes V, mesh edges E^M, and contact edges E^C are encoded into a common latent space with 3 respective multilayer perceptrons (MLPs). Then, L message-passing blocks with 3 separate MLPs per block sequentially aggregate and process information from adjacent nodes and edges. Finally, a decoder MLP takes the processed nodal features in the latent space and jointly outputs the predicted stress and Cartesian displacement per node in real units (Pa and m). Full details of the Encode-Process-Decode sequence can be found in [133].

7.4 Data Generation and Model Training

We now describe our simulation-based approach to training DefGraspNets. We design a set of 60 object primitive models as a high-level abstraction of real-world geometries grouped into geometric categories (e.g., cuboids, cylinders, ellipsoids, annuli), and instances within each category have different dimensions and aspect ratios. Our dataset also includes a set of 11 of fruits and vegetables (e.g., apples, eggplants, potatoes) based on 3D scans [69]. Tetrahedral volume meshes are generated for each deformable object using fTetWild [55]. Triangular surface meshes are generated for the gripper fingers using Onshape.

For each pre-contacted object mesh M_o, 100 grasps are generated using an antipodal sampler [33] wherein randomly-sampled surface points define gripper contact points, surface normals define grasp axes, and 4 rotations are regularly drawn about each grasp axis. These 100 grasps correspond to 100 gripper meshes M_g. Each grasp is evaluated using the DefGraspSim[63] simulation framework (built upon Isaac Gym[111] and the FleX FEM solver[103]) with the Franka parallel-jaw gripper. DefGraspSim evaluates the stress and deformation fields of the deformable object during grasping.

Given an object-grasp pair (M_o, M_g) in DefGraspSim, the gripper applies a linearly increasing amount of force on the object until $F_g^{max} = 15$N is reached in a zero-gravity environment.[3] This force was achieved by directly commanding DOF torque applied at the gripper joints. The values of the stress ($\vec{\sigma}$) and deformation fields ($\vec{d}$) at all object vertices are saved over 50 evenly-spaced substeps throughout the entire grasping trajectory. Formally, our dataset D is composed of input-output pairs, each consisting of a candidate grasp pose X_i and corresponding set of fields Y_i, that is, $D = \{X_i = (M_g, M_o, F_g), Y_i = (\vec{\sigma}, \vec{d})\}_{i=1}^{N}$, where $0 \leq F_g \leq 15$. Dataset D has $N = \#\ objects \times 100 \times 50 = 3.55e5$ unique points. Because our network performs one-step predictions of the final state and is ideal for quasistatic interactions, all unstable grasps involving chaotic dynamics are not included in D.

The values of the stress field $\vec{\sigma}$ at all object vertices are computed as follows: first, the second-order stress tensor at each tetrahedral element of M_o is acquired from DefGraspSim. The stress tensors at each vertex are calculated by averaging the stress tensors at all adjacent elements. Each stress tensor is then converted to the scalar von Mises stress (i.e., the second invariant of the deviatoric stress), which is widely used to quantify whether a material has yielded [162]. The values of the deformation field $\vec{d}$ are defined simply as the distance between the positions of the pre-contacted vertices of M_o and their positions under gripper force F_g.

Contact edges E^C are formed based on the threshold $\epsilon = 5$mm. Our networks are trained with a decaying learning rate from $5e^{-5}$ to $1e^{-6}$ over 25 epochs and a batch size of 1. A latent size of 128 and $L = 15$ message passing steps are used, where all MLPs have 2 hidden layers. Loss is defined as the sum of the MSE of stress and deformation over all nodes. On a single RTX 3090 GPU, the network trains at approximately 1600 steps per minute.

[3] For the elastic moduli examined ($1e^4 \leq E \leq 1e^7$ Pa), 15N was observed to induce substantial stress and deformation; gravity was ignored due to having negligible effect on stress and deformation compared to contact forces.

7.5 Grasp Planning

We demonstrate DefGraspNets as a grasp planner, where both gradient-free (i.e., evaluation of sampled grasps) and gradient-based refinement methods can be used to find an optimal grasp. We define Q as the optimization objective, which is any backwards pass-differentiable measure of the predicted deformation and/or stress fields (e.g., mean deformation, smooth differentiable approximation of maximum stress implemented in modern deep learning libraries).

Evaluation of sampled grasps

First, DefGraspNets supports online sampling-based grasp planning. For an unseen object, forward passes of DefGraspNets can be used to evaluate Q for 100 random antipodal grasps with parallel batches of size 5 in 7.3 seconds. In comparison, DefGraspSim requires approximately 3 hours to evaluate 100 grasps, which is ∼1500x slower.

The best grasp pose is identified as $T^* = \arg\min_{T \in T_s} Q(T; M_o)$, where T is a 6D rigid transformation applied to a constant initial state of the gripper M_g^0 wherein both fingers are maximally open. Any valid M_g can be fully defined by T and joint states $\vec{p}_g \in R^2$ that determine how much each finger closes in order to contact M_o. These joint states $\vec{p}_g$ are calculated analytically by projecting the vertices of M_o onto the gripper faces, backprojecting the vertices within each face, and computing the minimum perpendicular distance over these vertices (i.e., the minimum contact distance) per finger.

Grasp refinement

Unlike existing deformable object planners, DefGraspNets' differentiability enables gradient-based refinement of a grasp pose to optimize Q. Starting from an initial grasp pose T_{init}, we perform gradient updates in the direction of $\partial Q \partial T$ to achieve a refined T using backtracking line search [130] and simulated annealing [80]. With 12 refinement steps per grasp, refining 100 initial grasps requires approximately 8 minutes. A comparable time does not exist for DefGraspSim, as it is not differentiable.

7.6 Prediction Results

We test DefGraspNets' predictions of the ranking of grasps with respect to their mean stress and deformation values by quantifying the respective Kendall's τ rank correlation coefficients (τ_s and τ_d).[4] We answer the following questions for 4 levels of generalization:

1. Can DefGraspNets rank unseen grasps when trained on other grasps on the same object? (Ans: Yes. For an 80-20 train-test split over grasps on the same object, we get an average $\tau_s = 0.78$ and $\tau_d = 0.66$ over 15000 unseen X_i.)

[4]Kendall's τ was chosen over Spearman's ρ for its comparative robustness (i.e., smaller gross error sensitivity).

2. Can DefGraspNets generalize to unseen elastic moduli E on the same object? (Ans: Yes. For a 7-3 train-test split over unique E for grasps on the same object, we get an average $\tau_s = 0.81$ and $\tau_d = 0.72$ over 15000 unseen X_i.)

3. Can DefGraspNets generalize to unseen primitive objects within the same geometric category? (Ans: Yes. For a 5-1 train-test split over unique objects, we get an average $\tau_s = 0.48$ and $\tau_d = 0.54$ over 15000 unseen X_i.)

4. Can DefGraspNets generalize to unseen real-world objects? (Ans: Yes. Moreover, we generate useful predictions even when training on a small number of objects, as long as the train geometries are relevant to the test geometry as quantified by a low Chamfer distance. See Table 7.1, which also reports the mean absolute error (MAE)).

Full visualizations of predicted field quantities for the 4th (i.e., most challenging) generalization level is shown in Fig. 7.3 on an unseen mustard bottle and unseen strawberry, as well as for the 2nd generalization level on a sphere.

Table 7.1: Generalization to unseen real-world objects. Gray cells denote the best values per column. Train sets each contain only 5 objects; the "All" group contains all 15. The d_C column measures the best Chamfer distance between the test geometry and the train geometries. Lower d_C implies geometric similarity between the train and test objects, and corresponds to more favorable MAE and τ during prediction.

Train set	**Mustard bottle**					**Lemon half**					**Strawberry**				
	d_C [mm] ↓	Deformation [mm]		Stress [kPa]		d_C ↓	Deformation		Stress		d_C ↓	Deformation		Stress	
		MAE ↓	τ_d ↑	MAE ↓	τ_s ↑		MAE ↓	τ_d ↑	MAE ↓	τ_s ↑		MAE ↓	τ_d ↑	MAE ↓	τ_s ↑
Group 1	5.57	0.71	0.62	2.92	0.56	3.57	4.82	0.20	2.15	0.31	3.27	0.42	0.09	6.41	0.58
Group 2	6.77	0.74	0.20	4.72	0.45	3.30	3.98	0.50	1.30	0.43	2.61	0.30	0.29	2.85	0.54
Group 3	6.07	0.73	-0.31	4.57	0.45	4.50	4.28	-0.03	2.63	0.19	2.46	0.31	-0.05	2.79	0.64
All	5.57	0.73	0.60	3.66	0.56	3.30	3.98	0.43	1.36	0.43	2.46	0.30	0.39	2.68	0.66

7.7 Grasp Planning Results

We demonstrate DefGraspNets as a grasp planner on 3 unseen objects (a mustard bottle, a lemon, and a strawberry) from existing datasets [17, 167] with real-world elastic moduli. First, we perform evaluation of sampled grasps. On each unseen object, 100 random grasps T_r are generated, and the optimization metric $Q(T)$ is evaluated for each $T \in T_r$ via the forward pass of DefGraspNets. Of the 100 grasps, we select the 10 grasps that are predicted to yield the lowest Q ("threshold low" grasps), as well as 10 grasps that are predicted to yield the highest Q ("threshold high" grasps). We also randomly select 10 other grasps from the remaining 80 grasp candidates as a baseline. These 30 grasps are then evaluated within the ground-truth simulator DefGraspSim.

DefGraspNets is a reliable predictor of minimal- and maximal-Q grasps on the unseen objects, with 88% of these threshold-low and high grasps belonging to the set of 30 lowest and highest ground-truth-Q grasps, respectively.

Subsequently, we perform gradient-based grasp refinement on the threshold-low and threshold-high grasps to further reduce and increase Q, respectively. For each object, box plots in Fig. 7.4 visualize the distribution of ground-truth Q values for 5 groups of grasps: all sampled grasps, threshold-low grasps, threshold-low grasps after refinement, threshold-high grasps, and threshold-high grasps after refinement. In all cases, not only do threshold-high and low grasps from DefGraspNets yield substantially different ground-truth Q values, but refinement increases their polarity as desired.

The highest- and lowest-Q grasps generated by the sample-and-refine grasp planning procedure are shown in Fig. 7.5. These grasps align with physical reasoning (e.g., the highest-deformation grasps on the bottle and lemon compress the directions of lowest geometric stiffness; the highest-stress grasp on a strawberry concentrates force on minimal area). These grasps are also validated in the real world in Fig. 7.6.

7.8 Ablation Studies

We run several ablation studies on our network architecture design. Table 7.2 lists key design variables in DefGraspNets, with our selected conditions in bold. We compare our baseline model with 5 other trained models, each of which differ from baseline by exactly one condition. We compare performance on a fixed test set and report the Kendall's τ metric for mean $\vec{d}$ and $\vec{\sigma}$. We address the following questions:

- Does jointly predicting stress and deformation outperform using two separate networks to predict these quantities? (Ans: The τ metric is comparable in both cases, likely because stress and deformation are coupled through the equations of elasticity. Thus, training two networks would be strictly disadvantageous computationally, c.f. V1.)
- Does one-step prediction outperform multi-step prediction? (Ans: Yes, when predicting deformation. Otherwise, both are comparable when predicting stress, c.f. V2. In MeshGraphNets, multi-step prediction does not accumulate significant deformation errors because the trajectory of the actuators is exactly controlled. In DefGraspNets, gripper force is commanded; the positions of *both* M_o and M_g are predicted and subject to accumulating errors.)
- Should F_g be normalized by the number of contact edges? (Ans: Yes. This aligns with simulation, in which the total force is the sum of forces at all contact points, c.f. V3.)
- Should force features F_g^C be assigned to contact edges or to gripper nodes? (Ans: The network is able to incorporate this information equally well, c.f. V4.)

7.9 Discussion

We present DefGraspNets, a differentiable GNN-based model for FEM simulation of 3D stress and deformation fields. We demonstrate that training DefGraspNets on a diverse set of grasps on primitive geometries enables effective prediction and grasp planning on unseen,

Table 7.2: Ablation study variables and conditions. Our DefGraspNets network conditions are in bold. Best conditions are in gray.

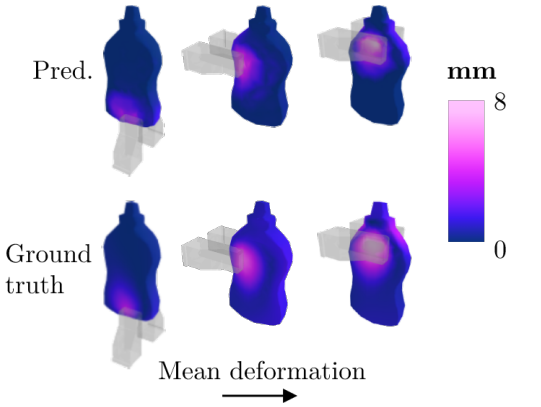
Variable	**Condition**	$\tau_d\uparrow$	$\tau_s\uparrow$
V1. Num. outputs	**Def. and stress**	0.61	0.82
	Def. only	0.57	
	Stress only		0.84
V2. Prediction type	**One-step predictions**	0.61	0.82
	Multi-step	0.37	0.70
V3. Value of F_g^W	**Distributed, $\mathbf{F_g}\vert\mathbf{E^W}\vert$**	0.61	0.82
	Non-distributed F_g	0.33	0.51
V4. Assignment of F_g	**On world edges E^W**	0.61	0.82
	On all nodes V	0.58	0.74

real-world geometries. DefGraspNets enables not only fast evaluation of sampled candidate grasps (1500x faster than GPU-accelerated FEM), but also gradient-based refinement of these grasps to optimize field quantities (e.g., max stress and mean deformation). We verify the effectiveness of optimized grasps on novel objects both in the ground-truth FEM simulator and in the real world.

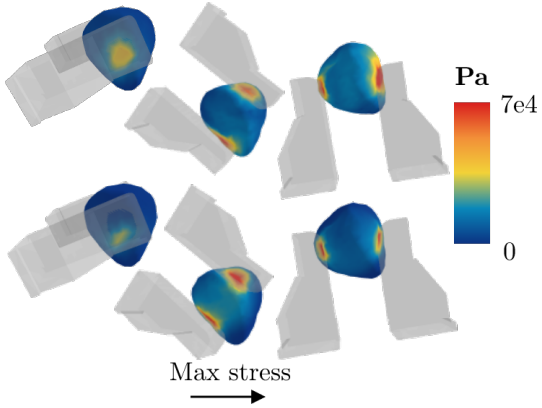
To expand DefGraspNets for use in downstream manipulation tasks such as food preparation or robotic surgery, prediction of additional quantities should be explored. These may include stability during transport and deformation and flow under reorientation and gravity. Furthermore, as FEM simulators evolve, DefGraspNets can be retrained to predict soft-soft contact or heterogeneous material responses.

Developing data augmentation techniques for *meshes* may enable vast dataset scaling from a minimal set of object models, further strengthening our ability to generalize to unseen objects. In addition, as our network is differentiable, techniques such as Stein variational gradient descent [96] and stochastic gradient Langevin dynamics [173] may allow us to provide probabilistic, multi-modal *distributions* of optimal grasps. Finally, architecture optimization (e.g., sparsity acceleration [19]) may lead to even faster performance.

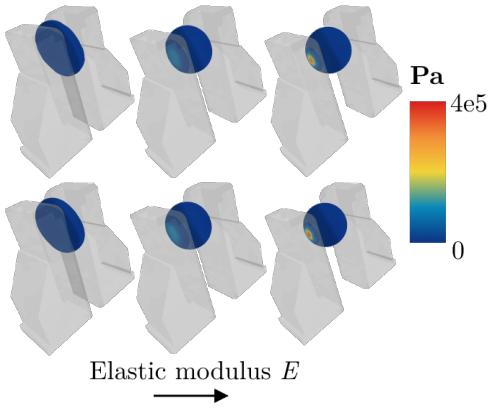
DefGraspNets contributes the first differentiable approach to deformable grasp planning capable of predicting and optimizing stress and deformation fields on novel objects. We believe this coupling of fast prediction of field quantities with a differentiable model will enable a wide range of users to apply deformable grasp planning to their target domains.



(a) Mustard bottle, $F_g = 12\text{N}$, $E = 1e7$



(b) Strawberry, $F_g = 6\text{N}$, $E = 5e4$



(c) Sphere, $F_g = 5\text{N}$, $E \in [5e5, 1e6, 5e6]$

Figure 7.3: A) Predicted and ground-truth deformation fields for a mustard bottle subject to grasps inducing increasing mean deformation, B) Predicted and ground-truth stress fields for a strawberry subject to grasps inducing increasing maximum stress, and C) Predicted and ground-truth stress fields for a sphere of increasing elastic moduli subject to identical grasps (deformation can be seen in resulting shape).

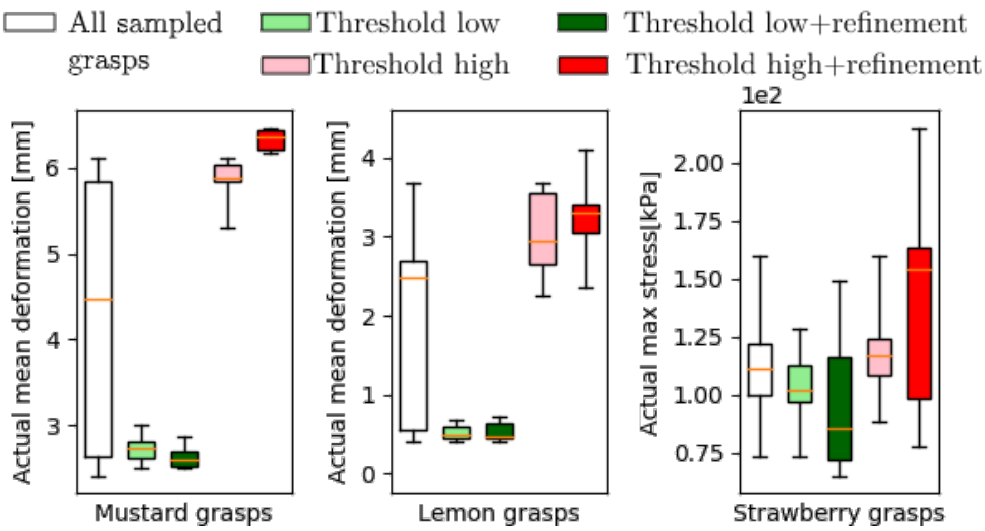


Figure 7.4: Box plots for 5 groups of grasps for each unseen object: 1) all grasps, 2) threshold low grasps from sampling only, 3) threshold low grasps after refinement, 4) threshold high grasps from sampling only, and 5) threshold high grasps after refinement. The y-axis is the ground-truth Q value of these grasps as computed in DefGraspSim. ©2023 IEEE.

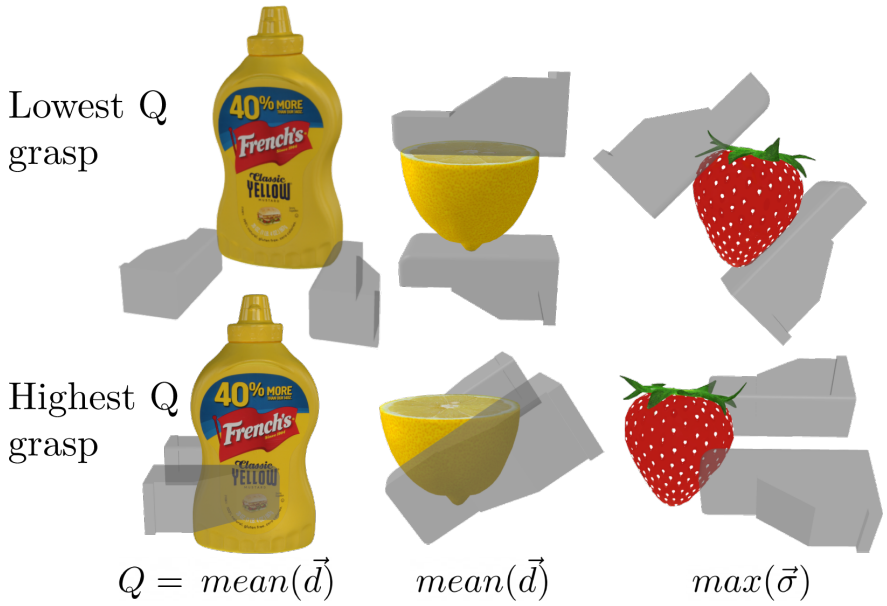


Figure 7.5: Highest- and lowest-Q grasps for the mustard bottle, lemon, and strawberry generated by the sample-and-refine procedure. ©2023 IEEE.

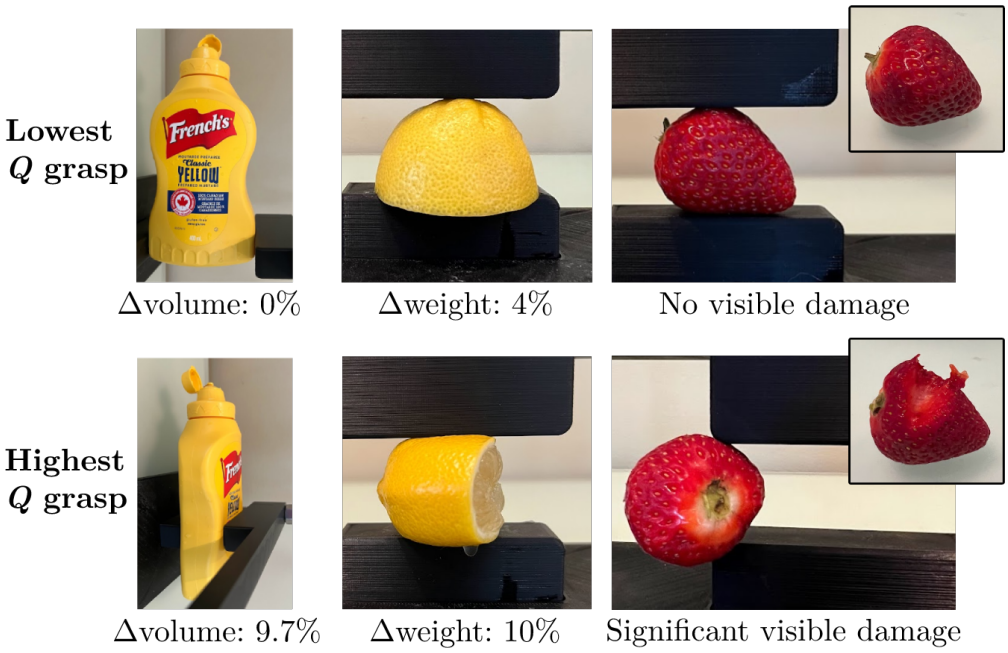


Figure 7.6: Validation of grasps from Fig. 7.5 using a Franka-based gripper gravitationally loaded under 15N. For the bottle and lemon, deformation is measured by proxy (change in volume and weight). For the strawberry, only the highest-Q grasp imparts damage. ©2023 IEEE.

Bibliography

[1] P. Kormushev et al. "Imitation Learning of Positional and Force Skills Demonstrated via Kinesthetic Teaching and Haptic Input". In: *Advanced Robotics* 25.5 (2011), pp. 581–603.

[2] B.D. Argall et al. "A Survey of Robot Learning from Demonstration". In: *Robot. Auton. Syst.* 57.5 (2009), pp. 469–483.

[3] Brenna D. Argall and Aude G. Billard. "A survey of Tactile Human-Robot Interactions". In: *Robotics and Autonomous Systems* 58.10 (2010), pp. 1159–1176. ISSN: 0921-8890.

[4] V. E. Arriola-Rios et al. "Modeling of Deformable Objects for Robotic Manipulation: A Tutorial and Review". In: *Frontiers in Robotics and AI* (2020).

[5] C. Chen B. McInroe. "Towards a Soft Fingertip with Integrated Sensing and Actuation". In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2018.

[6] Andrea Bajcsy et al. "Learning robot objectives from physical human interaction". In: Nov. 2017.

[7] P. Bakker and Y. Kuniyoshi. "Robot See, Robot Do: An Overview of Robot Imitation". In: *AISB96 Workshop on Learning in Robots and Animals*. 1996, pp. 3–11.

[8] R. Balasubramanian et al. "Human-guided grasp measures improve grasp robustness on physical robot". In: *IEEE Intl. Conf. on Robotics and Automation*. 2010.

[9] Jernej Barbič, Fun Shing Sin, and Daniel Schroeder. *Vega FEM Library*. http://www.jernejbarbic.com 2012.

[10] Peter Battaglia et al. "Relational inductive biases, deep learning, and graph networks". In: *arXiv preprint arXiv:1806.01261* (2018). URL: `https://arxiv.org/pdf/1806.01261.pdf`.

[11] Atilim Gunes Baydin et al. "Automatic differentiation in machine learning: A survey". In: *Journal of Machine Learning Research* (2018).

[12] P. Beeson and B. Ames. "TRAC-IK: An open-source library for improved solving of generic inverse kinematics". In: *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*. Nov. 2015, pp. 928–935.

[13] Antonio Bicchi. "On the Closure Properties of Robotic Grasping". In: *Intl. Journal of Robotics Research* (1995).

[14] S. Calinon, F. Guenter, and A. Billard. "On Learning, Representing, and Generalizing a Task in a Humanoid Robot". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part B* 37 (2007), pp. 286–298.

[15] S. Calinon et al. "Learning and Reproduction of Gestures by Imitation". In: *IEEE Robotics Automation Magazine* 17.2 (2010), pp. 44–54.

[16] B. Calli et al. "Benchmarking in Manipulation Research: Using the Yale-CMU-Berkeley Object and Model Set". In: *IEEE Robotics & Automation Magazine* (2015).

[17] Berk Calli et al. "The YCB object and Model set: Towards common benchmarks for manipulation research". In: *2015 International Conference on Advanced Robotics (ICAR)*. 2015, pp. 510–517. DOI: 10.1109/ICAR.2015.7251504.

[18] C. Chorley et al. "Development of a tactile sensor based on biologically inspired edge encoding". In: *2009 International Conference on Advanced Robotics*. 2009, pp. 1–6.

[19] Christopher Choy, JunYoung Gwak, and Silvio Savarese. "4D Spatio-Temporal ConvNets: Minkowski Convolutional Neural Networks". In: *IEEE Conf. on Computer Vision and Pattern Recognition*. 2019.

[20] M. Cianchetti et al. "Soft Robotics Technologies to Address Shortcomings in Today's Minimally Invasive Surgery: The STIFF-FLOP Approach". In: *Soft Robotics* 1.2 (2014), pp. 122–131.

[21] M. Ciocarlie, Corey Goldfeder, and P. Allen. "Dexterous Grasping via Eigengrasps: A Low-dimensional Approach to a High-complexity Problem". In: *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*. 2007.

[22] A. Clegg et al. "Learning to Collaborate From Simulation for Robot-Assisted Dressing". In: *IEEE Robotics and Automation Letters* (2020).

[23] E. Coumans and Y. Bai. *PyBullet: A Python module for physics simulation for games, robotics and machine learning*. http://pybullet.org. 2016–2019.

[24] R. S. Dahiya et al. "Tactile Sensing-From Humans to Humanoids". In: *IEEE Transactions on Robotics* 26.1 (2010), pp. 1–20.

[25] Hao Dang and Peter K. Allen. "Learning grasp stability". In: *IEEE Intl. Conf. on Robotics and Automation*. 2012.

[26] J. Dargahi. "A three sensing element piezoelectric tactile sensor for robotic and prosthetic applications". In: *Sensors and Actuators A-physical* 80 (2000), pp. 23–30.

[27] D. Ding, Y.H. Lee, and S. Wang. "Computation of 3-D form-closure grasps". In: *IEEE Transactions on Robotics and Automation* (2001).

[28] Athanasios C. Dometios et al. "Vision-Based Online Adaptation of Motion Primitives to Dynamic Surfaces: Application to an Interactive Robotic Wiping Task". In: *IEEE Robotics and Automation Letters* 3.3 (2018), pp. 1410–1417. DOI: 10.1109/LRA.2018.2800031.

[29] Tao Du et al. "DiffPD: Differentiable projective dynamics". In: *ACM Transactions on Graphics (TOG)* (2021).

[30] C. Duriez. "Control of elastic soft robots based on real-time finite element method". In: *IEEE Intl. Conf. on Robotics and Automation*. 2013.

[31] M. Edmonds et al. "Feeling the Force: Integrating Force and Pose for Fluent Discovery through Imitation Learning to Open Medicine Bottles". In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2017.

[32] Jonathan Engel, Jack Chen, and Chang Liu. "Development of polyimide flexible tactile sensor skin". In: *J. Micromech. Microeng* 13 (May 2003), pp. 359–366.

[33] Clemens Eppner, Arsalan Mousavian, and Dieter Fox. "A Billion Ways to Grasp: An Evaluation of Grasp Sampling Schemes on a Dense, Physics-based Grasp Data Set". In: *Int. Symp. on Robotics Research*. 2019.

[34] Zackory Erickson et al. "Multidimensional Capacitive Sensing for Robot-Assisted Dressing and Bathing". In: *2019 IEEE 16th International Conference on Rehabilitation Robotics (ICORR)*. 2019, pp. 224–231.

[35] R.S. Fearing. "Tactile Sensing Mechanisms". In: *The International Journal of Robotics Research* 9.3 (1990), pp. 3–23.

[36] T. Feix et al. "The GRASP Taxonomy of Human Grasp Types". In: *IEEE Transactions on Human-Machine Systems* (2016).

[37] C. Ferrari and J. Canny. "Planning optimal grasps". In: *IEEE Intl. Conf. on Robotics and Automation*. 1992.

[38] N. J. Ferrier and R. W. Brockett. "Reconstructing the Shape of a Deformable Membrane from Image Data". In: *The International Journal of Robotics Research* 19.9 (2000), pp. 795–816.

[39] C. Daniel Freeman et al. *Brax: A Differentiable Physics Engine for Large Scale Rigid Body Simulation*. `http://github.com/google/brax`. Version 0.0.15. 2021.

[40] C Gabrel. "Characteristics of elderly nursing home current residents and discharges: data from the 1997 National Nursing Home Survey". In: *Advance data* 312 (May 2000), pp. 1–15.

[41] Wei Gao and Russ Tedrake. "kPAM 2.0: Feedback Control for Category-Level Robotic Manipulation". In: *IEEE Robotics and Automation Letters* 6.2 (2021), pp. 2962–2969. DOI: `10.1109/LRA.2021.3062315`.

[42] Moritz Geilinger et al. "ADD: Analytically differentiable dynamics for multi-body systems with frictional contact". In: *ACM Transactions on Graphics (TOG)* (2020).

[43] M. C. Gemici and A. Saxena. "Learning haptic representation for manipulating deformable food objects". In: *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*. 2014.

[44] K. Gopalakrishnan and K. Goldberg. "D-space and Deform Closure Grasps of Deformable Parts". In: *Intl. Journal of Robotics Research* (2005).

[45] D. Grollman and O. Jenkins. "Can We Learn Finite State Machine Robot Controllers from Interactive Demonstration?" In: *From Motor Learning to Interaction Learning in Robots*. Vol. 264. 2010, pp. 407–430.

[46] Roderic A Grupen. "Planning Grasp Strategies for Multifingered Robot Hands". In: *IEEE Intl. Conf. on Robotics and Automation*. 1991.

[47] A. Gupta et al. "Learning dexterous manipulation for a soft robotic hand from human demonstrations". In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Oct. 2016, pp. 3786–3793.

[48] Toney-Butler TJ Haddad LM Annamaraju P. *Nursing Shortage*. https://www.ncbi.nlm.nih.gov/books/NBK493175/. 2021.

[49] Eric Heiden et al. "DiSECt: A Differentiable Simulator for Parameter Inference and Control in Robotic Cutting". In: *Autonomous Robots* (2022).

[50] Eric Heiden et al. "NeuralSim: Augmenting Differentiable Simulators with Neural Networks". In: *IEEE Intl. Conf. on Robotics and Automation*. 2021. URL: https://github.com/google-research/tiny-differentiable-simulator.

[51] Christoph Hennersperger et al. "Towards MRI-based autonomous robotic US acquisitions: a first feasibility study". In: *IEEE transactions on medical imaging* 36.2 (2017), pp. 538–548.

[52] Peter C Horak and Jeff C Trinkle. "On the similarities and differences among contact models in robot simulation". In: *IEEE Robotics and Automation Letters* (2019).

[53] A.M. Howard and G.A. Bekey. "Intelligent Learning for Deformable Object Manipulation". In: *Autonomous Robots* (2000).

[54] D. Hristu, N. Ferrier, and R. W. Brockett. "The performance of a deformable-membrane tactile sensor: basic results on geometrically-defined tasks". In: *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)*. Vol. 1. 2000, 508–513 vol.1.

[55] Yixin Hu et al. "Fast Tetrahedral Meshing in the Wild". In: *ACM Trans. on Graphics* (2020).

[56] Yuanming Hu et al. "ChainQueen: A real-time differentiable physical simulator for soft robotics". In: *IEEE Intl. Conf. on Robotics and Automation*. 2019.

[57] Yuanming Hu et al. "DiffTaichi: Differentiable programming for physical simulation". In: 2020.

[58] Isabella Huang and Ruzena Bajcsy. "High Resolution Soft Tactile Interface for Physical Human-Robot Interaction". In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. 2020, pp. 1705–1711. DOI: 10.1109/ICRA40945.2020.9197365.

[59] Isabella Huang and Ruzena Bajcsy. "Robot Learning from Demonstration with Tactile Signals for Geometry-Dependent Tasks". In: *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2020, pp. 8323–8328. DOI: 10.1109/IROS45743.2020.9340818.

[60] Isabella Huang, Dylan Chow, and Ruzena Bajcsy. "Soft Tactile Contour Following for Robot-Assisted Wiping and Bathing". In: *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2022.

[61] Isabella Huang, Jingjun Liu, and Ruzena Bajcsy. "A Depth Camera-Based Soft Fingertip Device for Contact Region Estimation and Perception-Action Coupling". In: *2019 International Conference on Robotics and Automation (ICRA)*. 2019, pp. 8443–8449. DOI: 10.1109/ICRA.2019.8793612.

[62] Isabella Huang et al. "DefGraspNets: Grasp Planning on 3D Fields with Graph Neural Nets". In: *2023 IEEE International Conference on Robotics and Automation (ICRA)*. 2023.

[63] Isabella Huang et al. "DefGraspSim: Physics-Based Simulation of Grasp Outcomes for 3D Deformable Objects". In: *IEEE Robotics and Automation Letters* (2022).

[64] Zhiao Huang et al. "PlasticineLab: A soft-body manipulation benchmark with differentiable physics". In: 2021.

[65] R. Bajcsy I. Huang J. Liu. "A Depth Camera-Based Soft Fingertip Device for Contact RegionEstimation and Perception-Action Coupling". In: *2019 International Conference on Robotics and Automation (ICRA)*. 2019.

[66] F. Iida and C. Laschi. "Soft Robotics: Challenges and Perspectives". In: *FET*. 2011.

[67] A.J. Ijspeert et al. "Dynamical Movement Primitives: Learning Attractor Models for Motor Behaviors". In: *Neural Computation* 25.2 (2013), pp. 328–373.

[68] Y. Ito et al. "Contact State Estimation by Vision-Based Tactile Sensors for Dexterous Manipulation with Robot Hands Based on Shape-Sensing". In: *International Journal of Advanced Robotic Systems* 8.4 (2011), p. 54.

[69] P. Jamdagni and Y.B. Jia. *Real Food Dataset*. 2021.

[70] Krishna Murthy Jatavallabhula et al. "gradSim: Differentiable simulation for system identification and visuomotor control". In: 2020.

[71] Y.B. Jia, F. Guo, and H. Lin. "Grasping deformable planar objects: Squeeze, stick/slip analysis, and energy-based optimalities". In: *Intl. Journal of Robotics Research* (2014).

[72] George Em Karniadakis et al. "Physics-informed machine learning". In: *Nature Reviews Physics* (2021).

[73] Chung Min Kim et al. "Simulation of Parallel-Jaw Grasping using Incremental Potential Contact Models". In: *IEEE Intl. Conf. on Robotics and Automation*. 2022.

[74] Joohyung Kim, Alexander Alspach, and Katsu Yamane. "3D printed soft skin for safe human-robot interaction". In: Sept. 2015, pp. 2419–2425.

[75] Sangbae Kim, Cecilia Laschi, and Barry Trimmer. "Soft robotics: a bioinspired evolution in robotics". In: *Trends in Biotechnology* 31.5 (2013), pp. 287–294. ISSN: 0167-7799.

[76] Chih-Hung King et al. "Towards an assistive robot that autonomously performs bed baths for patient hygiene". In: *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2010, pp. 319–324. DOI: 10.1109/IROS.2010.5649101.

[77] Marek Kopicki et al. "One-Shot Learning and Generation of Dexterous Grasps for Novel Objects". In: *Intl. Journal of Robotics Research* (2016).

[78] K. Kosuge et al. "Dance partner robot - Ms DanceR". In: *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003) (Cat. No.03CH37453)*. Vol. 4. Oct. 2003, 3459–3464 vol.3.

[79] D. Kulić et al. "Incremental learning of full body motion primitives and their sequencing through human motion observation". In: *The International Journal of Robotics Research* 31.3 (2012), pp. 330–345.

[80] P. J. M. Laarhoven and E. H. L. Aarts. *Simulated Annealing: Theory and Applications*. USA: Kluwer Academic Publishers, 1987. ISBN: 9027725136.

[81] Nicolas Lauzier. *Robot force control: An introduction*. `https://blog.robotiq.com/bid/53553/Robot-Force-Control-An-Introduction`. 2016.

[82] Z. Lazher et al. "Modeling and analysis of 3D deformable object grasping". In: *Intl. Conf. on Robotics in Alpe-Adria-Danube Region*. 2014.

[83] Tran Nguyen Le et al. "A Novel Simulation-Based Quality Metric for Evaluating Grasps on 3D Deformable Objects". In: *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*. 2023.

[84] Mark H. Lee. "Tactile Sensing: New Directions, New Challenges". In: *The International Journal of Robotics Research* 19.7 (2000), pp. 636–643. DOI: `10.1177/027836490001900702`.

[85] Ian Lenz, Honglak Lee, and Ashutosh Saxena. "Deep Learning for Detecting Robotic Grasps". In: *Intl. Journal of Robotics Research* (2013).

[86] Ian Lenz, Honglak Lee, and Ashutosh Saxena. "Deep Learning for Detecting Robotic Grasps". In: *Intl. Journal of Robotics Research* 34.4-5 (2015), pp. 705–724.

[87] Beatriz León et al. "OpenGRASP: A toolkit for robot grasping simulation". In: *SIMPAR*. 2010.

[88] R. Li et al. "Localization and manipulation of small parts using GelSight tactile sensing". In: *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2014, pp. 3988–3993.

[89] R. Li et al. "Localization and manipulation of small parts using GelSight tactile sensing". In: *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2014, pp. 3988–3993.

[90] Rui Li and Edward H. Adelson. "Sensing and Recognizing Surface Textures Using a GelSight Sensor". In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2013.

[91] Y. Li et al. "Folding deformable objects using predictive simulation and trajectory optimization". In: *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*. 2015.

[92] Yunzhu Li et al. "Learning particle dynamics for manipulating rigid bodies, deformable objects, and fluids". In: (2019).

[93] Z. Li and S. S. Sastry. "Task-oriented optimal grasping by multifingered robot hands". In: *IEEE Journal on Robotics and Automation* (1988).

[94] Timothy P Lillicrap and Adam Santoro. "Backpropagation through time and the brain". In: *Current Opinion in Neurobiology* 55 (2019), pp. 82–89.

[95] H. Lin et al. "Picking up a soft 3D object by "feeling" the grip". In: *Intl. Journal of Robotics Research* (2015).

[96] Qiang Liu and Dilin Wang. "Stein Variational Gradient Descent: A General Purpose Bayesian Inference Algorithm". In: *Advances in Neural Information Processing Systems*. Ed. by D. Lee et al. Vol. 29. Curran Associates, Inc., 2016.

[97] N. Lu and D.H. Kim. "Flexible and Stretchable Electronics Paving the Way for Soft Robotics". In: *Soft Robotics* 1.1 (2014), pp. 53–62.

[98] Q. Lu et al. "Multi-Fingered Grasp Planning via Inference in Deep Neural Networks". In: *IEEE Robotics & Automation Magazine* (2020).

[99] Qingkai Lu et al. "Multi-Fingered Grasp Planning via Inference in Deep Neural Networks". In: *IEEE Robotics & Automation Magazine* (2020).

[100] Jens Lundell et al. "Multi-FinGAN: Generative Coarse-To-Fine Sampling of Multi-Finger Grasps". In: *IEEE Intl. Conf. on Robotics and Automation*. 2021.

[101] Miles Macklin. *Warp: A High-performance Python Framework for GPU Simulation and Graphics*. `https://github.com/nvidia/warp`. NVIDIA GPU Technology Conference (GTC). Mar. 2022.

[102] Miles Macklin et al. "Non-Smooth Newton Methods for Deformable Multi-Body Dynamics". In: *ACM Trans. on Graphics* (2019).

[103] Miles Macklin et al. "Non-smooth Newton methods for deformable multi-body dynamics". In: *ACM Transactions on Graphics (TOG)* (2019).

[104] J. Mahler et al. "Dex-Net 2.0: Deep Learning to Plan Robust Grasps with Synthetic Point Clouds and Analytic Grasp Metrics". In: *Robotics Science and Systems*. 2017.

[105] Jeffrey Mahler et al. "Dex-Net 2.0: Deep Learning to Plan Robust Grasps with Synthetic Point Clouds and Analytic Grasp Metrics". In: *Robotics Science and Systems*. 2017.

[106] J. Maitin-Shepard et al. "Cloth grasp point detection based on multiple-view geometric cues with application to robotic towel folding". In: *IEEE Intl. Conf. on Robotics and Automation*. 2010.

[107] C. Majidi. "Soft Robotics: A Perspective-Current Trends and Prospects for the Future". In: *Soft Robotics* 1.1 (2014), pp. 5–11.

[108] Carmel Majidi. "Soft Robotics: A Perspective—Current Trends and Prospects for the Future". In: *Soft Robotics* (2014).

[109] MakerBot. *Thingiverse: Digital Designs for Physical Objects*. `https://www.thingiverse.com`. 2020.

[110] Koshi Makihara et al. "Grasp pose detection for deformable daily items by pix2stiffness estimation". In: *Advanced Robotics* (2022).

[111] Viktor Makoviychuk et al. "Isaac Gym: High Performance GPU-Based Physics Simulation For Robot Learning". In: *CoRR* (2021). URL: https://arxiv.org/abs/2108.10470.

[112] Charles C Margossian. "A review of automatic differentiation and its efficient implementation". In: *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* (2019).

[113] Matthew T Mason. *Mechanics of Robotic Manipulation*. MIT press, 2001.

[114] A. T. Miller and P. K. Allen. "GraspIt!: A versatile simulator for robotic grasping". In: *IEEE Robotics & Automation Magazine* (2004).

[115] A. T. Miller et al. "Automatic grasp planning using shape primitives". In: *IEEE Intl. Conf. on Robotics and Automation*. 2003.

[116] B. Mirtich and J. Canny. "Easily computable optimum grasps in 2D and 3D". In: *IEEE Intl. Conf. on Robotics and Automation*. 1994.

[117] Tracy Mitzner et al. "Identifying the Potential for Robotics to Assist Older Adults in Different Living Environments". In: *International Journal of Social Robotics* 6 (Apr. 2014), pp. 213–227. DOI: 10.1007/s12369-013-0218-7.

[118] Y. Mohammad and Y. Nishida. "Tackling the Correspondence Problem". In: *Active Media Technology*. Cham: Springer International Publishing, 2013, pp. 84–95.

[119] B. Mosadegh et al. "Pneumatic Networks for Soft Robotics that Actuate Rapidly". In: *Advanced Functional Materials* 24.15 (2014), pp. 2163–2170.

[120] Arsalan Mousavian, Clemens Eppner, and Dieter Fox. "6-DOF GraspNet: Variational Grasp Generation for Object Manipulation". In: *IEEE Conf. on Computer Vision and Pattern Recognition*. 2019.

[121] Arsalan Mousavian, Clemens Eppner, and Dieter Fox. "6-DOF GraspNet: Variational Grasp Generation for Object Manipulation". In: *IEEE Conf. on Computer Vision and Pattern Recognition*. 2019.

[122] T. Mukai et al. "Development of the Tactile Sensor System of a Human-Interactive Robot "RI-MAN"". In: *IEEE Transactions on Robotics* 24.2 (Apr. 2008), pp. 505–512.

[123] Richard M Murray, Zexiang Li, and S Shankar Sastry. *A Mathematical Introduction to Robotic Manipulation*. CRC Press, 1994, p. 480.

[124] Fusaomi Nagata et al. "CAD/CAM-based position/force controller for a mold polishing robot". In: *Mechatronics* 17.4 (2007), pp. 207–216.

[125] Yashraj Narang et al. "Interpreting and Predicting Tactile Signals via a Physics-Based and Data-Driven Framework". In: *Robotics Science and Systems*. 2020.

[126] Yashraj Narang et al. "Sim-to-Real for Robotic Tactile Sensing via Physics-Based Simulation and Learned Latent Projections". In: *IEEE International Conference on Robotics and Automation*. 2021.

[127] Yashraj S Narang et al. "Interpreting and predicting tactile signals for the SynTouch BioTac". In: *Intl. Journal of Robotics Research* (2021).

[128] E. R. Nascimento et al. "BRAND: A robust appearance and depth descriptor for RGB-D images". In: *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 2012, pp. 1720–1726.

[129] Rhys Newbury et al. *Deep Learning Approaches to Grasp Synthesis: A Review*. 2022.

[130] Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. USA: Springer, 2006.

[131] OnRobot. *The power and importance of sensing technologies*. `https://onrobot.com/en/blog/the-power-and-importance-of-sensing-technologies`. 2019.

[132] Z. Pan, X. Gao, and D. Manocha. "Grasping Fragile Objects Using A Stress-Minimization Metric". In: *IEEE Intl. Conf. on Robotics and Automation*. 2020.

[133] Tobias Pfaff et al. "Learning Mesh-Based Simulation with Graph Networks". In: 2021.

[134] Samuel Pfrommer, Mathew Halm, and Michael Posa. "ContactNets: Learning discontinuous contact dynamics with smooth, implicit representations". In: *Conference on Robot Learning*. 2020.

[135] P. Polygerinos et al. "Soft Robotics: Review of Fluid-Driven Intrinsically Soft Devices; Manufacturing, Sensing, Control, and Applications in Human-Robot Interaction". In: *Advanced Engineering Materials* 19.12 (2017), p. 1700016.

[136] M. Pozzi et al. "Efficient FEM-Based Simulation of Soft Robots Modeled as Kinematic Chains". In: *IEEE Intl. Conf. on Robotics and Automation*. 2018.

[137] Yiling Qiao et al. "Differentiable simulation of soft multi-body systems". In: *Advances in Neural Information Processing Systems*. 2021.

[138] Mohammad Rahman et al. "Development and Control of a Robotic Exoskeleton for Shoulder, Elbow and Forearm Movement Assistance". In: *Applied Bionics and Biomechanics* 9 (July 2012), pp. 275–292.

[139] Maziar Raissi, Paris Perdikaris, and George E Karniadakis. "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations". In: *Journal of Computational physics* (2019).

[140] J N Reddy. *Introduction to the Finite Element Method*. McGraw Hill, 2019.

[141] M.A. Roa and R. Suarez. "Grasp Quality Measures: Review and Performance". In: *Autonomous Robots* (2014).

[142] L. et al Rozo. "A Robot Learning from Demonstration Framework to Perform Force-based Manipulation Tasks". In: *Journal of Intelligent Service Robotics, Special Issue on AI Techniques for Robotics* 6 (2013), pp. 33–51.

[143] C. Rubert et al. "On the relevance of grasp metrics for predicting grasp success". In: *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*. 2017.

[144] R. B. Rusu and S. Cousins. "3D is here: Point Cloud Library (PCL)". In: *2011 IEEE International Conference on Robotics and Automation*. 2011, pp. 1–4.

[145] S. Saga, H. Kajimoto, and S. Tachi. "High-resolution tactile sensor using the deformation of a reflection image". In: *Sensor Review* 27.1 (2007), pp. 35–42.

[146] Anis Sahbani, Sahar El-Khoury, and Philippe Bidaud. "An Overview of 3D Object Grasp Synthesis Algorithms". In: *Robotics and Autonomous Systems* (2012).

[147] Namiko Saito et al. "Utilization of Image/Force/Tactile Sensor Data for Object-Shape-Oriented Manipulation: Wiping Objects With Turning Back Motions and Occlusion". In: *IEEE Robotics and Automation Letters* 7.2 (2022), pp. 968–975. DOI: 10.1109/LRA.2021.3136657.

[148] J. Sanchez et al. "Robotic Manipulation and Sensing of Deformable Objects in Domestic and Industrial Applications". In: *Intl. Journal of Robotics Research* (2018).

[149] Alvaro Sanchez-Gonzalez et al. "Learning to simulate complex physics with graph networks". In: *International Conference on Machine Learning*. 2020.

[150] Schunk. *Parallel gripper: From micro assembly to heavy-load handling*. https://schunk.com/ca_en/gripping-systems/category/gripping-systems/schunk-grippers/parallel-gripper/. 2021.

[151] M. Sellier. "An iterative method for the inverse elasto-static problem". In: *Journal of Fluids and Structures* 27.8 (2011), pp. 1461–1470.

[152] Haochen Shi et al. "RoboCraft: Learning to See, Simulate, and Shape Elasto-Plastic Objects with Graph Networks". In: *Robotics Science and Systems*. 2022.

[153] J. Smolen and A. Patriciu. "Deformation Planning for Robotic Soft Tissue Manipulation". In: *Intl. Conf. on Advances in Computer-Human Interactions*. 2009.

[154] Peng Song, Juan Antonio Corrales Ramón, and Youcef Mezouar. "Dynamic Evaluation of Deformable Object Grasping". In: *IEEE Robotics and Automation Letters* (2022).

[155] Olga Sorkine-Hornung and Michael Rabinovich. *Least-Squares Rigid Motion Using SVD*. https://igl.ethz.ch/projects/ARAP/svd_rot.pdf. 2017.

[156] Thomas H. Speeter. "Three-dimensional Finite Element Analysis of Elastic Continua for Tactile Sensing". In: *The International Journal of Robotics Research* 11.1 (1992), pp. 1–19.

[157] David E Stewart. "Rigid-body dynamics with friction and impact". In: *SIAM Review* (2000).

[158] W.D. Stiehl et al. "The design of the huggable: A therapeutic robotic companion for relational, affective touch". In: vol. 2005. Sept. 2005, pp. 408–415. ISBN: 0-7803-9274-4.

[159] H.G. Sung. "Gaussian Mixture Regression and Classification". PhD thesis. Rice University, 2004.

[160] JaYoung Sung, Rebecca Grinter, and Henrik Christensen. "Domestic Robot Ecology". In: *International Journal of Social Robotics* 2 (Dec. 2010), pp. 417–429. DOI: 10.1007/s12369-010-0065-8.

[161] S. Takenawa. "A soft three-axis tactile sensor based on electromagnetic induction". In: *2009 IEEE International Conference on Mechatronics*. 2009, pp. 1–6.

[162] S. Timoshenko and J.N. Goodier. *Theory Of Elasticity*. McGraw-Hill Education, 2010. URL: https://books.google.com/books?id=8FRHzwEACAAJ.

[163] Kyoko Toda et al. "Seed Protein Content and Consistency of Tofu Prepared with Different Magnesium Chloride Concentrations in Six Japanese Soybean Varieties". In: *Breeding Science* (2003).

[164] E. Todorov, T. Erez, and Y. Tassa. "MuJoCo: A physics engine for model-based control". In: *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*. 2012.

[165] Giovanni Tonietti, R. Schiavi, and Antonio Bicchi. "Design and Control of a Variable Stiffness Actuator for Safe and Fast Physical Human/Robot Interaction". In: May 2005, pp. 526–531. ISBN: 0-7803-8914-X.

[166] Deepak Trivedi et al. "Soft Robotics: Biological Inspiration, State of the Art, and Future Research". In: *Appl. Bionics Biomechanics* 5.3 (2008), pp. 99–117. ISSN: 1176-2322.

[167] TurboSquid. *3D Models for Professionals*. https://www.turbosquid.com. 2023.

[168] Benjamin Ummenhofer et al. "Lagrangian Fluid Simulation with Continuous Convolutions". In: 2020.

[169] D. Vella et al. "The indentation of pressurized elastic shells: from polymeric capsules to yeast cells." In: *Journal of the Royal Society, Interface* 9 68 (2012), pp. 448–55.

[170] Francisco E. Viña B. et al. "Adaptive control for pivoting with visual and tactile feedback". In: *IEEE Intl. Conf. on Robotics and Automation*. 2016.

[171] C. Vogel. *Computational Methods for Inverse Problems*. Society for Industrial and Applied Mathematics, 2002.

[172] W. Wang, D. Berenson, and D. Balkcom. "An online method for tight-tolerance insertion tasks for string and rope". In: *IEEE Intl. Conf. on Robotics and Automation*. 2015.

[173] Max Welling and Yee Whye Teh. In: *Proceedings of the International Conference on Machine Learning*. Ed. by Lise Getoor and Tobias Scheffer. Omnipress, 2011, pp. 681–688.

[174] Keenon Werling et al. "Fast and feature-complete differentiable physics for articulated rigid bodies with contact". In: *Robotics Science and Systems*. 2021.

[175] Zhirong Wu et al. "3D ShapeNets: A deep representation for volumetric shapes". In: *IEEE Conf. on Computer Vision and Pattern Recognition*. 2015.

[176] J. Xu et al. "Minimal Work: A Grasp Quality Metric for Deformable Hollow Objects". In: *IEEE Intl. Conf. on Robotics and Automation*. 2020.

[177] Jie Xu et al. "Accelerated policy learning with parallel differentiable simulation". In: 2022.

[178] Jie Xu et al. "An end-to-end differentiable framework for contact-aware robot design". In: *Robotics Science and Systems*. 2021.

[179] A. Yamaguchi and C. G. Atkeson. "Implementing tactile behaviors using FingerVision". In: *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*. 2017, pp. 241–248.

[180] H. Yin, A. Varava, and D. Kragic. "Modeling, learning, perception, and control methods for deformable object manipulation". In: *Science Robotics* (2021).

[181] W. Yuan et al. "Shape-independent hardness estimation using deep learning and a GelSight tactile sensor". In: *2017 IEEE International Conference on Robotics and Automation (ICRA)*. May 2017, pp. 951–958.

[182] X. Zhou and J. Lu. "Inverse formulation for geometrically exact stress resultant shells". In: *International Journal for Numerical Methods in Engineering* 74.8 (2007), pp. 1278–1302.

[183] Jihong Zhu et al. "Challenges and Outlook in Robotic Manipulation of Deformable Objects". In: *IEEE Robotics & Automation Magazine* (2022).