Self supervised tactile perception for robot dexterity

Akash Sharma

CMU-RI-TR-26-05

Jan 30, 2026

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

Thesis Committee:
Michael Kaess, *chair*
Shubham Tulsiani
Guanya Shi
Mustafa Mukadam, *Amazon Robotics*
Jitendra Malik, *UC Berkeley*

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

Abstract

Humans are incredibly dexterous. We interact with and manipulate tools effortlessly, leveraging touch without a second thought. Yet, replicating this level of dexterity in robots is a major challenge. While the robotics community, recognizing the importance of touch in fine manipulation, has developed a wide variety of tactile sensors, how best to leverage these sensors for both perception and manipulation is unclear. In this thesis, we address how to efficiently integrate tactile sensing for robot perception and dexterous manipulation.

Specifically, we turn to *self-supervised learning* (SSL) to train tactile representations that can generalize across sensors, standardize usage across downstream tactile tasks, and further alleviate the need to collect labeled task data which is often impractical to collect for tasks such as uncalibrated force field estimation. To this end, we discuss Sparsh and Sparsh-skin, a family of SSL models for vision and magnetic-skin based tactile sensors respectively. Sparsh and Sparsh-skin are trained via self-distillation for full-hand tactile sensors in downstream tasks. We find that both Sparsh and Sparsh-skin not only outperform task and sensor-specific end-to-end models by a large margin, but also that they are data efficient for downstream task training.

Second, we note that existing work often overlooks the multimodal aspects of human touch, such as vibration and heat sensing. We discuss Sparsh-X, a compact tactile representation fusing image, pressure, audio and inertial measurements from the DIGIT360 sensor. With Sparsh-X we demonstrate that multimodal sensing improves both passive perception tasks as well as dexterous manipulation tasks such as in-hand rotation.

Finally, we present *privileged tactile latent distillation* (PTLD), a novel method to imbue tactile sensing in dexterous manipulation policies trained via reinforcement learning. PTLD avoids simulating tactile sensors and uses *privileged sensors* to bridge the sim-to-real gap. With PTLD, we first show that one can improve existing RL trained policies such as in-hand rotation and then that it can enable learning more challenging tasks such as in-hand reorientation.

Jointly these contributions provide a path to leverage tactile sensing in both imitation and reinforcement learning based robot manipulation.

Acknowledgments

This PhD was a long, meandering experience, shaped by many people who made the time I spent pursuing it both enjoyable and memorable. I want to take this opportunity to thank them for being who they are and for making this journey possible.

First and foremost, I am grateful to my advisor, Michael Kaess, for his patience, guidance, and belief in me over the years. He gave me the freedom to explore my own ideas and the space to fail. He impressed upon me the importance of writing rigorously and ideating early; most importantly, however, he taught me to be kind. I am deeply indebted to him for his support and for the ideas we explored through our discussions over the last six years. I consider myself very fortunate to have had the opportunity to work with him

I want to thank Mustafa Mukadam, who was unofficially my second advisor and mentor. I still vividly remember our first meeting where we discussed an entirely different project direction, only to end up working on tactile representations for dexterous manipulation. I will cherish the time we spent working together as part of the GUM team. I am grateful that you provided a launchpad for my career and believed in my ideas and capabilities. You taught me to take it easy at times, and to be resilient in the face of problems. I am so glad to have found a mentor in you.

I am grateful to Shubham Tulsiani, Guanya Shi, and Jitendra Malik for agreeing to serve on my committee without hesitation and for supporting my research. I look up to your careers as trajectories I would be thrilled to follow. I want to thank Shubham for helping me with research at the beginning of my PhD and for pushing me to ask better research questions. I thank Guanya Shi for finding the time to discuss research and for creating opportunities for me. Finally, I want to thank Jitendra Malik for supporting my research ideas, first at FAIR Meta and later as a committee member. You have unknowingly opened several doors for me, and for that, I am deeply thankful.

The CMU Robotics Institute is an eclectic melting pot of researchers working on diverse interdisciplinary topics. I am fortunate to have met so many excellent researchers and friends within this community and beyond. I am thankful to Sebastian Scherer, Katerina Fragkiadaki, and Simon Lucey for their research guidance during my formative years. I would like

to thank Deva Ramanan for making me feel like a part of his lab and for being open to collaboration. I spent my first summer in Seattle with the Surreal team at Meta, where I was fortunate to work with Tianwei Shen, Julian Straub, and others. I learned a great deal about systems, writing fast code, and lattes! Finally, I am grateful to my labmates at RPL: Wei Dong, Allie Chang, and Ruoyang Xu. To my other labmates, I am thankful for the discussions that led to breakthroughs and for the fun moments we shared.

My PhD would be incomplete without mentioning the two years I spent at FAIR, Meta. I am thankful for the incredible colleagues and friends I made there. Carolina Higuera was simultaneously an amazing friend and co-author; I value your work ethic and am glad we had the opportunity to work on the Sparsh series together. I want to thank the ”ye old” GUM team: Mustafa Mukadam, Tingfan Wu, Francois Hogan, Carolina Higuera, Luis Pineda, Tess Hellebrekers, Changhao Wang, Taosha Fan, Mike Lambeta, Patrick Lancaster, Krishna Bodduluri, Ishita Prasad, Mrinal Kalakrishnan, Alex Yin, Jessica Yin, and Youngsun Wi. I truly believe we influenced the robotics community toward tactile sensing and generalizable dexterous manipulation. Without your hardware support, ideas, and discussions, none of this work would have seen the light of day.

Next, I would like to thank my friends. I am especially grateful to the cohort in Deva's lab and friends from Smith Hall for making me feel welcome: Jay Karhade, Neehar Peri, Gautam Gare, Nikhil Keetha, Himangi Mittal, Nupur Kumari, Jonathan Luiten, Zhiqiu Lin, Sally Chen, Arun Vasudevan, Jeff Tan, Khiem Vuong, Sriram, Anish Madan, Poorvi Hebbar, Rakshanda Hassan, Gaurav Parmar, Ruihan Gao, Mehar Khurana, and Bart Duisterhof. Thank you for sharing a love for Indian food, many meals, and much laughter. I am also glad to have met Unnat Jain, Homanga Bharadwaj, Anshika Gupta, Priyam Parashar, Ben Newman, and several friends at FAIR. Unnat taught me not to fret over perfection, introduced me to climbing, and helped me enjoy my PhD. From Homanga, I learned the importance of being nonchalant and working quietly toward one's goals. I also want to thank my climbing and running friends: Unnat Jain, Jayesh Singla, Jay Karhade, Rosy Chen, James Emilian, Andrew Wong, and Changhao Wang.

I am also grateful to the amazing mentees (and importantly friends) I have had: Rosy Chen, Jay Karhade, Trinity Chung, Servesh Patil, Irmak Guzey, Chaoyi Pan, and my high school mentees. Mentorship is a two-way

street, and I have learned a lot from all of you. To Rosy: you have a way of getting things done. I will cherish our conversations and our shared love for the mountains. To Jay: I am amazed by your energy and enthusiasm for life.

I would be remiss not to thank my wife, Tarasha Khurana, whose unfettered support made this PhD possible. She has lived through the ups and downs with me, and I am extremely grateful to have her by my side. On the research side, she taught me how to formulate better experiments to probe hypotheses from first principles; personally, I have learned from her how to wonder, observe, and appreciate the simple things in life.

Finally, I would like to thank my family. My father, Prakash Sharma, instilled in me a drive for excellence and fed my inquisitive nature. My mother, Champavathi Sharma, for always being on my side, checking on me every few days, and chiding me when I was wrong! I am also thankful to Mamata Ma'am (Maa); you provided vital financial support, and without you, none of this would have started. I am very fortunate to have a strong support system in my extended family, and I am glad to stand on their shoulders.

Contents

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

List of Figures

List of Tables

Chapter 1

Introduction

As humans, we are incredibly dexterous, we are able to use our sense of touch and proprioception effortlessly in multiple ways without giving it a second thought. We are able to coordinate our fingers to use complex tools such as scissors. We also have an implicit understanding of pressure we apply on different objects, which enables us to perform careful controlled maneuvers such as scraping a horse nail as visualized in fig. 1.1. In fact, our sense of touch is so ingrained that we also sense the tiny bumps on the road and the weight of the steering wheel, when driving a car through indirect contact. When this 'sense of touch' is lost, our ability to maintain grasp of objects becomes inaccurate and uncontrolled [185].

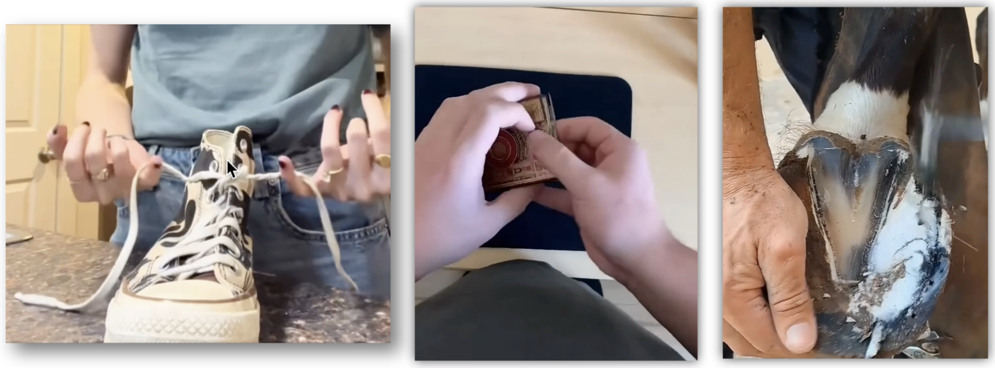


Figure 1.1. Humans are incredibly dexterous and are capable of impressive dexterous manipulation.

The robotics community – of course – recognizes the importance of touch, and over the past decade has developed a plethora of tactile sensors [116] in hopes of replicating the dexterity displayed by humans. However, even though there has been significant progress in robot learning, we are yet to see effortless robotic dexterous manipulation skills.

In contrast to general robot manipulation, dexterous manipulation is typically defined from an object centric point of view where multiple manipulators or fingers coordinate to grasp and manipulate an object [130]. Therefore, dexterous manipulation in principle assumes the use of a multi-fingered robot hand, instead of simpler manipulators such as parallel-jaw grippers. Multi-fingered robot hands are difficult to control due to a higher degree of freedom, and large action space, however they offer the versatility to study fine manipulation and promise true generality in terms of robotic tool use [25, 152, 202] and functional object grasping [159] without needing specific environment orchestration. When dealing with such contact-rich tasks with multi-fingered dexterous hands, the role of tactile-sensing is clear: Tactile sensing offers high observability into the state of the system, when other observation modalities such as vision experience high-occlusion.

Initial attempts in incorporating tactile sensing for robot learning leveraged tactile sensors for passive perception tasks such as pose estimation, reconstruction, object classification, property estimation, grasp stability classification and slip detection among others [20, 88, 163, 166, 206]. A common theme across these approaches has been to leverage supervised learning with custom datasets and architectures to tailor a solution specific to the type of tactile sensor being used in the work. Once the perception output is obtained, the hope is to then use these outputs in robot policies in a modular manner. However, such an approach with supervised learning is not scalable. Collecting *labeled* tactile data is expensive not only because it requires interactions in the real world like other embodied data, but also because obtaining ground truth labels for tactile tasks is challenging. In many cases such as shear estimation over a contact area, obtaining labels is even infeasible due to a lack of sensors to obtain the labels. The scalability is further exacerbated by the fact that one needs to further tailor and repeat data collection for every different type of tactile sensor that is being used.

Several other approaches have also leveraged tactile sensors in learning manipulation directly. There are two paradigms primarily in robot learning: a) Learning from demonstrations and b) Learning from experience (in simulation). For learning from demonstrations, the primary challenge is in obtaining tactile demonstration data for learning robot skills. Collecting data via tele-operation is impractical for multi-fingered robot hands as we do not yet have reliable techniques to tele-operate the robot without haptic feedback. On the other hand, kinesthetic teaching can produce some dexterous behaviors, but controlling multiple fingers with two human hands is quite challenging. For learning from simulation, several works attempt incorporating tactile sensing in learning dexterous skills such as in-hand rotation, and reorientation [25, 141, 199, 201], however most of these approaches end up simplifying tactile sensing to binary contact or single point of contact models to simplify simulation. Tactile simulation requires soft-body simulation which is quite slow and not yet amenable to RL based policy learning.

In this thesis, we therefore explore the natural question of how can one minimize the effort in obtaining tactile data to learn robot dexterity. Specifically, we explore several ways to efficiently incorporate tactile sensing in both robot perception as well as robot dexterous manipulation. To this end, we provide answers to the following questions:

1. Can we improve sample efficiency for downstream tactile tasks by learning from *play* data?
2. What kind of self-supervised objectives are suitable for tactile sensors?
3. If tactile simulation is difficult, can we learn tactile dexterous policies in the real world from proxy sources?

1.1 Organization

This thesis can be divided into three major parts (see fig. 1.2).

In Part 1, we turn to *self-supervised learning* to train tactile representations that can generalize across sensors, and standardize the usage across downstream tactile tasks. Here, in **chapter 2** we will introduce Sparsh – which means 'to touch' in Sanskrit / Hindi. Sparsh is a family of representation models for vision-based tactile sensors that works across three different commonly used visuo-tactile sensors. Then in

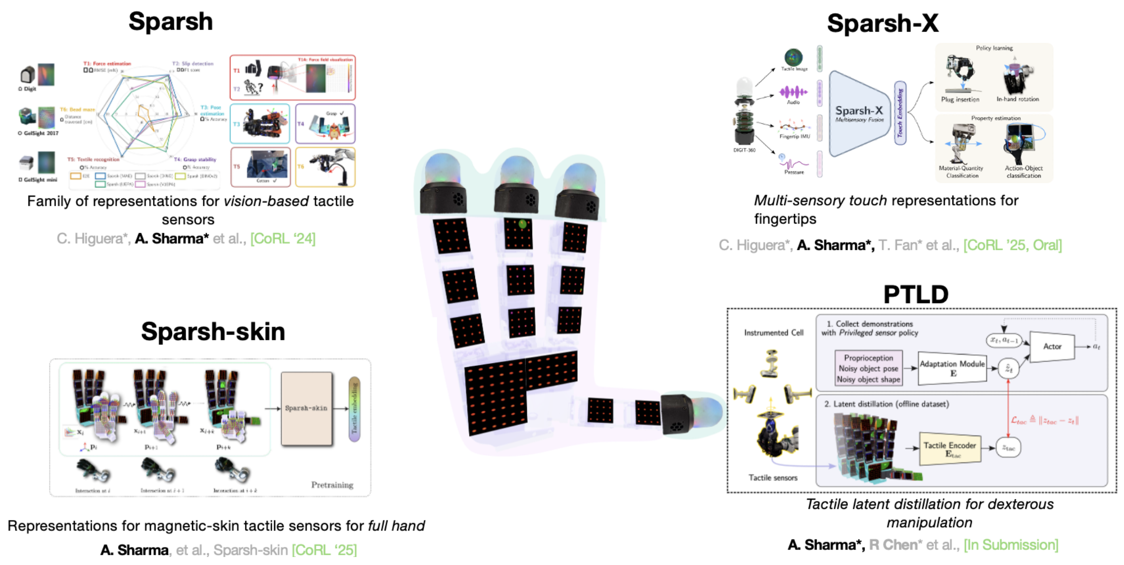

Sparsh
Family of representations for *vision-based* tactile sensors
C. Higuera*, A. Sharma* et al., [CoRL '24]
Sparsh-X
Multi-sensory touch representations for fingertips
C. Higuera*, A. Sharma*, T. Fan* et al., [CoRL '25, Oral]
Sparsh-skin
Representations for magnetic-skin tactile sensors for *full hand*
A. Sharma, et al., Sparsh-skin [CoRL '25]
PTLD
Tactile latent distillation for dexterous manipulation
A. Sharma*, R Chen* et al., [In Submission]

Figure 1.2. This thesis is divided into three parts: Part 1 includes Sparsh and Sparsh-skin, which explore different self-supervised objectives that are suitable for tactile sensors. Part 2 includes Sparsh-X, where we extend self-supervised learning to multisensory touch, and finally in Part 3, we introduce PTLD which is a novel method to learn tactile dexterous manipulation in the real world without having to simulate tactile sensors.

chapter 3 we present self-supervised representation learning for full-hand magnetic skin based tactile sensors, which cover the rest of the robot hand. Both of these works demonstrate the power of learning self-supervision from *play* data, particularly showing significant improvements in various downstream tasks, including both robot perception tasks as well as manipulation.

In Part 2, we recognize that existing tactile sensors are largely unimodal and do not consider the multisensory nature of touch, unlike humans, who sense pressure, vibrations, proprioception and surface deformation as part of touch. To this end, in **chapter 4** we introduce Sparsh-X which learns multi-sensory tactile representations for a tactile sensor such as DIGIT360, and we demonstrate that unifying disparate modalities such as tactile images, audio, IMU and pressure, together is beneficial. In Sparsh-X we also present a novel method to improve dexterous manipulation policies via tactile adaptation in ther real world.

Finally, in Part 3, we present privileged tactile latent distillation (chapter 5), where we show that we can learn dexterous tactile policies using simulation, but do not really have to simulate tactile sensors. With Privileged tactile latent distillation, we learn tactile encoders by distilling the latent representations from *privileged sensor* policies being deployed in the real world, using an instrumented cell as the bridge between sim-to-real.

Chapter 2

Sparsh: Self-supervised touch representations for vision based tactile sensing

In this work, we introduce general purpose touch representations for the increasingly accessible class of vision-based tactile sensors. Such sensors have led to many recent advances in robot manipulation as they markedly complement vision, yet solutions today often rely on task and sensor specific handcrafted perception models. Collecting real data at scale with task centric ground truth labels, like contact forces and slip, is a challenge further compounded by sensors of various form factor differing in aspects like lighting and gel markings. To tackle this we turn to self-supervised learning (SSL) that has demonstrated remarkable performance in computer vision. We present `Sparsh`, a family of SSL models that can support various vision-based tactile sensors, alleviating the need for custom labels through pre-training on 460k+ tactile images with masking and self-distillation in pixel and latent spaces. We also build `TacBench`, to facilitate standardized benchmarking across sensors and models, comprising of six tasks ranging from comprehending tactile properties to enabling physical perception and manipulation planning. In evaluations, we find that SSL pre-training for touch representation outperforms task and sensor-specific end-to-end training by 95.1% on average over `TacBench`, and `Sparsh` (`DINO`) and `Sparsh` (`IJEPA`) are the most competitive, indicating the merits of learning in latent space for tactile

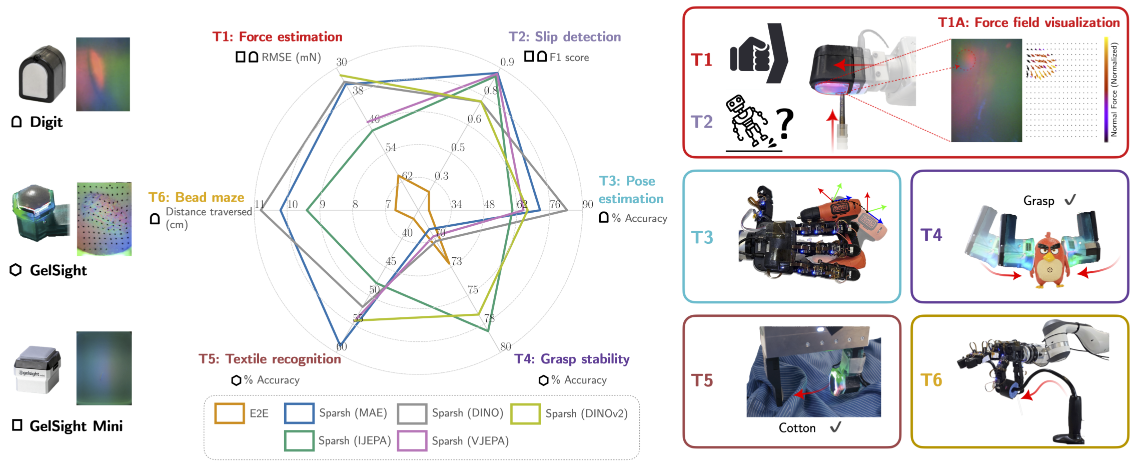


Figure 2.1. We present `Sparsh`, a family of general touch representations, and `TacBench`, a standardized benchmark of six touch-centric tasks (`[T1]`-`[T6]`) covering prominent problems in vision-based tactile sensing. We find `Sparsh` pre-trained with self-supervision on a dataset of 460k+ tactile images can generalize across many tasks (right) and sensors (left) outperforming task and sensor specific models (`E2E`). Performance in the plot (middle) is with task decoders using 33% labeled data (except `[T6]` that uses 50%).

2.1 Introduction

Touch comes before sight, before speech. In today's AI landscape, this Margaret Atwood quote is playing out in reverse despite touch being a crucial modality for humans to physically interact with the world. Touch provides a direct window into information like forces and contact during hand-object interactions, enabling dexterity. Vision-based tactile sensors [45, 100, 105, 207] have emerged as the leading form factor capable of capturing images of physical interactions at the sensor-object-environment interface, often inaccessible through vision. These images contain properties such as contact geometry, texture, and forces and have been leveraged across tasks like insertion [43, 81], pushing [114], grasping [19], localization [163], and pose and shape estimation [13, 164].

The prevailing approach to incorporating vision-based tactile sensors in robot tasks is to train custom models using labeled data [35, 81, 111, 191] to estimate useful states. However, this can be inefficient and results in repeated effort across different type of sensors like GelSight 2017 [207] (with markers) and DIGIT [100] (without markers) or different variety of tasks. For example, feature extractors trained on GelSight with markers may not transfer to other sensors, and encoders optimized for texture recognition [212] may not be suitable for tasks that require reasoning about forces or slip [206]. Supervision for building large general models is prohibitive as collecting large scale real world data with ground truth labels is challenging. For instance, properties like forces [115] and slip [17] require careful and expensive instrumentation in lab settings, while other properties like tracking deformations [49] or extrinsic contact [81] can be infeasible. To address this fragmentation in the literature across custom solutions, there is a need for touch representations that are broadly applicable to many tasks and many sensors, along with a benchmark of standardized tasks useful in measuring progress. Taking inspiration from self-supervised learning (SSL) methods in computer vision, we extend these approaches to the tactile sensing domain and build a benchmark for evaluation (Figure 2.1).

In this work, we introduce a family of touch representations for vision-based tactile sensors trained with SSL. Specifically, we provide a recipe to adapt masking-based objectives from computer vision to the tactile domain, and train general-purpose touch encoders by curating a new Touch-Slide dataset and existing datasets of tactile images (Figure 2.2), namely YCB-Slide [163], Touch-and-Go [196], and ObjectFolder [60]. Pulling together additional unlabeled data points from the existing datasets we train our models on a total of 460k+ tactile images. Finally, we construct `TacBench`, a benchmark consisting of six touch-centric tasks that cover the space of relevant problems on tactile properties such as force estimation and slip detection, on perception such as pose estimation and grasp stability, and on robot manipulation such as policies for solving a bead maze. Our contributions are as follows:

1. General touch representations, `Sparsh` pre-trained with SSL on 460k+ tactile images,
2. `TacBench` a benchmark of standardized tasks to evaluate touch representations and models, and
3. Curation of new & existing datasets, unlabeled for SSL and labeled for benchmarking.

In evaluations on `TacBench`, we find that `Sparsh` with SSL pre-training yield on average 95.1% improvement over task and sensor specific end-to-end models under limited labeled data budget (33%-50% of the collected amount) for any task.

Additionally, we find Sparsh (DINO) and Sparsh (IJEPA) to be the most competitive outperforming Sparsh (MAE), indicating the merits of learning in latent space over pixel space for tactile images.

2.2 Related work

Self-Supervised Learning with its success in natural language processing and computer vision, has become the new learning paradigm. In the last three years, a variety of general-purpose frameworks [26, 29, 67, 78] have been proposed for learning representations. We refer to [8, 132] for a comprehensive survey on SSL frameworks and their categorization based on pretext tasks and learning algorithms. In Appendix 2.10.1, we expand on Masked Image Modeling (MIM), self-distillation, and Joint-Embedding Predictive Architecture (JEPA), as we explore them in this study.

Traditionally, tactile sensing has relied on preprocessing tools like marker tracking and finite element method models to extract contact properties, such as shear forces [170, 206], dense normal estimation [117, 170], and contact area prediction [101]. From a learning perspective, a trend is to use custom encoder architectures tailored for specific tasks and sensors, which are either pre-trained or trained end-to-end [30, 35, 75, 81, 111, 163, 191]. Nevertheless, there is an increasing interest in representation learning for vision-based tactile sensors. For instance, MAE has shown effectiveness at material classification and texture recognition [21]. Fine-tuning convolutional encoders for BioTac, RoboSkin, and GelSight performs well on fabric decomposition tasks [212]. Even nearest-neighbor retrieval over pretrained representations, for the XELA [174] uskin sensor, can enable some success in dexterous manipulation [70]. Crucially, the current state of standardization in learning touch representations and the wide variety of tactile sensors available, has made it challenging to develop and share pre-trained models across the research community in this domain.

Another direction is exploring the alignment of visual and tactile modalities in latent space using multimodal datasets [19, 54, 59, 61, 62, 196] and techniques like contrastive coding and cross-sensory retrieval [39, 197, 205], yielding promising results in tasks like material classification, grasp stability, and tactile-driven image stylization. However, current approaches [39, 48, 95, 197] primarily focus on texture and visual

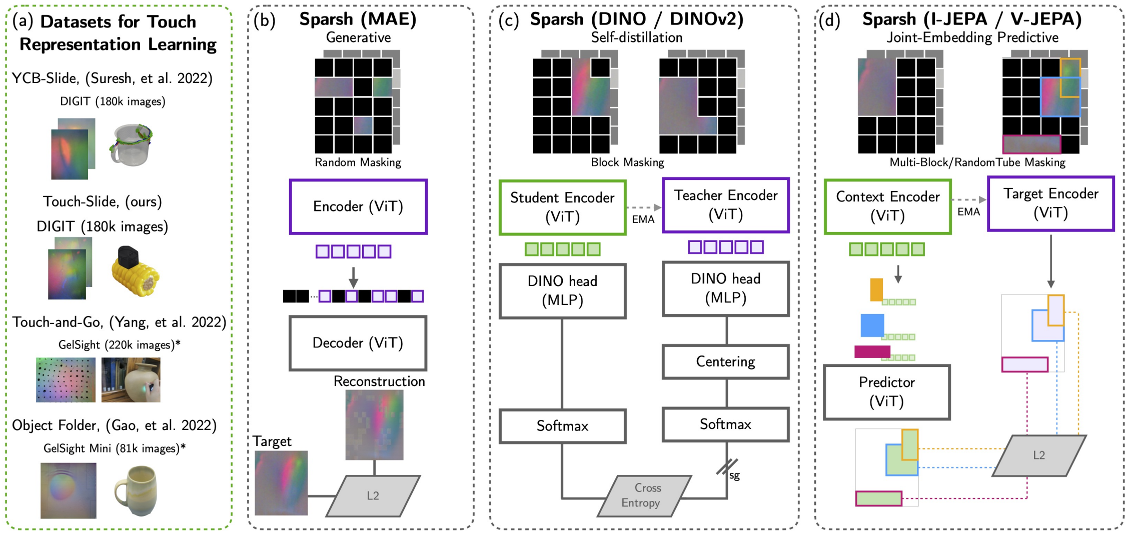


Figure 2.2. (a) We curate new and existing datasets of vision-based tactile sensors to train touch representations by adapting state-of-the-art SSL vision methods to the tactile domain, namely (b) Masked Autoencoder (MAE) [78], (c) DINO/DINOv2 [22, 131], and (d) Image/Video Joint-Embedding Predictive Architecture (JEPA) [4, 11]. *Without need for labels we can sample more images than reported in Touch-and-Go [196] and Object Folder [60].

properties and overlook physical contact properties, such as forces, slippage, and poses, which are essential for dexterous manipulation.

The works closest and concurrent to ours are T3 [215] and UniT [193]. T3 trains sensor-specific encoders to capture shared latent information through a shared trunk, using both the MAE objective and labeled task-specific data as supervision. UniT is a VQGAN [177] model with a patch-based discriminator for representation learning only for GelSight Mini (markers). On the other hand, we introduce a family of models trained with the latest SSL algorithms for the three most commonly used families of tactile sensors: DIGIT, GelSight 2017, and GelSight Mini. Similar to T3 and UniT, we evaluate touch representations for policy learning, however we also introduce a standardized benchmark to comprehensively evaluate representations and their ability to solve several relevant touch-centric tasks along tactile properties, physical perception, and manipulation planning.

2.3 Touch representations via self-supervision

Current approaches incorporating vision-based tactile sensors in robotic tasks lean on custom task and sensor specific solutions. As highlighted in the introduction, this can be inefficient, and there is a growing need for general-purpose touch representations that can be more broadly useful. We envision the following guiding principles for such general touch representations: (i) provide performance benefits across many tasks including real-time robot manipulation, (ii) generalize across multiple types of sensors built on a similar operating principle, like vision-based tactile sensors, and (iii) improve performance by leveraging computation and diverse data at scale without the need for manual labels. Self-supervised learning (SSL) is promising in this regard, as it offers data-agnostic objectives based on wide-reaching concepts such as analysis-by-synthesis to train generalist models. This motivates the question of whether vision techniques such as masked image modeling (MIM) [4, 78] and self-distillation [22, 131] can be extended to the domain of vision-based tactile sensors.

To this end, inspired from advances in self-supervised learning (SSL) in computer vision, we introduce Sparsh, a family of touch representation trained with SSL across multiple sensors such as DIGIT [100], GelSight 2017 (with markers [207]), and GelSight Mini (without markers). The tactile domain, however, imposes several challenges that impede a straightforward application of SSL approaches from vision towards touch representations.

Tactile sensors inherently provide local information; thus, images can be ambiguous when observed independently and can vary across grasp forces, materials, and shapes. Therefore, we investigate the optimal space for training SSL encoders. Specifically, we are interested in the efficiency of pixel reconstruction, latent reconstruction, or clustering approaches to learning representations in the presence of aforementioned ambiguities. We hypothesize that latent reconstruction and clustering could be more efficient in learning representations, as they focus model capacity away from fine reconstruction details [103]. Tactile images contain distractors, such as markers and light placement variations, which can significantly vary due to manufacturing discrepancies. To increase robustness to distractors, we perform background subtraction for both DIGIT and GelSight Mini (markerless). This process provides the model with a reference to no-contact, which conveys static shear information when a perpendicular force is applied to the elastomer. We find empirically that background subtraction helps models generalize across the same type of sensor.

To address the scarcity of labeled and even unlabeled data in the tactile domain that limits the training of large encoder models, we curate together new and existing vision-based tactile sensor datasets [60, 163, 196], totaling ∼661k images as illustrated in Figure 2.2. 70% or 462.7k images are used for SSL pre-training which is an order of magnitude larger than any prior work on touch representations [21, 48, 197] (the rest are held out for monitoring training using online probes).

Tokenizing tactile images appropriately for SSL is important because many tasks such as slip detection and relative pose estimation require temporal reasoning. For SSL methods that operate on images, we concatenate two tactile images with a temporal stride of 5 samples across the channel dimension, $I_t \oplus I_{t-5} \rightarrow x \in \mathbb{R}^{h \times w \times 6}$. For a sensor operating at 60FPS, this corresponds to an inference window of approximately 80 ms, the reaction time that humans need to adjust the grip force when detecting partial slip [213]. For SSL methods that operate on video (e.g. V-JEPA), we generate clips with 4 frames at $[t, t-2, t-4, t-6] \in \mathbb{R}^{4 \times h \times w \times 3}$ corresponding to an inference window of ~ 100 ms. Currently `Sparsh` is limited by data streaming rates, and not by inference time, as the models support inference rates of upto 112FPS (measured on an Nvidia RTX3080). See Appendix 2.10.2 for additional details on model architectures and training.

2.4 `TacBench`: Tactile sensing benchmark

We introduce `TacBench`, a collection of touch-centric tasks, and labeled datasets for standardized evaluation for vision-based tactile sensing. We compile data for all tasks from various sensors to evaluate the generalization of representations. These tasks are categorized under three main questions.

1. Do the representations comprehend tactile properties? Tactile sensing informs finger-object contact interaction properties like forces and slip that are crucial for robot manipulation. In Section 2.5, we evaluate learned representation on estimating instantaneous normal and shear forces `[T1]` and visualizing force fields `[T1A]` [115, 117, 170, 206], and detecting slip `[T2]` [17, 41, 179, 206].

2. Do the representations enable perception? Tracking and accumulating slip states is essential for tasks like finger-gaiting and in-hand reorientation [141, 198]. In Section 2.6, we evaluate the ability of the representations to track SE(2) pose changes of the object relative to the sensor `[T3]` [13], prediction of the stability of a grasp `[T4]` [19], and textile recognition `[T5]` [209].

3. Do the representations enable manipulation planning? Pre-trained representations can provide tactile features to a manipulation policy, improving training efficiency and test performance by eliminating the need to extract the states from raw sensor data. In Section 2.7, we design a bead maze `[T6]` manipulation problem as illustrated in Figure 2.3 (c), where the robot using tactile sensing is tasked to move a bead along a curved wire.

Evaluation protocol. We adopt a frozen evaluation procedure with an encoder-decoder architecture. Specifically, we *freeze* the pre-trained `Sparsh` encoder weights and train the parameters of an attentive decoder [11, 28] to assess what touch representations have captured from self-supervised pre-training alone. All tasks in the benchmark, except force field visualization and policy learning, train an attentive decoder containing a cross-attention module and a two-layer MLP using labeled datasets from Table 3.3. We also include an end-to-end (`E2E`) baseline with identical model capacity where the same encoder and decoder probe are initialized with random weights and all the parameters (both encoder and decoder) are trained. Further, we train downstream decoders with different amounts of labeled data to evaluate task performance under progressively low labeled data regimes.

In the following sections, we describe the design, metrics and results of each task in `TacBench`. Additional details are provided in Appendix 2.10.3, and ablations with unfrozen `Sparsh` encoder, encoder model sizes, and few-shot cross-sensor transfer are provided in Appendix 2.10.4.

2.5 Comprehending tactile properties

2.5.1 `[T1]` Force estimation

Task. Force estimation is defined as the prediction of 3-axis normal and shear forces applied on the sensor's elastomer. Figure 2.3(a) shows our data collection setup. We use three different indenter shapes to collect force-labeled data: hemisphere, sharp, and flat. Our dataset contains 75k time-aligned samples of 3-axis force measurements, end-effector poses, and tactile images from DIGIT at 60fps and GelSight at 25fps. We train the decoder using normalized force measurements scaled between $[-1, 1]$, supervised using $\mathbf{L_1}$ loss and optimized using Adam until convergence. We compare performance using the average root mean squared error (RMSE) across all three axes.

Results. GelSight Mini images are of higher resolution (HD) compared to DIGIT (320×240) resulting in smaller contact regions against the background. For this reason, we observe that when sufficient supervised data is available for DIGIT, it is possible to train a model from scratch to achieve high accuracy, but for GelSight Mini the end-to-end model does not perform well. Figure 2.4 (i)-(ii) shows across the board that our frozen Sparsh representations can estimate forces with low error. Specifically, we find Sparsh (DINO) to be robust even when access to labeled data is sparse, a common scenario in tactile sensing. Additional details are in Appendix 2.10.3.

2.5.2 [T1A] Force field visualization

We qualitatively evaluate the representations for rendering normal and shear force fields to understand sensor-object interactions. Although obtaining a shear field for sensors with markers nowadays is trivial via marker tracking [41], it is challenging and underexplored for markerless sensors. We train a CNN decoder using the reassemble-fusion approach for dense predictions [144] unsupervised, since we do not have access to ground truth for markerless sensors. We frame normal field estimation as depth estimation [64] and shear field estimation as optical flow [90, 91, 112, 161]. Figure 2.4 (vi) shows visualizations for the top-performing model Sparsh (DINO) in [T1] that provides directional information about the relative motion of the contact patch. For instance, sliding motion (a, c, e, f), torsional slip (b), and divergence field upon contact (d). Additional details are in Appendix 2.10.3.

2.5.3 [T2] Slip detection

Task. Shear and slip are closely related. Using the same setup as force estimation, we collect strokes where a hemispherical probe slides over the sensor, producing trajectories with both sticking and slipping samples. Slip is labeled using the friction cone model with an empirically estimated static friction coefficient (see Appendix 2.10.3). The dataset, with a notable imbalance between no-slip and slip classes, contains 125k samples with 13% slip instances. We train two decoders: one for slip detection and

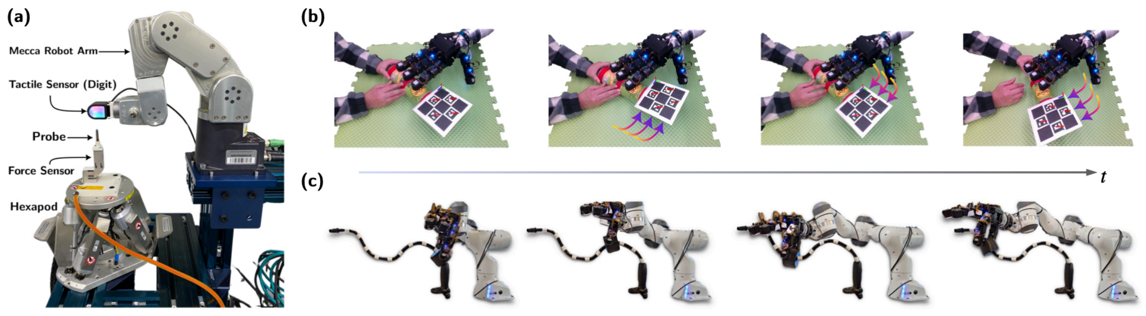


Figure 2.3. Real labeled data collection setup for TacBench tasks (a) [T1] Force estimation and [T2] Slip detection, (b) [T3] Pose estimation, and (c) [T6] Bead maze.

another for normalized force changes (Δ) as we find that predicting the two correlated quantities jointly enhances slip detection. The MLP decoders use cross-entropy for slip detection and mean absolute error for Δ force regression, reserving 25k samples for evaluation.

Results. We report F1 score instead of accuracy due to the imbalance in the slip labels in the dataset. Figure 2.4 (iii)-(iv) illustrates the advantages of frozen Sparsh features trained under a JEPA paradigm for slip detection, particularly challenging for DIGIT sensor, even when using only 1% of the training dataset. In particular, Sparsh (VJEPA) achieves the highest F1 score among the models. Although all models detect slip from the 80 ms history of tactile data, Sparsh (VJEPA) benefits from a detailed temporal perspective, as its encoder processes a video clip with four frames spanning this window. Sparsh backbones also show better performance than the E2E model when labeled training data is significantly reduced. Additional details are in Appendix 2.10.3.

2.6 Enabling physical perception

2.6.1 [T3] Pose estimation

Task. Estimating object pose changes can help tasks such as tracking object drift for in-hand translation [96], rotation [141, 198], and pushing [114], among others. Given that tactile images capture local changes between sensor-object, we evaluate Sparsh representations to estimate $\mathbf{SE}(2)$ transformations of the object relative to the sensor. Figure 2.3 (b) illustrates the data collection procedure. The dataset consists of time-synchronized pairs of DIGIT observations $\mathbf{z}_t \in \mathbb{R}^{h\times w\times 3}$ and object poses $\mathbf{T}_t \in \mathbf{SE}(3)$. $\mathbf{T}_t$ are then preprocessed to produce relative pose changes on the sensor gel as $\mathbf{S}_t^{t-1} \triangleq (\Delta x, \Delta y, \Delta\theta) \in \mathbf{SE}(2)$. We follow the regression-by-classification paradigm for this task [13, 96]. Relative object poses are binned into a grid, capturing translations with a resolution of $\pm$5mm and rotations with a resolution of $\pm 2°$. For each degree-of-freedom (DOF), we train a head to predict probability distribution over the discretized grid using cross-entropy loss and Adam optimizer.

Results. Multiclass accuracy reveals that `E2E` approaches perform well with ample data, but drastically decline when labeled data is reduced, as shown in Figure 2.4 (v). Small datasets make it difficult to distinguish between close categories, such as orientation changes from $[0.5°, 1.0°]$ to $[1.0°, 2.0°]$. Pre-trained representations, however, maintain good performance even with only a third of the data. In low data scenarios, decoders using tactile representations often revert to extreme values, reducing estimation resolution and accuracy. Additional details and examples are provided in Appendix 2.10.3.

2.6.2 `[T4]` Grasp stability

Task. Grasp stability is well-studied in the tactile sensing literature for parallel jaw grippers [92, 97, 156, 197]. We evaluate whether representations aid in predicting grasp success given a short history of tactile images from a single finger. Specifically, we take inspiration from [19] and adapt the Feeling of Success dataset. Each sample consists of a triplet of tactile images corresponding to 'before', 'during', and 'after' grasping a set of objects. The dataset consist of 64% successful grasps and 36% failed grasps. We pass to the SSL model the 'before' and 'during' as tactile history. Since [19] does not specify an official train/test split, we create our randomized split with all objects, using approximately 8k grasps for training and the remaining 1.3k grasps for evaluation.

Results. Training with the full dataset, all models achieve similar accuracy. `Sparsh` (`IJEPA`) or `Sparsh` (`VJEPA`) reach $\sim 80\%$ classification accuracy, surpassing results from [19] that combined tactile and vision modalities as shown in Figure 2.4 (vii). Our model, relying solely on touch from a single finger, shows competitive performance even with only 33% and 10% of the data. However, with just 80 training samples, performance drops significantly. More details are in Appendix 2.10.3.

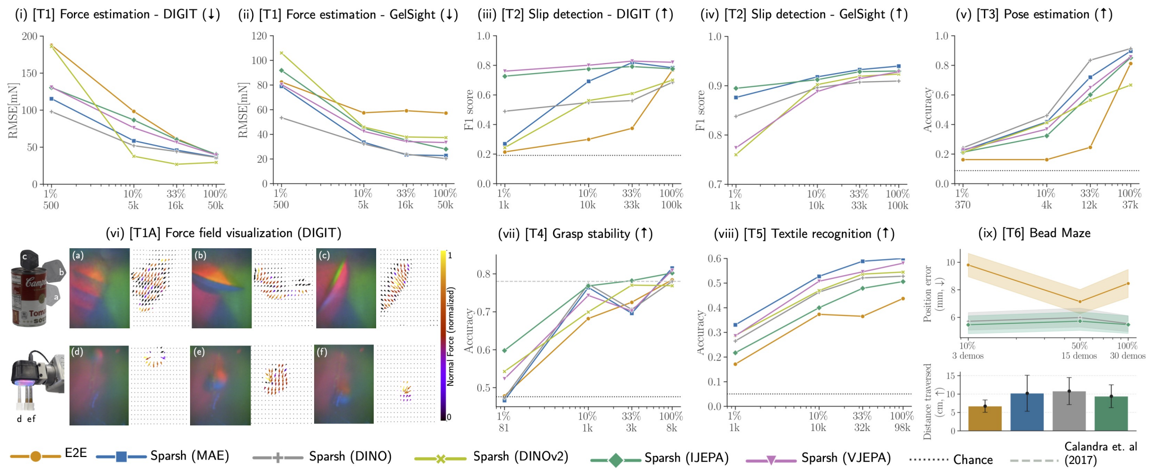


Figure 2.4. Summary of results comparing `Sparsh` and `E2E` on `[T1]`-`[T6]` tasks in `TacBench` across varying amounts of labeled data. Pre-training with SSL yields general touch representations that work across several tasks and sensors outperforming task and sensor specific models particularly under limited labeled data budget.

2.6.3 `[T5]` Textile recognition

Task. Vision-based tactile sensors are broadly used for material property recognition, since their compliant gel and high resolution cameras make them effective at discriminating different materials by surface texture [54, 209]. Specifically, we take the task definition from [209] and adapt the Clothing Dataset. The dataset consist of 4467 short video clips (10-25 frames), of a robot with a GelSight 2017 (markers) grasping several types of textile (20 classes), such as leather, cotton, polyester, etc. We follow the train-test split provided in the metadata of the dataset.

Results. Training an `E2E` specialist model for textile recognition using the full dataset can be challenging, as noted in [209]. By leveraging pre-trained touch representations, as shown in Figure 2.4 (viii) the performance of the task can be significantly improved, even when training with only 10% of the labeled data. `Sparsh` (`MAE`) is particularly effective, as it heavily relies on pixel-level features (see Appendix 2.10.3). Additionally, we evaluate the cross-sensory ability of the representations, finding that with few samples (10-shot) `Sparsh` quickly adapts the downstream task to DIGIT (see Appendix 2.10.4).

2.7 Enabling manipulation planning

2.7.1 [T6] Bead maze

Task. The bead maze is a children's toy to enhance fine motor skills. We adapt this task to robot policy learning, where the goal is to guide a bead from one end to another following the path of the wire (maze). Given a small history of tactile images $(\ldots, \mathbf{z}_{t-1}, \mathbf{z}_t)$, and robot proprioception $(\ldots, q_{t-1}, q_t)$, we train a policy to predict changes in joint angles as actions $\mathbf{a} \triangleq (\Delta q_t, \Delta q_{t+1}, ...); \Delta q \in \mathbb{R}^7$, to make progress on this task. This task is fundamentally tactile-focused, as the robot needs to react to resistance encountered by changes in the maze pattern and the subtle local movement of the bead are difficult to perceive from vision even when not occluded by the hand. A prior version of the bead maze task has been explored in robotics relying solely on tactile feedback [85]. In our setup, illustrated in Figure 2.3 (c), we assume that the robot starts with an initial stable grasp. We collect a dataset of 50 demonstrations on different maze patterns with a mix of VR-based and manual kinesthetic-based teleoperation, corresponding to a total of ∼34k training pairs of tactile images and robot joint angles. Since we are training policies with real data, we use diffusion policy [32] for this task as it is one of the leading behavior cloning methods. For tactile observation conditioning, we replace the vision encoder in Diffusion Policy with the pre-trained `Sparsh` encoder.

Results. We evaluate `Sparsh` (`DINO`) and `Sparsh` (`IJEPA`) for policy learning, as these representations exhibit the best performance across the rest of the benchmark. For completeness, we also consider `Sparsh` (`MAE`), and `E2E` which trains a tactile encoder and policy end-to-end. Due to covariate shift [154] in behavior cloning, prediction errors can accumulate over time; therefore, we report position error between the predicted trajectory and a demonstration trajectory from a held-out maze sequence over small chunks of 3cm followed by the robot corresponding to 15 timesteps of action predictions. Figure 2.4 (ix) shows the position error over access to different number of demonstrations for training. We find that `Sparsh` (`DINO`) and `Sparsh` (`IJEPA`) produce significantly (a difference of ∼16%) lower trajectory errors compared to training the policy `E2E`.

Additionally, we evaluate real rollouts of the learned policies (using all 50 demonstrations) over a set of 10 randomized novel starting locations on the maze. In Figure 2.4 (ix) we report distance traversed (in cm) before failure. We find that policies using Sparsh representations outperform E2E by ∼20-53%. We note that given the high precision nature of this task and the considerations for real system deployment for the policy, none of the models succeeds in completing the full maze on real robot rollouts. We expect that increasing the diversity of training data with different maze patterns will highlight the generalist capabilities of touch representations, and that temporal ensembling will aid in improving the smoothness of the policy [216]. Additional details are in Appendix 2.10.3.

2.8 Discussion

Summary. We present Sparsh, a family of general touch representations trained with self-supervision for vision-based tactile sensors. We learn general-purpose, cross-sensor representations from a curated, unlabeled dataset of 460k+ samples from DIGIT, GelSight 2017, and GelSight Mini sensors. We evaluated five SSL approaches (see Figure 2.2) comparing their performance against task and sensor specific models through TacBench, a benchmark of six touch-centric tasks designed to assess the content and quality of the representations. Our results indicate that Sparsh representations are performant across various sensors and tasks capturing tactile properties, and enhancing physical perception and manipulation planning.

Analysis. Overall Sparsh excels on all tasks. In particular, we find Sparsh (DINO) is well suited for physics-based tasks like force and pose estimation, while Sparsh (IJEPA) performs better at touch semantic understanding like slip state, stability of a grasp, and textile recognition. On average Sparsh (DINO) outperforms Sparsh (IJEPA) by 5.6% across the benchmark. Both models perform similarly in bead maze test demonstrations, which require implicit knowledge of shear forces and slip. However, this did not translate to real robot performance due to lack of force control and system-level confounding variables not captured during training. These include the high precision required to keep the bead in place, the impossibility of error recovery once grip is lost, and trajectory drift due to local decision-making. Specialist policies or models trained from scratch exhibiting better robot rollout performance is due to the narrow task domain setting that leads to overfitting, a trend similarly observed when studying pre-trained vision models [32, 76, 216].

Learning touch representations in latent space is more advantageous than in pixel space, as these representations can filter out and generalize over noise or lighting differences. Tasks traditionally challenging for markerless sensors (like DIGIT and GelSight Mini), such as shear force (and field) estimation and slip detection, become solvable with our general touch representations. On average, `Sparsh` achieves a 95.1% improvement compared to an end-to-end approach when all models have access to only $33\% - 50\%$ of the labeled dataset per downstream task. Using as little as 10% or 1% of the labeled data for force estimation and slip detection still yields acceptable results (e.g. force error below 0.1N with `Sparsh` (`DINO`)). Fine-tuning `Sparsh` encoders is another method of assessing the quality of pre-trained representations. We provide in Appendix 2.10.4 experiments with partial and full fine-tuning. Notably, models pre-trained in latent space perform better in downstream tasks when fully fine-tuned, especially in regression tasks like force and pose estimation. In contrast, partial fine-tuning offers minor improvements, aligning closely with the performance of frozen models. We also evaluate `Sparsh` decreasing the model capacity, finding the biggest impact in performance for regression-like tasks when training with limited amount of labeled data (see Appendix 2.10.4).

`Sparsh` is a significant step towards a general pre-trained backbone for vision-based tactile sensors. Our aim is to enable efforts to compile larger tactile datasets that include additional vision-based tactile sensors and leverage the benefits of scaling up SSL backbones, as seen in computer vision and natural language processing. `TacBench` serves as an initial benchmark for evaluating these representations, and additional tasks can be incorporated based on the needs of the tactile sensing and manipulation community. For instance, further exploration of pre-trained touch representations in tactile policy learning, or tracking dynamic object properties like changing mass during pouring.

2.9 Limitations.

Open-source tactile datasets we considered in this study predominantly feature discrete contact interactions. We believe that incorporating data rich in shear interactions can further improve the representations. We do not ablate the length of tactile image history for learning the representations. Such ablations could provide guidance on improving their quality for downstream tasks. Our bead maze policies with pre-trained touch representations deployed on the real robot are only able to complete the maze partially before compounding error leads to the bead falling out of the fingers. Further research is needed to understand how to effectively leverage pre-trained touch representations in behavioral cloning for robot manipulation tasks.

2.10 Appendix

2.10.1 Broader related work

Self Supervised Learning. We detail recent developments in masking-based self-supervised learning approaches.

Masked Image Modeling (MIM) is the strategy of corrupting a data sample by significantly masking a portion of the sample and training a model to recover the missing portion, conditioned on the corrupt sample. It has become a prominent framework in SSL with the success of [78, 220]. An important design consideration here is the output space of the model for supervision, which can be either raw pixels [78, 190] or an alternative representation space [10, 57, 183, 220]. While training Masked auto-encoders is simple, these models are comparatively sample inefficient during training [4].

Self-distillation [84] is the idea of training two (usually identical) networks such that a *student* network learns to predict the output representations of a *teacher* [169] network via a small predictor network when observing augmentations of the same data sample. It has been shown to improve performance significantly even in the case of abundant data [189]. While degenerate constant representations is a concern, a common strategy is to stop gradient backpropagation [29] through the teacher network and employ momentum based weight updates [67]. A concrete instance is DINO [22] utilizing ViTs [46] as the student & teacher encoder networks. More recently DINOv2 [131] improved downstream performance significantly by combining self-distillation and MIM.

Joint-Embedding Predictive Architectures (JEPA) [103] share similarities with MIM, as both rely on masking. However, the JEPA framework conceptually prescribes two key changes: a) information restoration in a latent representation space, rather than in input space (pixels or tokens) b) prediction of latent embedding conditioned on the *masking parameters*. This framework has had success across various modalities, including audio [7, 51], images [4, 12], and pointclouds [148]. Notably, in this paper we consider masking strategies from I-JEPA [4] and V-JEPA [11]. I-JEPA utilizes a spatial block-masking strategy and V-JEPA utilizes tube-masking [176] with varying aspect ratios for learning representations efficiently in latent space circumventing decoding unnecessary pixel-level details.

Representation learning in robotics. Pretraining models for multi-task capability has become popular recently, especially after the success of self-supervised learning (SSL) in computer vision tasks like object classification, segmentation, depth estimation, and image generation. These tasks, while typically tested on computer vision datasets, are also very common in robotics. The idea of using these pre-trained representations for robot learning was initially explored in [134], showing that pre-trained visual representations can sometimes even be better than using ground-truth state representations for training control policies.

Generative SSL via masked image modeling (MIM) [142, 188] has shown successful transfer of pre-trained representations from in-the-wild data to real-robot scenarios, enabling basic motor skills such as reaching, pushing, and picking. Furthermore, many other works investigate contrastive learning approaches to learning general visual representations in robotics [95, 118, 128]. These methods usually employ a pixel reconstruction objective based on a time-contrastive objective or focus on contrasting video clips leveraging natural language for video-language alignment.

The field has been moving towards finding general-purpose representations that work well across a wide range of problems in robot manipulation learning. Voltron [93], is a framework for language-driven visual representation learning for robotics that combines both masked auto-encoding and contrastive learning techniques, focusing on multi-task performance. This model is trained to learn representations that capture both low-level spatial reasoning and high-level semantic understanding by using language supervision from human videos.

Tactile sensor simulation. Multiple simulators have been proposed for vision-based tactile sensors such as [31, 65, 155, 158, 182] with the hope of sim2real generalization of learned policies [27]. However, many of these methods are either limited to marker-based tactile sensors [27], or narrow tasks [157, 219]. Certain other methods [197] also leverage simulated data to train multi-modal representations. However, in general we find that tactile simulators are still unable to model shadows, as well as real-world per-sensor-instance discrepancies, hampering their potential use for representation learning.

2.10.2 Touch representation and self-supervision details

To ensure fair evaluation of all models, our SSL algorithms are largely adapted from official MAE, I-JEPA, V-JEPA, DINO, DINOv2 codebases.

Training details

We train all models on 8 Nvidia A-100 (80G) GPUs. In addition to training losses, to monitor training progress, we rely on online probes. Specifically, we find that for joint embedding predictive architectures, the training losses are not indicative of model convergence during optimization; therefore, proxy metrics such as reconstruction quality are helpful. For all methods, we utilize DPT [143] based decoders to decode the tactile representations back into tactile images. See Figure 2.5 for some examples of tactile reconstructions from `Sparsh` embeddings. All encoder models are trained for 150 epochs. We use AdamW optimizer and use a linear rampup followed by a cosine schedule as the learning scheduler. Further, we find that tuning momentum value as well as the weight decay factor was important in observing training convergence without collapse. Additional information of hyperparameters is detailed in Table 3.2.

	Arch.	EMA decay	LR	Batch size
`Sparsh (MAE)`	ViT-B/14	N/A	1e-4	100
`Sparsh (DINO)`	ViT-B/14	0.998	1e-4	150
`Sparsh (IJEPA)`	ViT-B/14	0.996	6.25e-4	150
`Sparsh (VJEPA)`	ViT-B/14	0.996	6.25e-4	150

Table 2.1. **Training hyperparameters for `Sparsh` models.** All models run for 150 epochs with optimizer AdamW, a weight decay cosine schedule from 0.04 to 0.4, and a learning rate warmup of 30 epochs.).

	`Sparsh (MAE)`	`Sparsh (DINO)`	`Sparsh (IJEPA)`	`Sparsh (VJEPA)`
N. parameters	86254848	86255616	86386944	86537472
FPS	104	112	112	60

Table 2.2. Number of parameters and inference time for `Sparsh` backbones

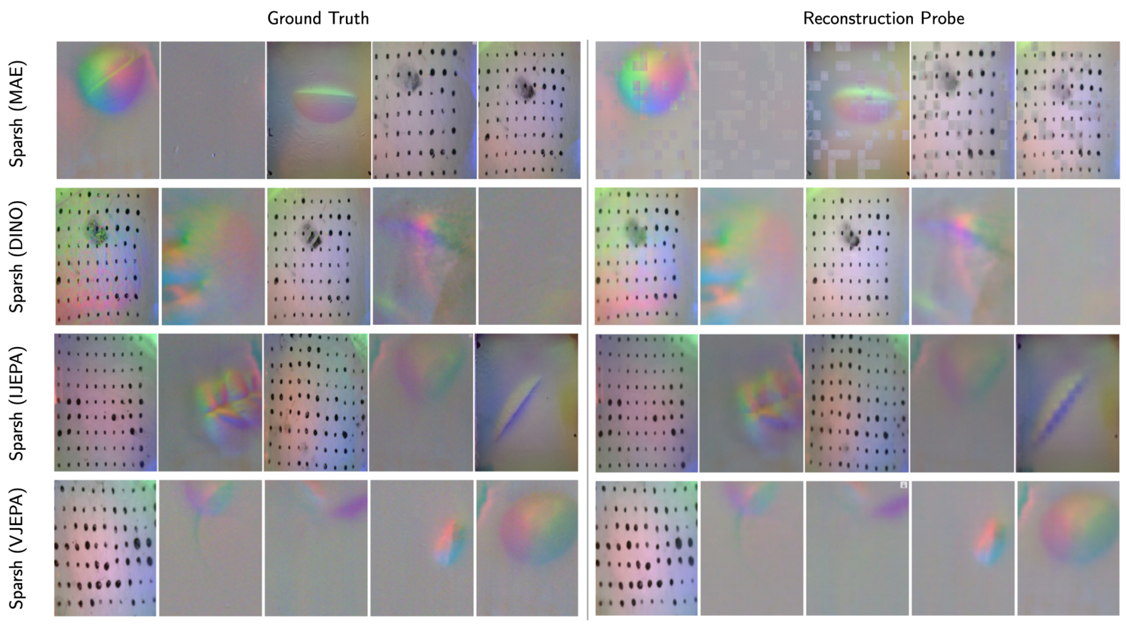


Figure 2.5. Visualization of reconstructed tactile images using the online probe to monitor SSL training of `Sparsh` models.

Architecture details

All encoder models are Vision Transformers (ViT) [46]. Although the main encoder models use ViT-B/14 as the standard architecture, following [4] we use a small ViT as the predictor network. All the models are pretrained without a `[cls]` token. For DINO, which decodes the `[cls]` token into classes, we repurpose ViT registers [38] to predict classes. In Table 2.2 we report the number of parameters for each encoder and their respective inference times.

Tactile images with a stride of 5 i.e., $\mathbf{I}_t \oplus \mathbf{I}_{t-5} \in \mathbb{R}^{h \times w \times 6}$ are concatenated along the channel dimension before the background is removed and reshaped to 224×224 for ViT processing. We choose a stride of 5 as consecutive images are similar due to high sensor sampling rates, and to match the slip detection window in humans. Ablating the effect of the input image and patch resolution may be important for better performance and is left for future work.

Dataset splits

We use three available datasets for training `Sparsh`, namely YCB-Slide [163], Touch-and-Go [196] and Object Folder [59]. The YCB-Slide dataset consist of human sliding interactions with 10 YCB objects. Each object has 5 trajectories, with around 3500

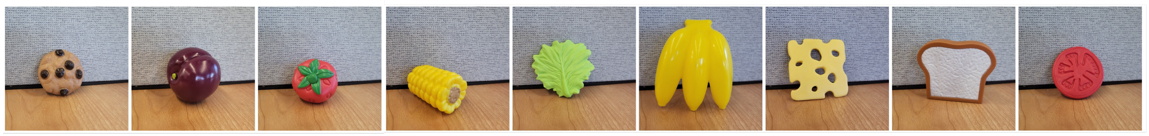


Figure 2.6. Set of objects for collecting sliding contact trajectories in the Touch-Slide dataset.

frames each from DIGIT sensors with different optical characteristics (180k frames in total). For each object, we dedicate four trajectories for training and the last one for validation. Touch-and-Go consists of discrete human contact interactions with in-the-wild objects, using a GelSight sensor. It consist of 140 videoclips and plain files with labels for the frames with a clear contact. We use all frames (220k) in the videoclips since we do not rely on labeled data for SSL training, from which 70% is used for training and the remaining for validation. The data used from ObjectFolder consist of 81k frames of robot discrete contact interactions with objects in a controlled setting. We also use a train/val split of 70/30.

To complement the dataset, we collected Touch-Slide with additional human sliding interactions on toy-kitchen objects with the DIGIT sensor. We use 9 objects, shown in Figure 2.6 and collected 5 trajectories for each, generating 180k frames in total.

For all downstream tasks we use tactile data from real sensors/hardware (DIGIT, GelSight17, and GelSight Mini) that were not seen during Sparsh SSL training. Under our problem formulation, this allows us to investigate generalization of `Sparsh` to new sensor instances (consider the case of swapping out a sensor from a robotic hand due to wear-off).

Similarly, all objects used for downstream tasks were not used for SSL training. For example, `[T1]` and `[T2]` tasks use a real robot arm to slide the sensor elastomer (DIGIT, GelSight Mini) against an indenter to collect force-labeled data. `[T4]` uses an open source dataset for grasp stability [4], which includes data from a real robot grasping over 100 unique objects using an unseen GelSight17 with printed markers during SSL training. Similarly, `[T6]` uses DIGIT data collected from real hardware, a robot pulling and moving a bead along the wire. Note that none of the data used for learning representations comes from this kind of object-robot hand interactions.

Short summary of SSL methods

In this paper, we consider three SSL paradigms, namely `Sparsh` (`MAE`), `Sparsh` (`DINO`), and `Sparsh` (`IJEPA`) & `Sparsh` (`VJEPA`).

`Sparsh (MAE)` is based on the principle of masked image modeling, where an encoder model is tasked with learning the contextual representations of substantially masked images, such that it enables reconstruction of the masked regions via a lightweight decoder. We use a ViT encoder and decoder for Sparsh, and the MAE loss corresponds simply to a L2 reconstruction loss:

$$\mathcal{L}_{\text{MAE}} = \|\mathbf{I}_{\text{target}} - \mathbf{I}_{\text{recon}}\|_2^2 \tag{2.1}$$

`Sparsh (DINO)` is based on the principle of self-distillation between two identical networks, where a student network learns to track the output predictions of a EMA teacher network. Cross-entropy loss is employed between the predictions of the student and teacher network, both of which consume different crops of the same input. Specifically, feature representations from each branch are passed through a MLP head, producing probability vector over an arbitrarily chosen number of classes. These scores are normalized to produce $\mathbf{p}_s$ and $\mathbf{p}_t$ for the student and teacher respectively.

$$\mathcal{L}_{\text{DiNO}} = -\sum \mathbf{p}_t \log \mathbf{p}_s \tag{2.2}$$

`Sparsh (IJEPA)` and `Sparsh (VJEPA)` share similarities with both masked image modeling and self-distillation. Here, we employ two identical networks termed context and target networks. The context network corresponds to a student network, which is tasked to predict the features from a EMA target network (teacher network), through a small predictor network. In this case, a L_2 loss over features is used to enforce similarity between the two branches. Specifically, the context network observes M global masks of an image to produce contextual features, which are then passed through a predictor to predict target network features of B_i local crops of the same image $\hat{\mathbf{s}}_{y_j}$. On the other hand, the target network consumes local crops of the image to produce $\mathbf{s}_{y_j}$.

$$\mathcal{L}_{\text{jepa}} = \sum_{i \in M} \sum_{j \in B_i} \|\hat{\mathbf{s}}_{y_j} - \mathbf{s}_{y_j}\|_2^2 \tag{2.3}$$

Task	Dataset	Sensor	Size	Collector	Label
[T1] Force estimation	Shear load	DIGIT	75k	Robot	3-axis force
	(indenter: sphere, flat, sharp)	GelSight Mini	75k	Robot	3-axis force
[T2] Slip detection	Shear load (indenter: sphere)	DIGIT	125k	Robot	Friction cone
[T3] Pose estimation	Object sliding	DIGIT	49k	Human	Object pose **SE**(2)
[T4] Grasp stability	Feeling of Success	GelSight 2017	9.3k	Robot	Success (yes/no)
[T5] Textile recognition	Clothing Dataset	GelSight 2017	120k	Robot	Textile ID
[T6] Bead maze	Demonstrations	DIGIT	34k	Human	Joint angles

Table 2.3. Datasets in TacBench for evaluating representations on downstream tasks.

2.10.3 TacBench tasks and evaluation details

Labeled datasets

See Table 3.3 for details on labeled data curation for TacBench tasks.

Probe details

The parameters of the model updated via EMA (target encoder for Sparsh (IJEPA) and Sparsh (VJEPA), teacher network for Sparsh (DINO) and Sparsh (DINOv2), encoder from Sparsh (MAE)) are fixed and used for evaluation. The features are pooled via attentive pooling for tasks that require global representations, such as slip detection, resultant force estimation, and classification tasks. For tasks that require dense reasoning, we use DPT decoders [143] to decode patch representations into full input resolution quantities such as normal and shear force fields, and reconstructed tactile images. See Figure 2.7 for a visual illustration of the probe architectures.

We follow attentive probing[11, 28] to assess the capabilities of tactile representations on the benchmark, as this approach allows us to determine what representations capture from self-supervision alone. For most tasks – except force field visualization and policy learning – in the benchmark, we freeze Sparsh and train a cross-attention module (hyperparameters in Table 2.4) followed by a light 2-layer MLP probe supervised, using the labeled dataset for each task.

[T1] Force estimation

After attentive pooling, the tactile features with 768 dimensions are passed to a 2-layer MLP with 192 and 3 units respectively, to get the 3-axis force estimations. Two independent force decoders are trained using DIGIT and GelSight Mini data

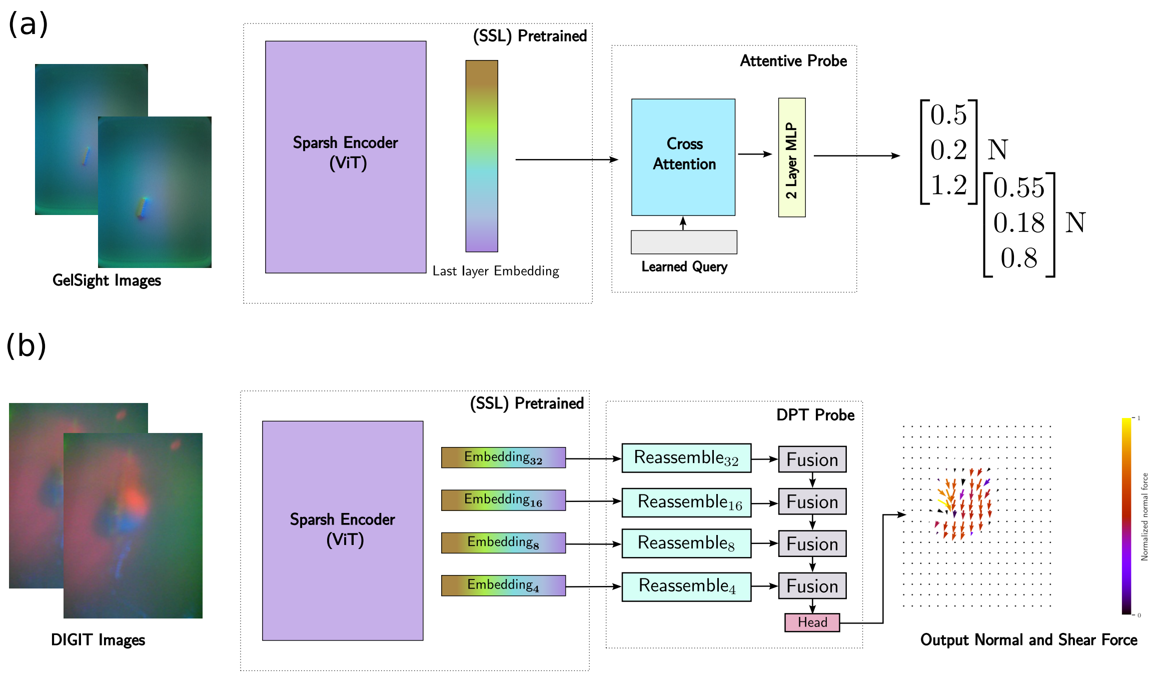


Figure 2.7. (a) Attentive probe architecture consists of a cross attention layer followed by a linear layer to regress *resultant* output quantities such as resultant force or slip state (b) Dense Prediction Transformer (DPT) [143] consists of multiple reassemble and fusion layers to decode features from intermediate layers of the Sparsh backbones to produce dense outputs such as normal and shear fields

respectively, using the sharp and sphere probe data during training and the flat indenter data for testing. The target forces are normalized to be ± 1.0 and scaled back after prediction. We train the force decoder using Adam optimizer with 1e-4 learning rate.

DIGIT. In Table 2.5 we report the average RMSE over 25k samples of unseen DIGIT data for the force estimation task. We report metrics for each Sparsh model and the E2E approach, under four different budgets of training data. We also provide a 95% confidence interval to ground the error ranges of each model.

In Figure 2.8 we plot the friction cone from the test data, where the colormap represents the error in mN for each axis. Note that E2E exhibit larger errors (around 500mN) for the tangential component and they are more predominant as the normal force increases. In contrast, the top model Sparsh (DINOv2) estimates with low error ($<$ 100mN) in general across the whole range of tangential and normal forces.

GelSight. In Table 2.6 we report the average RMSE over 25k samples of unseen

Parameter	Setting
Embedding dimension	768
N heads	12
MLP ratio	4.0
Depth	1
Layer normalization	Yes

Table 2.4. Attentive pooling hyperparameters used for evaluation protocol of representation in downstream tasks.

Model	Full dataset (50k)	1/3 dataset	1/10 dataset	1/100 dataset
`E2E`	39.34 [39.21, 39.48]	61.42 [61.12, 61.72]	98.22 [97.61, 98.84]	187.51 [185.51, 188.51]
`Sparsh (MAE)`	36.61 [36.51, 36.71]	45.96 [45.80, 46.12]	58.55 [58.31, 58.79]	115.39 [114.69, 116.09]
`Sparsh (DINO)`	36.09 [36.01, 36.17]	44.03 [43.87, 44.19]	51.89 [51.69, 52.10]	**97.95 [97.36, 98.52]**
`Sparsh (DINOv2)`	**29.31 [29.14, 29.46]**	**26.85 [26.70, 26.99]**	**37.66 [37.45, 37.86]**	185.86 [184.94, 186.78]
`Sparsh (IJEPA)`	40.27 [40.16, 40.38]	60.04 [59.72, 60.34]	86.57 [86.06, 87.08]	130.37 [129.59, 131.15]
`Sparsh (VJEPA)`	39.38 [39.30, 39.47]	56.34 [56.07, 56.62]	76.11 [75.67, 76.55]	130.83 [130.29, 131.38]

Table 2.5. Root Mean Squared Error (mN) and 95% confidence interval for force estimation with DIGIT data. All models were evaluated on flat indenter data over 25k test samples.

GelSight data and the corresponding 95% confidence interval. Notice from Figure 2.9 that the majority of errors are localized around the dynamic shear region. It is worth noting that the errors associated with `Sparsh` (`DINO`) remain below 150mN, whereas `E2E` exhibits higher errors, particularly in the estimation of normal forces.

`[T1A]` Force field visualization

Since rendering the force field is a dense prediction task, we do not apply the attentive probing protocol. Instead, we follow DPT [144], training a CNN encoder with reassemble-fusion modules at layers 2,5,8,11 of the `Sparsh` encoder to progressively upsample the representations to obtain a fine-grained prediction of the force field. After the reassemble-fusion modules, we attach two task-specific task head, for normal and shear field prediction.

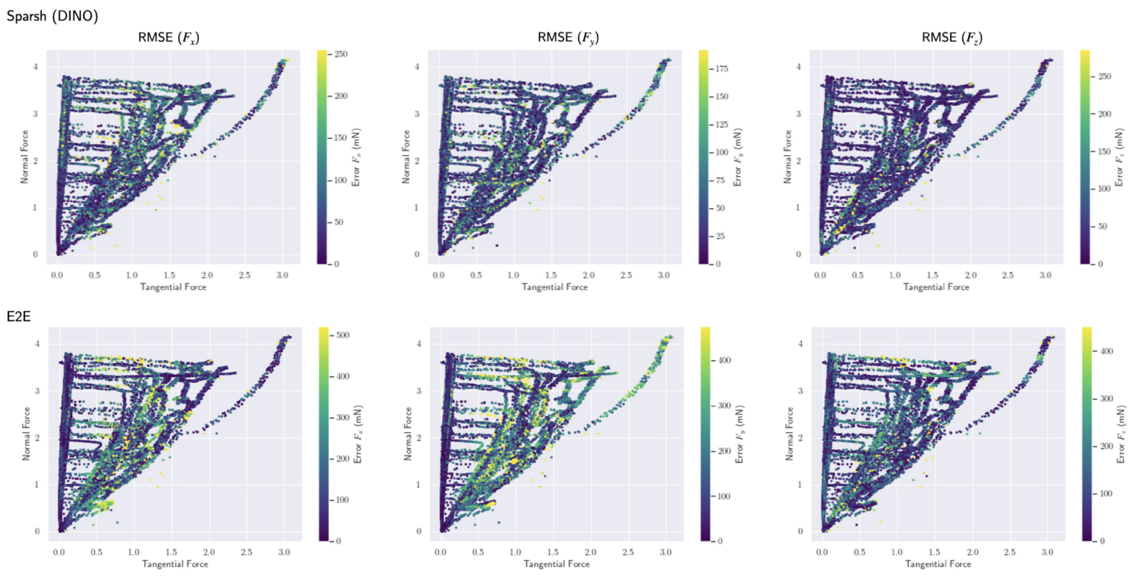


Figure 2.8. Friction cone of test data and RMSE (mN) for force estimation task with DIGIT sensor.

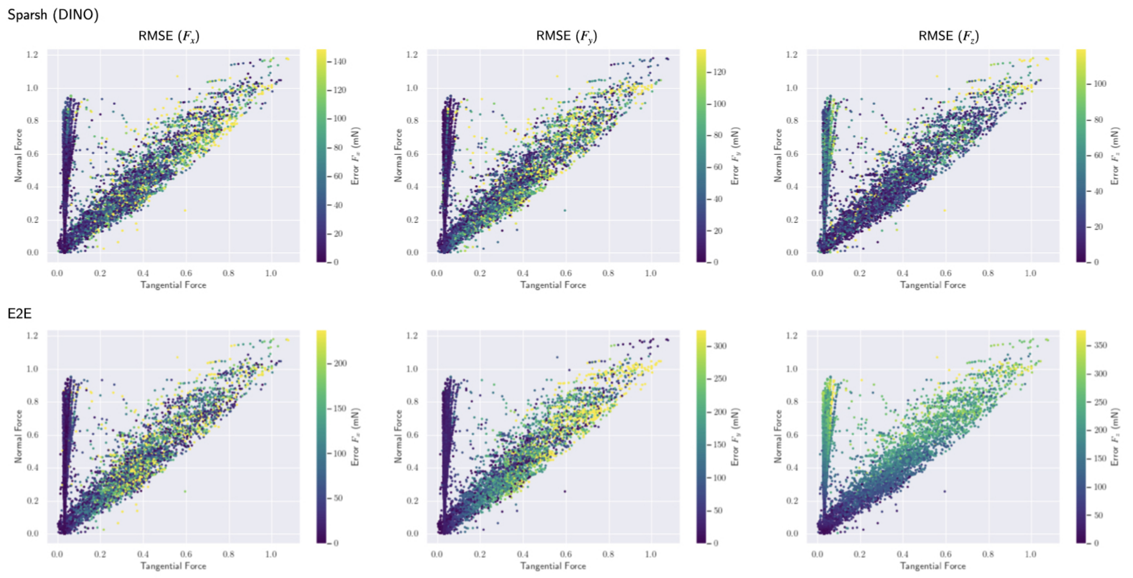


Figure 2.9. Friction cone of test data and RMSE (mN) for force estimation task with GelSight sensor.

Since for markerless vision-based sensors it is not trivial to get ground truth of the force field, we turn to unsupervised learning. Depth estimation and optical flow are analogous to the estimation of normal and shear force fields, areas where the computer vision community has proposed several unsupervised methodologies [64, 90, 91, 112,

Model	Full dataset	1/3 dataset	1/10 dataset	1/100 dataset
E2E	57.21 [56.44, 57.98]	59.09 [58.15, 60.04]	57.43 [56.44, 58.42]	82.42 [80.98, 83.86]
Sparsh (MAE)	22.72 [22.27, 23.17]	**23.28** [22.83, 23.72]	33.56 [33.04, 34.08]	78.98 [77.74, 80.21]
Sparsh (DINO)	**20.25** [19.85, 20.65]	23.79 [23.40, 24.18]	**32.17** [31.67, 32.67]	**53.43** [52.69, 54.17]
Sparsh (DINOv2)	37.30 [36.71, 37.88]	37.79 [37.22, 38.37]	45.86 [45.14, 46.59]	105.95 [104.28, 107.62]
Sparsh (IJEPA)	27.91 [27.37, 28.44]	35.20 [24.57, 35.82]	44.93 [44.13, 45.73]	91.81 [90.76, 92.86]
Sparsh (VJEPA)	33.26 [32.67, 33.84]	34.07 [33.39, 34.75]	42.35 [41.60, 43.10]	80.36 [79.26, 81.47]

Table 2.6. Root Mean Squared Error (mN) and 95% confidence interval for force estimation with GelSight Mini data. All models were evaluated on flat indenter data over 25k test samples.

161]. We borrow ideas of unsupervised monocular depth estimation, where from two tactile images I_t and I_{t-n}, we learn a pose estimator for getting the transform between frames. With the sensor intrinsic K, we map image I_t from pixel space to camera plane, translate estimated depth D_t, apply transform from t to $t-n$, and transform back to image plane to get $\hat{I}_{t-n}$. We supervised based on the reprojection error, MSE between I_{t-n} and predicted $\hat{I}_{t-n}$. To reconstruct the shear field, we transfer ideas from unsupervised optical flow, where we warp the features of image I_t to I_{t-n} based on the estimated flow and compute a photometric consistency loss that encourages the estimated flow(shear) to align image patches with a similar appearance. This loss is a linear combination of the Charbonnier loss and the structural similarity (SSIM) between I_{t-n} and $\hat{I}_{t-n}$. We also add a smoothness loss that acts as a regularization term, encouraging the shear field to align the boundaries with the visual edges in the tactile image. In Figure 2.10 we show snapshots of the normal and shear field predictions during sliding trajectories of the DIGIT sensor on YCB and spherical probe objects.

[T2] Slip detection

To collect labeled slip data we perform a normal/shear load test. Using a firmly affixed hemispherical probe on a flat surface, a robot presses the DIGIT sensor toward

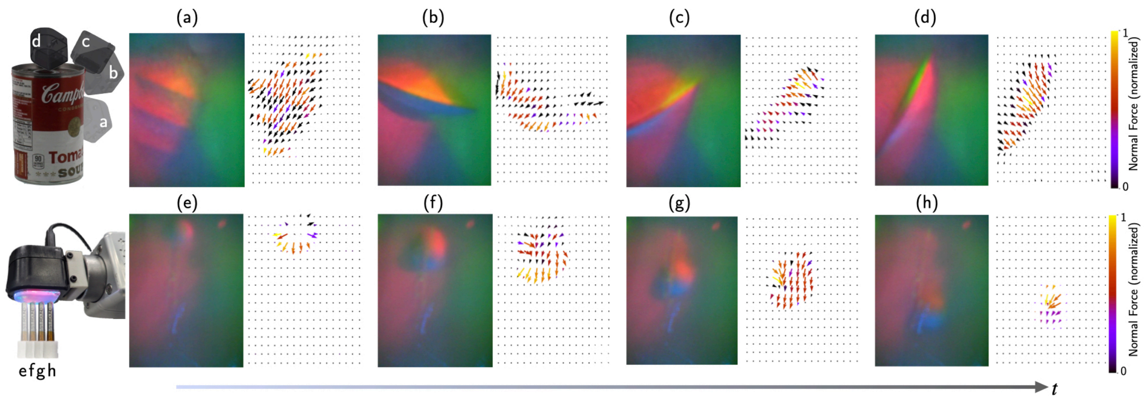


Figure 2.10. Normalized tactile flow (unitless) visualizations using `Sparsh` (`DINO`). Top row shows predicted force field for four key-frames from a representative YCB-Slide trajectory and bottom row shows interaction with the spherical probe. Arrows represent the tangential forces, while the colors depict the normal forces. These visualizations provide directional information about the relative motion of the contact patch. For instance (a) shows torsional motion resulting from rotating along the edge, (b, c, d) show sliding on the edge, (e) shows a diverging field when making contact with a spherical probe, and (f, g, h) show forces produced by sliding the probe top-down.

the probe, applying random normal forces of up to 5N. Upon reaching the target normal force, the robot slides the probe 2mm to a randomly selected position on the sensor surface, allowing us to capture the shear profile with a F/T sensor. To label slip, we rely on the friction cone to identify samples on the sticking and slippage regions. A description of the procedure is illustrated in Figure 2.11.

As eluded to in Section 5.4, `Sparsh`'s inference window is approximately 80 milliseconds. This is appropriate since this duration matches the reaction time needed by humans to adjust the grip force when detecting partial slip [213]. We train two heads: one for slip detection and the other for the estimation of normalized force change (Δ). We find empirically that training both heads simultaneously improves slip detection, given their high correlation. The MLP probes are trained with cross-entropy for slip detection and mean absolute error for Δ force regression as loss functions. Our dataset comprises 125k samples, with only 13% corresponding to slip instances. We reserve 25k samples for evaluating model performance.

Table 2.7 provides F1-score metrics for all models under different amounts of training data. `Sparsh` (`VJEPA`) outperforms all models, even when trained under low data regimes. In Figure 2.12 we contrast the predictions over time for a sample

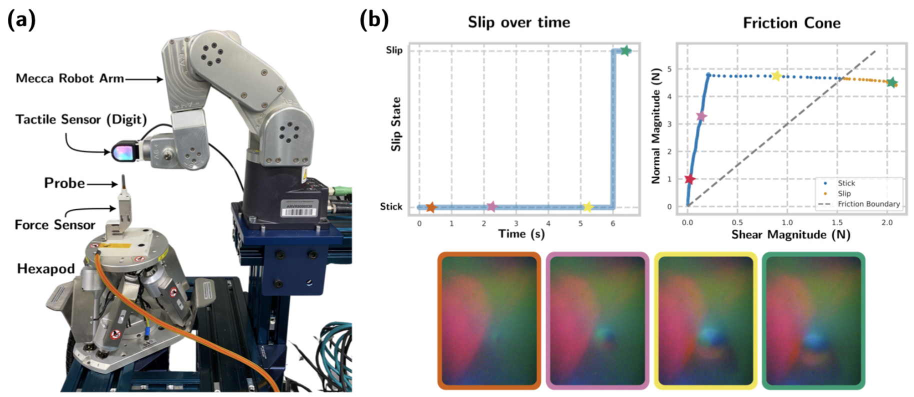


Figure 2.11. (a) Data collection setup for [**T1**] **Force Estimation** and [**T2**] **Slip Detection**. The Mecca Robot Arm with DIGIT / Gelsight is pressed against a static probe with random normal force. The arm then slides the sensor over the probe which induces shear forces. (b) Slip states over one representative stroke. When the sensor is pressed against the probe the normal force increases. The gel sensor initially resists sliding due to friction, but gives in, which results in a slight drop in normal force while the magnitude of shear force increases.

trajectory between `Sparsh` (`VJEPA`) and `E2E` models trained with 33% of the data. Note that for `Sparsh` (`VJEPA`) the errors are around the friction boundary, where the probe is starting to slide. Also, it is worth noticing that a poor estimation of changes in shear and normal forces is reflected in the accuracy of distinguishing between slip and no-slip. In Figure 2.13, we illustrate a failure case for `Sparsh` (`VJEPA`), as its results do not align with the ground truth. However, it is important to note that slip labeling is prone to errors due to its reliance on an experimental coefficient of friction. Despite the inaccuracies in the friction boundary for this trajectory, `Sparsh` (`VJEPA`) successfully detects the slip samples.

Model	Full dataset	1/3 dataset	1/10 dataset	1/100 dataset
`E2E`	0.767	0.238	0.299	0.214
`Sparsh` (`MAE`)	0.783	0.818	0.691	0.269
`Sparsh` (`DINO`)	0.685	0.561	0.548	0.489
`Sparsh` (`DINOv2`)	0.687	0.601	0.561	0.243
`Sparsh` (`IJEPA`)	0.776	0.791	0.775	0.726
`Sparsh` (`VJEPA`)	**0.820**	**0.828**	**0.800**	**0.760**

Table 2.7. Performance of models on slip detection task under different budgets of training data. We use F1 score as metric, given that it ensures the model accurately identifies slip events without favoring the majority class. A high F1 score indicates effective and reliable slip detection.

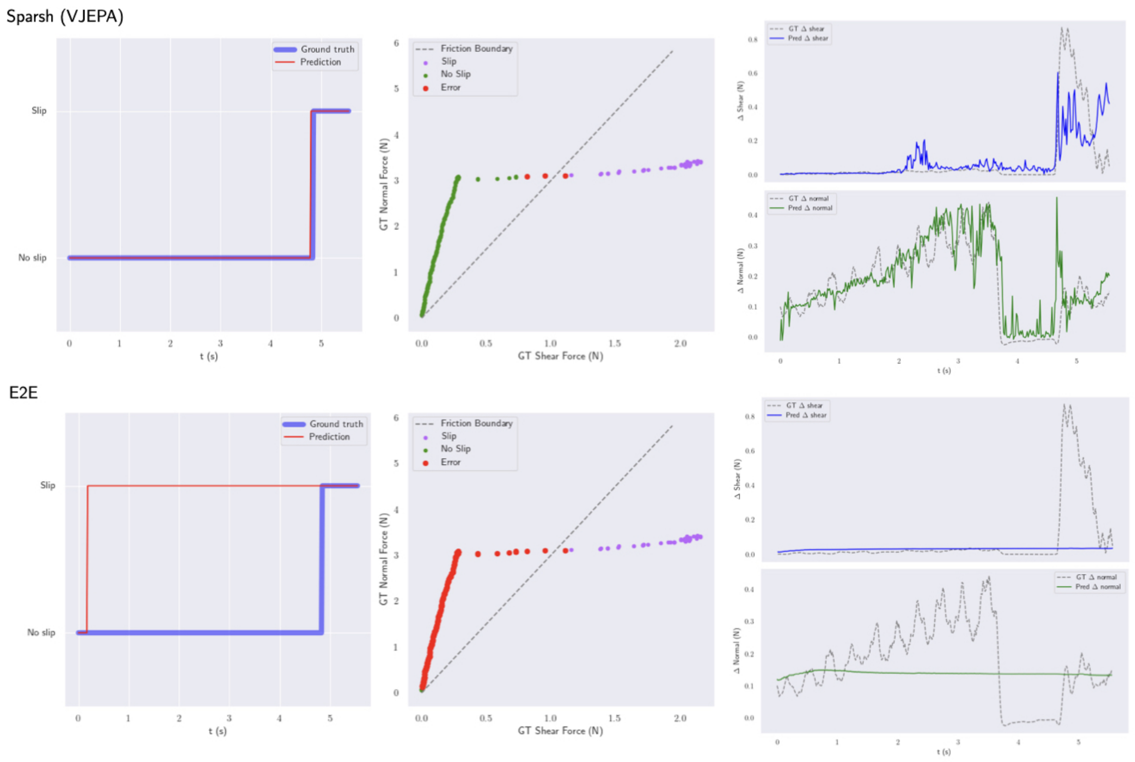


Figure 2.12. Contrast between `Sparsh` (`VJEPA`) and `E2E` for a test trajectory with a spherical probe sliding on the DIGIT sensor. `Sparsh` (`VJEPA`), even though trained only on 33% of the data, can detect slip accurately, which is correlated with its ability to estimate changes in normal and shear forces.

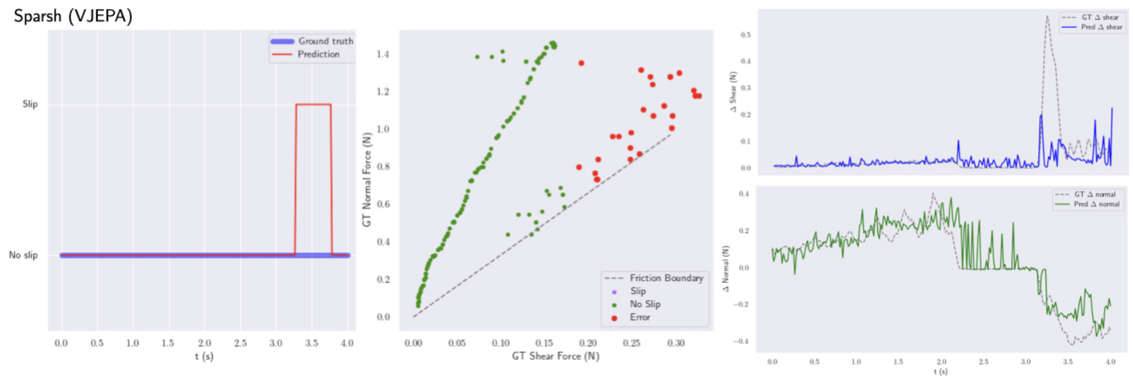


Figure 2.13. Failure case where the ground truth does not reflect slip since it relies on an experimental coefficient of friction. Despite the inaccuracies in the friction boundary for this trajectory, `Sparsh` (`VJEPA`) successfully detects slip samples.

[T3] Pose estimation

We collect a dataset of trajectories with time-synchronized pairs of object pose measurements and sensor observations using an Allegro hand equipped with DIGIT sensors on each finger, mounted on a robot arm. The object was placed on a table and with the palm facing downward, we pressed against it with the fingertips (see Figure 2.3). We manually perturbed the object's pose by sliding and rotating it under the Allegro fingertips. The pose of the object was tracked using ArUco tags. Given ground truth object pose measurements in the world frame, we preprocess them into relative pose change $(\Delta x, \Delta y, \Delta\theta) \in \mathrm{SE}(2)$ in the sensor frame.

Since we follow a regression-by-classification approach, we discretize the range of motion for each degree of freedom into multiple intervals in Log-uniform space. This allows us to achieve a better data distribution across all classes, as most pose changes are concentrated around zero. The strategy of classification-regression is also commonly explored for monocular depth estimation [151].

After attentive pooling, the features are passed to three heads, one for each degree of freedom. Each head is an MLP with two layers, which outputs the probability distribution over 11 classes (pose change bins). In Figure 2.14 we present the binning as well as the confusion matrices on test data for each degree of freedom, comparing `E2E`, `Sparsh` (`DINO`) and `Sparsh` (`IJEPA`) for pose estimation when trained on 33% of the available labeled data. Note that `Sparsh` can accurate distinguish pose changes in a low data regime, while a conventional task-specific approach struggles discerning the differences between adjacent bins, and finally tends to default to zero or maximum relative pose change, losing resolution in estimation.

Figure 2.15 shows a test trajectory over time with its ground truth labels. The colors on the plot represent the class agreement between the pose decoders trained with `Sparsh` (`DINO`) (using 33% of the data) and the ground truth. Darker colors indicate no error, while brighter colors indicate greater misclassification. In Table 2.8 we report for each model accuracy in pose estimation over 630 test samples and 95% confidence interval.

[T4] Grasp stability

We use the Feeling of Success dataset [19], which contains data from a pair of GelSight sensors (with markers) attached to a jaw gripper (left and right fingers). The goal is

Model	Full dataset	1/3 dataset	1/10 dataset	1/100 dataset
`E2E`	0.812 [0.811, 0.813]	0.245 [0.244, 0.247]	0.162 [0.160, 0.164]	0.162 [0.160, 0.164]
`Sparsh (MAE)`	0.896 [0.896, 0.897]	0.719 [0.718, 0.721]	0.417 [0.414, 0.420]	0.223 [0.221, 0.225]
`Sparsh (DINO)`	**0.913** **[0.912, 0.914]**	**0.834** **[0.832, 0.836]**	**0.460** **[0.457, 0.461]**	**0.242** **[0.240, 0.245]**
`Sparsh (DINOv2)`	0.665 [0.658, 0.673]	0.565 [0.559, 0.570]	0.411 [0.408, 0.415]	0.210 [0.209, 0.211]
`Sparsh (IJEPA)`	0.851 [0.850, 0.852]	0.601 [0.599, 0.603]	0.323 [0.321, 0.325]	0.212 [0.210, 0.215]
`Sparsh (VJEPA)`	0.856 [0.854, 0.857]	0.648 [0.646, 0.651]	0.368 [0.367, 0.370]	0.228 [0.225, 0.231]

Table 2.8. Accuracy and 95% confidence interval for pose estimation task following the regression-by-classification paradigm. Relative pose between object and ring finger. Metrics computed over 630 test samples.

to determine the success or the failure of the grasp attempt.

We pass to the SSL model the 'before' and 'during' as tactile history. We create our randomized split with all objects, using approximately 8k grasps for training and the remaining 1.3k grasps for evaluation. Using attentive probing, we freeze `Sparsh` and train a 2-layer MLP with two output units for grasp success classification.

In Table 2.9 report the accuracy for binary classification to compare the performance of the models across different training budgets, including a 95% confidence interval. Figure 2.16 shows the confusion matrices on test samples for `E2E`, `Sparsh` (`DINO`) and `Sparsh` (`IJEPA`) trained on a 33% of labeled data.

`[T5]` Textile recognition

This tasks allows to study the capabilities of the representations for semantic understanding of the contact, as in recognizing the type of textile that is being touched by the sensors. We use the task definition and the data set introduced in [209]. This data set contains 4467 short video clips (10-25 frames), of a robot with a GelSight (markers) mounted parallel gripper grasping several types of clothing, across 20 textile classes, such as leather, cotton, polyester, etc.

We follow the train-test split provided in the metadata of the dataset. Using

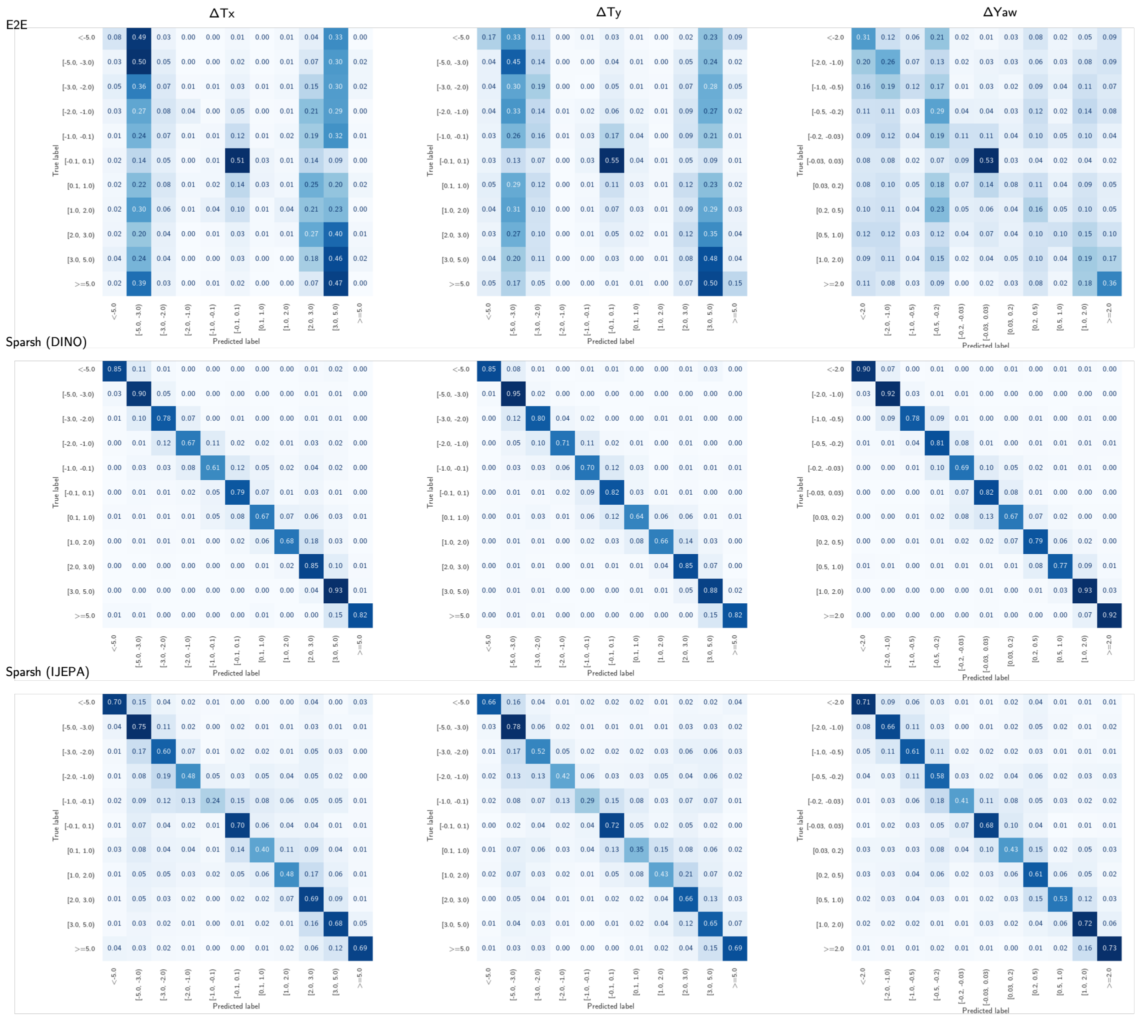


Figure 2.14. Confusion matrix on test data for ΔT_x, ΔT_y, ΔYaw for `E2E`, `Sparsh` (`DINO`) and `Sparsh` (`IJEPA`) trained on 33% of the available labeled data. The test dataset consist of 630 samples.

attentive probing, we freeze `Sparsh` and train a 2-layer MLP with 20 output units for textile classification. In Table 2.10 and Figure 2.19(c) we report the accuracy for multiclass classification, comparing the performance of the models in different training budgets.

`[T6]` Bead maze

The goal in bead maze is to guide the bead along the wire, as shown in Figure 2.3. We don't rely on vision for hand-eye coordination, making the task fundamentally tactile since forces in the fingers indicate whether the bead is moving smoothly or

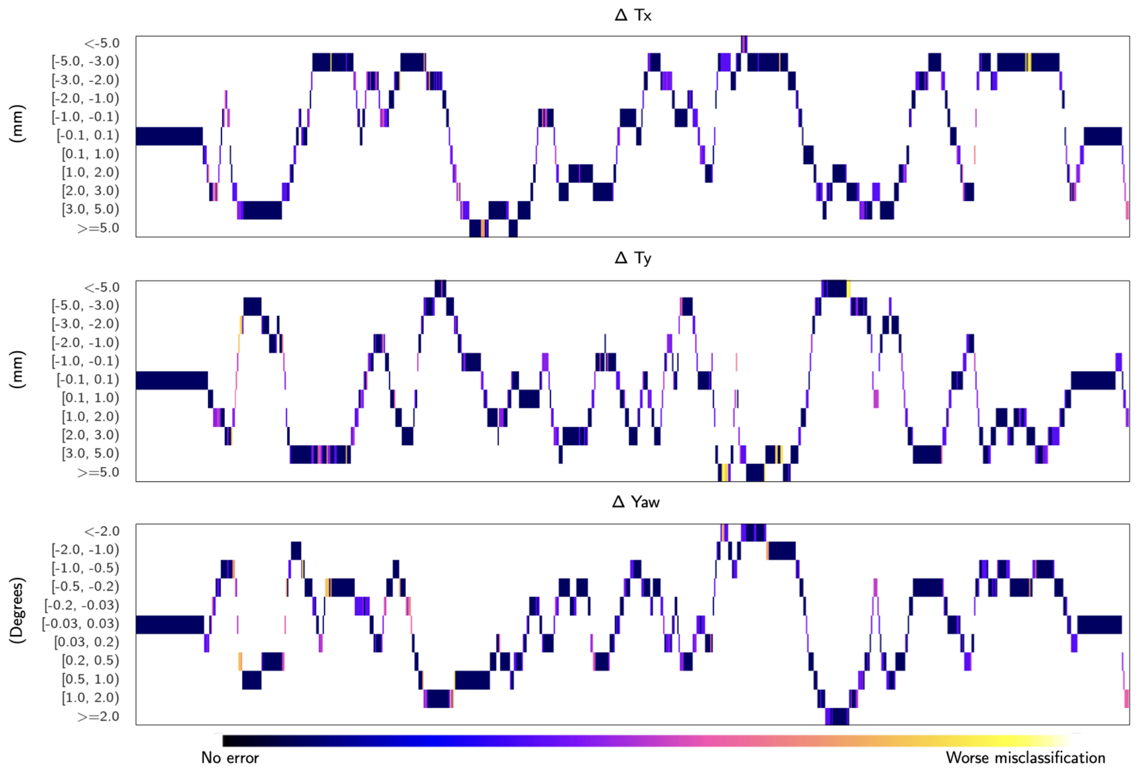


Figure 2.15. Ground truth relative pose classes for T_x, T_y, and Yaw for a test trajectory. The colormap represents the class agreements between the ground truth and the pose decoder, with darker colors indicating no error and brighter colors indicating greater misclassification.

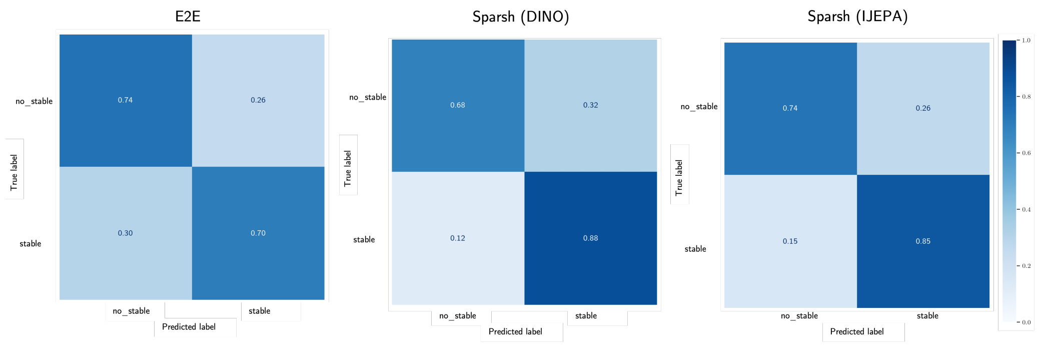


Figure 2.16. Confusion matrix on test data for grasp stability, comparing `E2E`, `Sparsh` (`DINO`) and `Sparsh` (`IJEPA`) trained on 33% of the available labeled data. The test dataset consist of 1.3k grasps.

encountering resistance. In our setup, we use a Franka arm with a robotic hand mounted on the wrist and DIGIT sensors on the fingers. To collect demonstrations for training the policy, we start the task with the bead grasped between the thumb

Model	Full dataset	1/3 dataset	1/10 dataset	1/100 dataset
`E2E`	0.784 [0.783, 0.785]	0.725 [0.722, 0.728]	0.682 [0.680, 0.684]	0.478 [0.472, 0.482]
`Sparsh (MAE)`	0.815 [0.813, 0.817]	0.696 [0.691, 0.702]	0.764 [0.761, 0.768]	0.466 [0.461, 0.471]
`Sparsh (DINO)`	0.780 [0.777, 0.782]	0.706 [0.702, 0.710]	0.773 [0.772, 0.775]	0.473 [0.467, 0.479]
`Sparsh (DINOv2)`	0.770 [0.767, 0.771]	0.770 [0.768, 0.772]	0.699 [0.697, 0.701]	0.543 [0.539, 0.546]
`Sparsh (IJEPA)`	0.802 [0.800, 0.804]	**0.782** **[0.779, 0.784]**	**0.768** **[0.766, 0.770]**	**0.598** **[0.597, 0.601]**
`Sparsh (VJEPA)`	**0.809** **[0.805, 0.813]**	0.702 [0.700, 0.704]	0.743 [0.740, 0.746]	0.523 [0.519, 0.527]

Table 2.9. Accuracy and 95% confidence interval for grasp stability classification over different budget sizes of training data, using Feeling of Success dataset. Results over 1.3k grasps.

and index fingers and move the arm to guide the bead along the wire. We collect 30 demonstrations on different maze patterns with mix of VR-based and manual kinesthetic-based teleoperation, corresponding to a total of ∼34k training pairs of tactile images and robot joint angles.

For training the policy, we adapt Diffusion Policy [32] to our problem setting. Given a small history of tactile images $(\ldots, \mathbf{z}_{t-1}, \mathbf{z}_t)$, and robot proprioception $(\ldots, q_{t-1}, q_t)$, we train the policy to predict changes in joint angles as actions $\mathbf{a} \triangleq (\Delta q_t, \Delta q_{t+1}, \ldots); \Delta q \in \mathbb{R}^7$, instead of position control. Following the guidelines in Diffusion Policy, we use an observation horizon of 2 and an action prediction horizon of 8. We adhere to the official implementation for policy architecture and training hyper-parameters. For conditioning on tactile input, we modify the CNN encoder from Diffusion Policy and replace it with `Sparsh` backbones with fixed parameters. For training an end-to-end policy, the encoder corresponds to a ViT-Base encoder with randomly initialized weights.

For each method, we evaluate the learned policies over a set of 10 randomized novel starting locations on the maze and we measure distance traversed (in cm) before failure. In Table 2.11, we report mean and variance of distance traversed comparing `Sparsh` (pre-trained only and pre-trained then fully fine-tuned) against

Model	Full dataset	1/3 dataset	1/10 dataset	1/100 dataset
E2E	0.437	0.365	0.373	0.171
Sparsh (MAE)	**0.599**	**0.588**	**0.527**	**0.330**
Sparsh (DINO)	0.527	0.520	0.463	0.264
Sparsh (DINOv2)	0.544	0.536	0.469	0.288
Sparsh (IJEPA)	0.506	0.478	0.399	0.217
Sparsh (VJEPA)	0.580	0.545	0.507	0.285

Table 2.10. Accuracy for textile classification over 20 classes using GelSight with markers dataset under different budget of labeled data. Results over 26k tactile images, where accuracy of chance is 0.05.

E2E. All models use 50 demonstrations for training the policy via imitation learning. We find that policies using Sparsh representations outperform E2E by $\sim 20 - 53\%$. Most failure cases across methods are due to the bead getting stuck on the maze or the bead falling out of the robot hand. While prior work such as diffusion policy suggest that frozen pre-trained models may hurt imitation learning due to domain mismatch, we do not observe significant gains from fine-tuning in this application. Leveraging pre-trained models in imitation learning is an active area of research, however these results demonstrate the impact of Sparsh touch representations for robot applications.

(cm)	Sparsh (DINO)	Sparsh (IJEPA)	Sparsh (MAE)	E2E
Pre-trained	10.80 ± 3.68	9.4 ± 3.1	10.2 ± 4.9	6.70 ± 1.67
Fine-tuned	8.45 ± 3.21	10.02 ± 5.37	11.25 ± 3.85	6.70 ± 1.67

Table 2.11. Mean and variance of distance traversed (in cm) before failure for policies based on Sparsh and E2E. Results over 10 randomized novel starting locations on the bead maze.

In Table 2.12 we report to position error of E2E, Sparsh (DINO) and Sparsh (IJEPA) with respect to test demonstrations on an unseen maze, highlighting the fidelity of Sparsh (DINO) and Sparsh (IJEPA) to follow a similar trajectory. Nevertheless, this doesn't necessarily transfer to real-world performance, since the locality of the observations and predictions make the errors in the adjusted joint angles to compound fast, which results in unforeseen collisions and the subsequent lose of the grasp. In an overfitting setting, training a policy for a single maze, policies using Sparsh (DINO) and Sparsh (IJEPA) are able to complete almost 30% of the maze on the real robot. However, it is expected an specialist policy trained end-to-end to perform better in the overfitting setting. Experimentally, we found than an E2E policy trained for a single maze is able to complete almost 80% of the maze running on the real robot.

In Table 2.13 we summarize the performance of Sparsh across the benchmark. We find that with respect to an E2E approach, with Sparsh we can achieve an improvement of 98.75% on average. Sparsh (DINO) and Sparsh (IJEPA) are in general the best models across the board, showing the benefits of learning touch representations in latent space. Sparsh (MAE), which relies on pixel space supervision, is still competitive, although it was not evaluated on the policy task.

Model	Full dataset	1/2 dataset	1/10 dataset
Sparsh-(E2E)	8.46 [7.61, 9.32]	7.14 [6.26, 8.05]	9.80 [8.78, 10.82]
Sparsh-(DINO)	5.54 [4.90, 6.17]	5.98 [5.29, 6.67]	5.71 [5.13,6.29]
Sparsh-(IJEPA)	5.47 [4.82, 6.13]	5.72 [5.05, 6.40]	5.46 [4.82, 6.10]

Table 2.12. Position error (mm) and 95% confidence interval for the Bead Maze task. We compare the ground truth trajectory from a test demonstration in an unseen maze against the compounded trajectory from the predicted delta joint angles from each policy.

Task	Best SSL vs E2E	DINO vs IJEPA	MAE vs Best	VJEPA vs Best
Force estimation (DIGIT)	28.31%	26.67%	−4.38%	−27.96%
Force estimation (GelSight)	59.74%	32.41%	1.72%	−64.23%
Slip detection	242.70%	29.08%	−1.21%	0.00%
Pose estimation	235.89%	−37.91%	−13.81%	−22.33%
Grasp stability	5.14%	8.45%	−10/17%	−7.83%
Bead maze	19.72%	−5.26%	-	-
Average	**98.75**%	8.91%	−5.57%	−24.47%

Table 2.13. Performance of Sparsh across TacBench and comparison between SSL approaches.

2.10.4 Sparsh ablations

TacBench evaluations via fine-tuning

Fine-tuning the Sparsh encoders is another method of assessing the quality of pre-trained representations. Fine-tuning can potentially enhance performance in downstream tasks when the pre-trained model lacks task-relevant information.

We evaluated both the full and partial fine-tuning of `Sparsh` on `TacBench`. In full fine-tuning, all encoder parameters are updated through task supervision. In partial fine-tuning, we update only the last transformer block of the encoder. Figure 2.19 shows the fine-tuning results in the benchmark with varying amounts of labeled data. Notably, models pre-trained in latent space (DINO, I-JEPA, V-JEPA) perform better in downstream tasks when fully fine-tuned, especially in regression tasks like force and pose estimation. For example, Figure 2.19(a) illustrates that errors in force estimation are significantly lower with full fine-tuning, even with only 33% and 10% labeled data. Full fine-tuning also enhances performance in classification tasks such as slip detection, grasp stability, and textile classification, as shown in Figures 2.19(b,c). Adding in-domain data to the encoder reduces performance gaps in the benchmark between `Sparsh` (`DINO`), `Sparsh` (`IJEPA`), and `Sparsh` (`VJEPA`). However, this method is less effective for the `Sparsh` (`MAE`) model, which is trained in pixel space. We hypothesize that MAE weights are potentially more brittle when compared to other SSL models which enjoy a wider basin of minima due to weight updates via exponential moving average. In contrast, partial fine-tuning offers minor improvements, aligning closely with the performance of frozen models.

`Sparsh` ViT-small and performance

We train `Sparsh` for all SSL approaches decreasing the model capacity by using a transformer ViT-small. This let us study the effect of the dimensionality of the touch representations on downstream tasks, from 768 with ViTbase to 384 with ViTsmall. We follow the same training procedure explained in Appendix 2.10.2.

We evaluate `Sparsh`-vitsmall across `TacBench`. In Figure 2.19 we report the performance of each task for different budgets of labeled data following the attentive probing protocol. Reducing the dimensionality of the representations do plays an important role for some tasks. Regression-like tasks such as `[T1]` Force estimation (see Figure 2.19a) exhibit a decrease in performance when reducing the capacity of the encoders, specially when the downstream tasks needs to be trained under a limited number of labeled data. For instance, `Sparsh` (`DINO`) increases the force estimation error by 74% for DIGIT and 50.3% for GelSight Mini when using representations from `Sparsh`-vitsmall and training the downstream tasks with 33% of labeled data. The decrease in performance is also observe in regression-by-classification tasks, as in [T3] Pose estimation. With Sparsh-vitsmall all models perform very similar but losing 20% accuracy even when training the downstream tasks with the full labeled dataset. Nevertheless the performance is still better than an E2E model with a vitbase encoder.

In general for classification tasks in the benchmark like [T2] Slip detection, [T4] Grasp stability and [T5] Textile recognition, there is no major effect of reducing the capacity of the encoder. The drop in performance is only significant when training the downstream task with the lowest amount of training data, 1% in our experiments.

Sparsh and cross-sensory representation

Since Sparsh is trained on multiple GelSight-like data, we investigate whether SSL training enables cross-sensory representations or if it helps downstream tasks trained for one sensor quickly adapt to another. To study this, we use as a baseline the decoder trained for [T5] Textile recognition, which was supervised with labeled data from GelSight with markers.

We collect new data using a DIGIT sensor for 10 out of the 20 textiles. Our dataset contains 11 samples for both training and testing. We load the trained decoders for [T5] using Sparsh (DINO) and E2E and perform zero-shot evaluation as well as 1-shot, 5-shot, and 10-shot training and subsequent evaluation using the DIGIT data. Table 2.14 reports accuracy on 110 samples of test data. Zero-shot evaluation with DIGIT performs close to chance, while with very few samples (10-shot) Sparsh (DINO) classifier quickly adapts and significantly outperforms E2E. This experiment empirically demonstrates the value of cross-sensor representations.

	zero-shot	1-shot	5-shot	10-shot
Sparsh (DINO)	9.1	19.1	28.2	61.8
E2E	3.6	0.0	15.5	10.9

Table 2.14. Accuracy of n-shot evaluation of [T5] Textile recognition on DIGIT data to study how Sparsh facilitates cross-sensory adaptation to the dowsntream task.

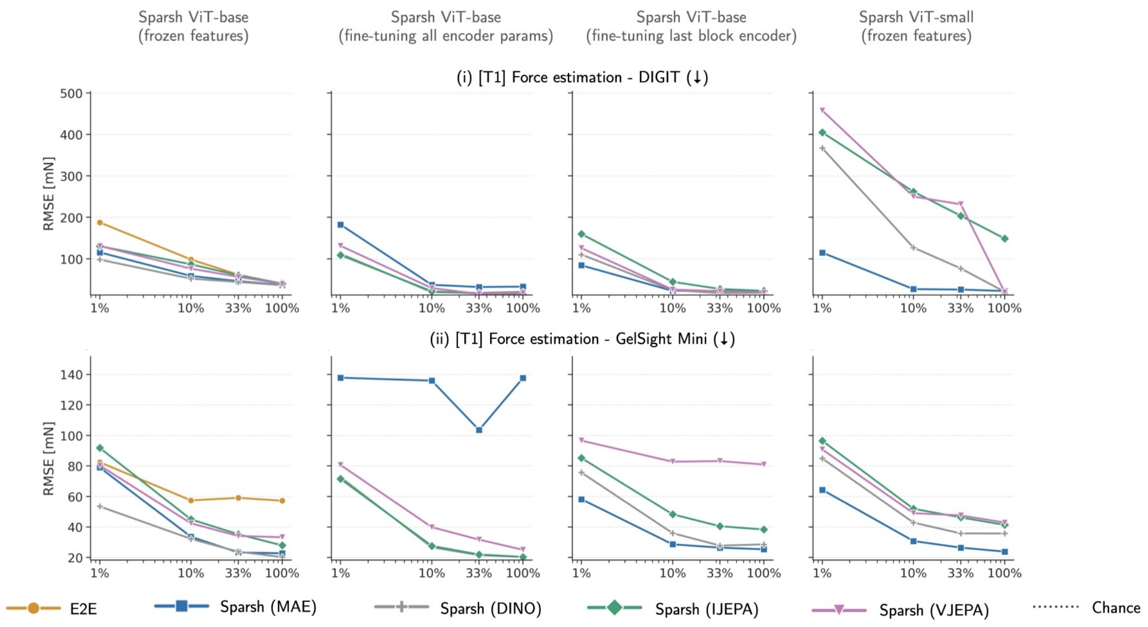


Figure 2.17. Additional evaluations for T1 force estimation.

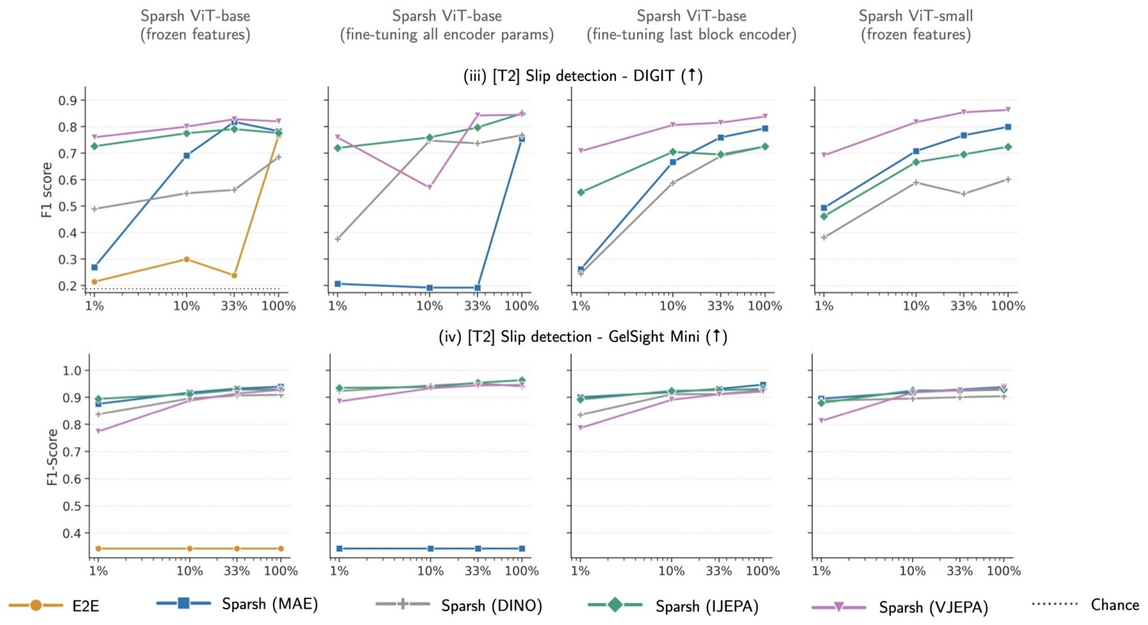


Figure 2.18. Additional evaluations for T2 slip detection.

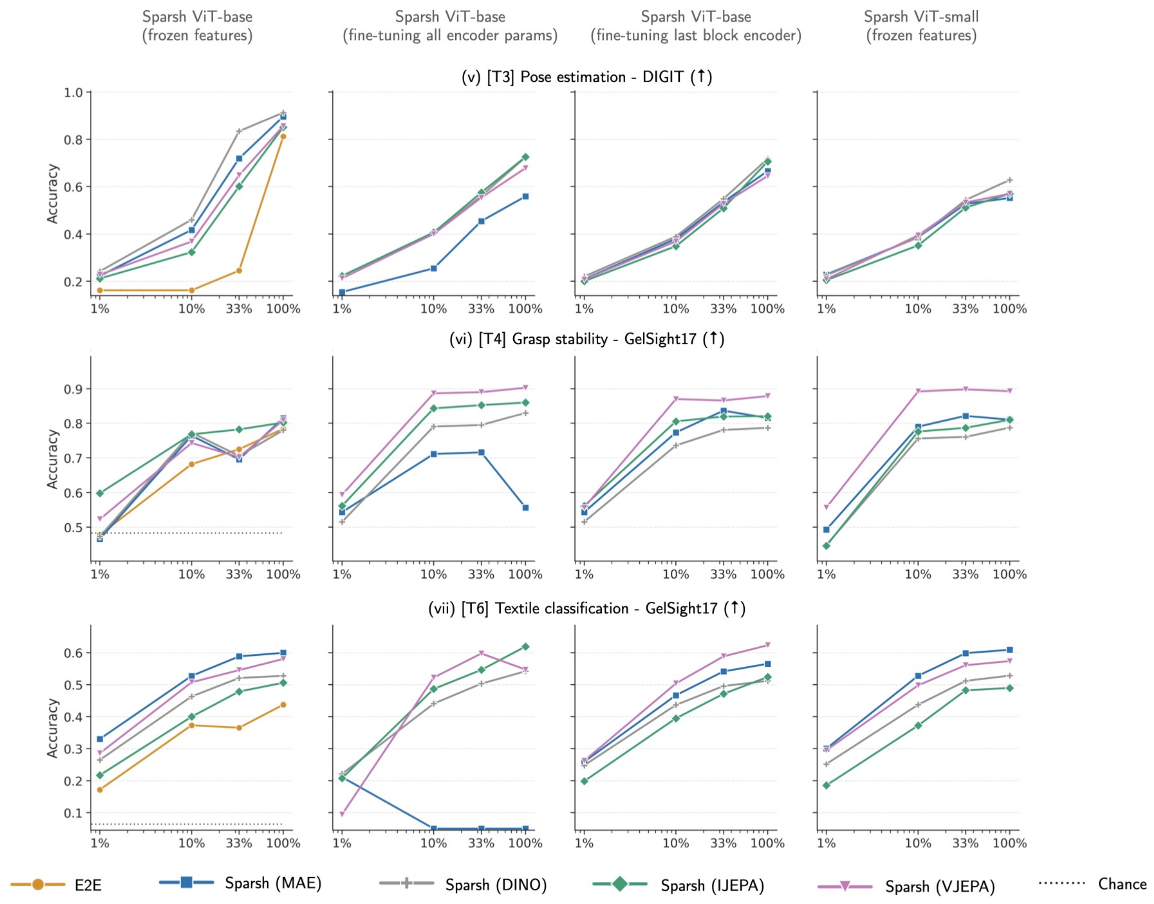


Figure 2.18. Additional evaluations for the perception tasks, T3 pose estimation (top), T4 grasp stability (middle) and T6 textile classification (bottom).

Figure 2.19. Additional evaluations of `Sparsh` representations on `TacBench`. We compare frozen `Sparsh` ViT-base (most left), `Sparsh` fully and partially fine-tuned (middle) and finally (most right) `Sparsh` ViT-small to gauge the effect of reducing the dimensionality of the representations.

Chapter 3

Sparsh-skin: self-supervised representations for full-hand magnetic tactile skin

Full hand tactile sensing is crucial for robot dexterity. Magnetic tactile skins offer a flexible form factor for hand-wide coverage with fast response times, in contrast to vision-based tactile sensors that are restricted to the fingertips and limited by bandwidth. However, challenges with interpreting magnetic flux changes, calibration, and the lack of general-purpose models have limited their adoption. In this work, we present `Sparsh-skin`, a pre-trained encoder for magnetic skin sensors distributed across the fingertips, phalanges, and palm of a dexterous robot hand. Given a temporal history of kinematics and tactile sensing across a hand, the encoder outputs a latent tactile embedding that can be used in any downstream task. `Sparsh-skin` is self-supervised via self-distillation on a variety of unlabeled hand-object interactions using an Allegro hand sensorized with Xela uSkin. We evaluate `Sparsh-skin` across several benchmark tasks from state estimation to policy learning, and find that pre-training results in significant improvements ($\sim 56.37\%$) in task performance and labeled data efficiency when compared to end-to-end learning with task-specific data.

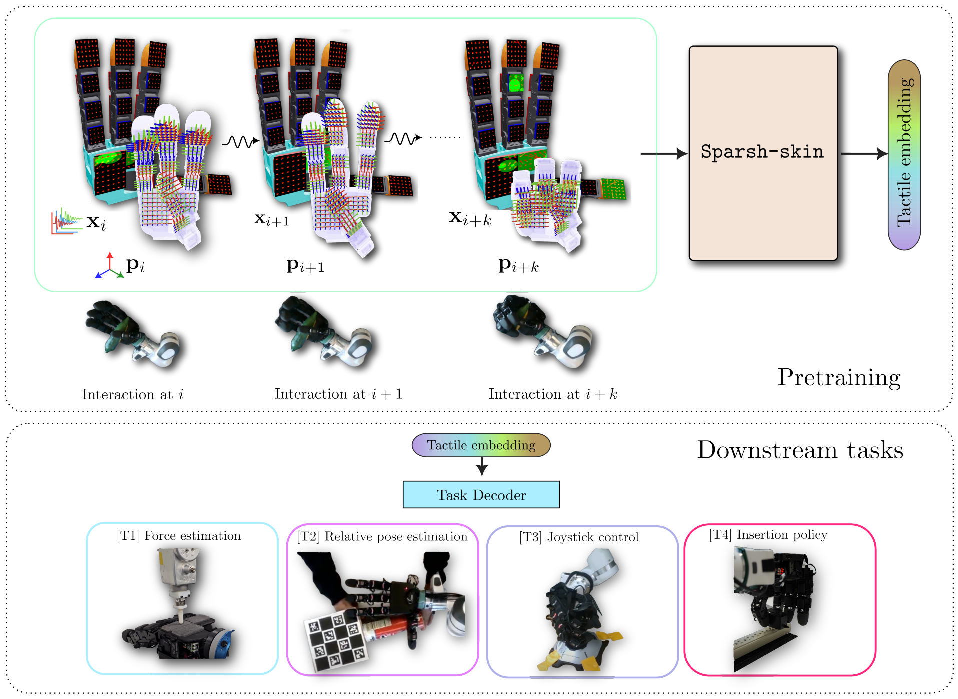


Figure 3.1. We present `Sparsh-skin` a self-supervised approach to learning general representations for magnetic tactile skins covering dexterous robot hands. `Sparsh-skin` is trained with a pretraining dataset (~ 4 hours) containing atomic in-hand interactions with a variety of objects. It takes as input a brief history of tactile observations $\mathbf{x}_i$ and 3D sensor positions $\mathbf{p}_i$ to produce performant full-hand contextual representations.

3.1 Introduction

Although often overlooked, touch comes at the forefront in dexterous manipulation, such as when playing guitar, or when vision is impaired, like plugging in a cord in the dark. Consequently, the robotics community has leveraged touch to enhance robot learning [82, 141, 145, 198], but has so far largely limited their attention to fingertip sensing via vision-based tactile sensors. Sensors such as the DIGIT [100], GelSight [207], GelSlim [45] and others [145, 181], are popular due to their high-resolution output, human-interpretable signals, and accessibility. Capturing touch as images is attractive, as advances in computer vision can be leveraged with minimal friction. Nevertheless, these sensors have limitations: they are slow compared to human skin's touch receptors, come in bulky form factors precluding large area sensing, and are often custom-designed for specific manipulators [145], making reproducibility a challenge. Magnetic skin-based sensors such as uSkin (Xela) [173, 175], ReSkin [14], and others [15, 136], offer an alternative for tactile feedback. They provide fast response times (~ 100 Hz) and flexible form factors that can be adapted to complex embodiments, such as multifinger robot hands. However, these sensors are difficult to interpret, difficult to model due to hysteresis and other factors, and are primarily hindered by a lack of infrastructure.

Self-supervised learning of general touch representations, offers a potential solution, through efficient downstream task learning, by learning priors from unlabeled data. However, while previous research [70, 186], has leveraged this idea for tactile learning, these approaches rely on choices from computer vision such as treating temporal signals as images [70] and using the masked reconstruction objective [78, 186] that may not be apt for noisy magnetic flux signals.

To this end, we present `Sparsh-skin`, a pre-trained tactile encoder model trained using self-supervised learning (SSL) for magnetic skin-like sensors covering a multifinger robot hand (see Figure 5.1). `Sparsh-skin` directly learns in-hand contact priors from tactile history and hand configuration using a robust classification objective. Our tactile encoder simplifies downstream task use, by introducing standardized magnetic time-series data, and reducing the need for real-world labeled data, which is non-trivial to collect and oftentimes infeasible. For instance, we do not yet have hardware to annotate spatially distributed ground truth force fields. By combining our representation learning algorithm, tactile signal tokenization, and a fully-sensorized multi-fingered hand, we achieve state-of-the-art tactile representations for magnetic-skin sensors, outperforming end-to-end training by $\sim 56.37\%$ in both performance and sample efficiency for downstream tasks.

We evaluate our model's understanding of dynamic contact through force prediction, pose estimation, object state estimation, and its capability for downstream policy learning via an insertion task. The main contributions of our work are:

1. `Sparsh-skin` a high-performance tactile representation model, trained via self-distillation [131] for magnetic-skin based tactile sensors.
2. A thorough evaluation of the design choices including tokenization, and the learning algorithm.
3. SSL dataset containing 4 hours of random play-data of the Allegro robot hand sensorized with the Xela tactile sensors, labeled datasets, metrics, and task design that cover relevant problems in tactile perception to evaluate learned representations.

3.2 Related work

3.2.1 Tactile sensors

Tactile sensors can be broadly categorized into vision-based (e.g. DIGIT [100], GelSight [207], GelSlim [45] and others [145, 181]), pressure-based (e.g. force sensitive resistors), impedance-based (e.g. BioTac [53]), and magnetic-based (e.g. uSkin (Xela) [173, 175], ReSkin [14], and others [15, 136]) sensors. Vision-based sensors commonly used in robot manipulation capture finger-object-environment interactions as images [100, 207]. However, their bulky form factor, low-frequency feedback and high bandwidth requirement limit their application in tasks that require large areas coverage. Impedance-based tactile sensors offer high temporal resolution, but are difficult to interpret, and currently do not provide full-hand coverage solutions either. Pressure-based sensors can offer a wide coverage area, but lacks capabilities in shear force sensing. Magnetic tactile sensors, on the other hand, provide a thin skin-like alternative with options such as ReSkin [14], AnySkin [15], and Xela [173, 175] being popular choices. They provide low-dimensional but high-frequency signals. However, when these sensor pads are distributed on all contact interfaces of a robot hand, the total output is high-dimensional. These sensors primarily use hall-effect sensing for force measurement. Xela [173] in particular works by transducing displacements of permanent magnets embedded in an elastomer arranged in a grid pattern to magnetic flux changes, essentially capturing 3-axis shear and normal forces. ReSkin and AnySkin [14, 15] magnetize the entire elastomer layer continuously, instead of using discrete magnets. This sensing modality has been explored for various contact-rich applications, including planar pushing [109], surface material classification [184], grasp stability [56], and policy learning tasks [107, 135].

3.2.2 Tactile representation learning

Representation learning for vision-based tactile sensors has recently gained significant attention. Since the sensor outputs are images, techniques from computer vision [78, 177] have been extended to tactile sensors. This is motivated by a move beyond task-specific encoders to pretrained encoders that promise generalization, with prior work leveraging mask-autoencoders (MAE) [21, 215], contrastive learning [39, 63, 197, 204], and state-of-the-art methods like self-distillation and joint-embedding predictive architectures [82] to learn tactile representations.

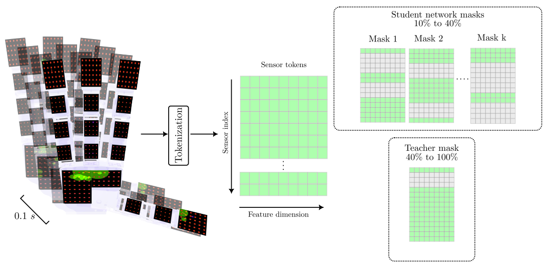


Figure 3.2. Illustration of Xela signal corruption via masking for SSL prediction task: Once a 100(ms) window of tactile measurements and sensor positions are tokenized, block masking is applied to corrupt the signal, . For each data sample, the student network receives k different masks, each randomly retaining 10% to 40% of the data denoted $\bar{z}_i$. The teacher network, in contrast receives 1-2 masks each retaining 40% to 100% of the data denoted z_i^*.

Research on learning representations for magnetic-based sensors remains relatively underexplored. Since these sensors produce low-dimensional signals, the consensus view is that representation learning is likely unnecessary. However, as we highlight in our work, these signals in context of the full hand sensing, varying tactile signals and hand poses over time, and physical properties of magnetic sensors are indeed high-dimensional, and benefit from large-scale pretraining to compress information into semantically rich representations, that improve downstream task performance.

Recently, HyperTaxel [107] applied contrastive learning to learn representations for the Xela sensor for the task of surface recognition but it did not show whether these representations capture contact dynamics. Similarly [70, 186] propose representation learning with self-supervised methods such as BYOL [67] and MAE [78]. While the idea of representation learning is sound, the choice of meaningful image augmentations without data corruption, is unclear for BYOL. Furthermore, by treating instantaneous tactile measurements as images, these methods discard temporal information and may therefore be suboptimal for tactile tasks that rely on contact dynamics.

3.3 Representations via self-supervised learning for tactile skins

Dexterous manipulation has primarily focused on fingertip tactile sensing, which provides crucial information for tasks like in-hand rotation [141, 198]. However, other skills such as in-hand translation [199], power grasp, and palm-to-finger retrieval, and maneuvers involved in tool use require the involvement of phalanges and the palm. Magnetic skin sensors, such as Xela [173], offer a form-factor that allows easy integration into dexterous hands to provide a full hand-object state with lower dimensionality and higher frequency than vision-based sensors. Therefore, we propose a self-supervised modeling approach to learn from random-play data, generalizable tactile features for dexterous hands equipped with magnetic-skin tactile sensors. Our method is designed for the Xela sensor but can be extended to any skin sensor with 3-axis time-series output signals.

3.3.1 Preliminaries

Self-distillation for representation learning Self-distillation [6, 67, 131] is a powerful paradigm in self-supervised learning involving a pair of identical neural networks, termed the student $\mathbf{E}_\theta$ and teacher network $\mathbf{E}_{\hat{\theta}}$. The student network receives a corrupted version of a data signal $\bar{\mathbf{x}}$ that is to be encoded, while the teacher network receives privileged information about the same data sample $\mathbf{x}$. Then, the student network is tasked with predicting through a small predictor network $\mathbf{P}_\phi$, the data representation that the teacher produced. To prevent the teacher from producing degenerate representations – for instance, a constant representation for all data – the teacher weights are not updated via back-propagation, but only through an exponential moving average (EMA) of the student weights. Specifically, the following objective is optimized:

$$\underset{\theta,\phi}{\operatorname{argmin}} \|\mathbf{P}_\phi(\mathbf{E}_\theta(\bar{x})) - \operatorname{sg}(\mathbf{E}_{\hat{\theta}}(\mathbf{x}))\| \tag{3.1}$$

where sg indicates stop gradient, and $\hat{\theta} \triangleq \text{EMA}(\theta)$.

Since the teacher network is an exponential moving average (EMA) of the student work, knowledge is *self-distilled* through the representation prediction task.

Robot setup and pretraining data Our setup consists of the Allegro hand sensorized with Xela uSkin, attached to a Franka Panda robot arm. The Allegro hand is equipped with 18 Xela uSkin sensing pads, consisting of 4 curved fingertip sensors with 30 individual sensors, 11 4x4 grid sensors pads attached to the finger phalanges, and 3 4x6 sensing pads attached to the palm, resulting in a total of 368 individual sensors.

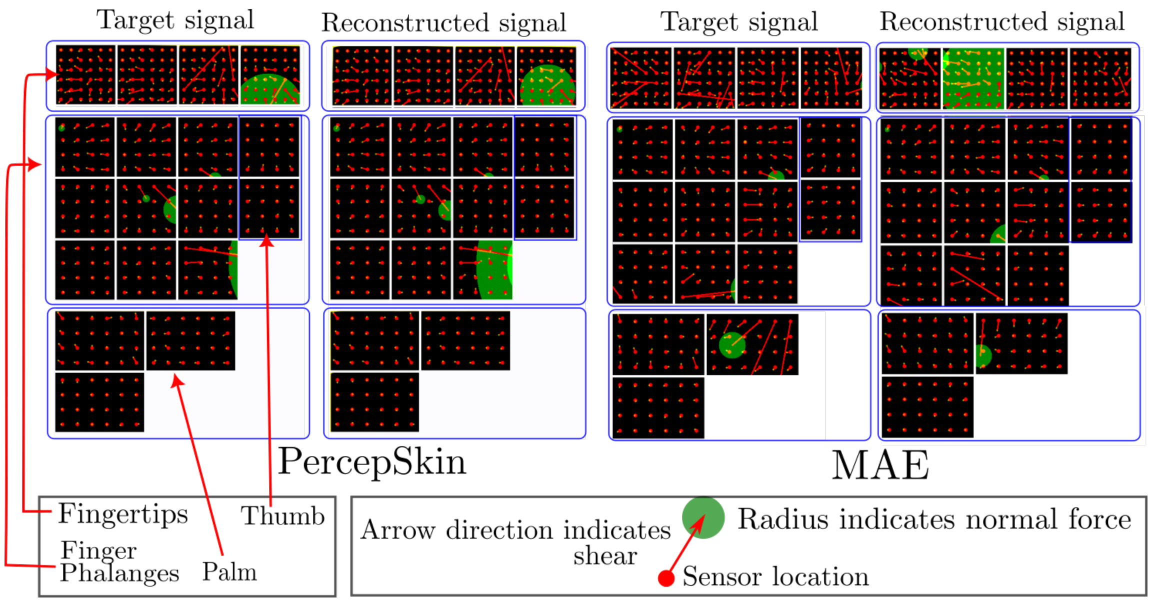


Figure 3.3. Visualization of reconstructions from the reconstruction online probe. Here, we show a comparison between MAE and `Sparsh-skin`. For `Sparsh-skin` we visualize a single frame from the tactile window.

We collected a dataset of the hand performing various atomic manipulation actions with 14 household objects and toys, including squeeze, slide, rotation, pick-and-drop, circumduction, pressing, wiping, and articulation. Using a VR-based teleoperation system with Meta Quest 3, which builds on the inverse kinematics-based re-targeting method proposed in [73], we recorded 11 sequences (approximately 2 minutes each) for each object, totaling around 4 hours of varied interactions. The dataset includes top/left camera views, Xela signals, and robot and hand joint states, covering a range of rigid and deformable objects with diverse tactile properties (see Figure 3.4).

3.3.2 Architecture

`Sparsh-skin` uses a Transformer[178] as the student and teacher network for self-distillation.

Sensor tokenization We perform baseline subtraction on Xela signals to account for their uncalibrated nature and consistent biases. A single baseline signal is collected with the Allegro hand in a resting configuration (palm up and flat) and used for all downstream tasks, unlike prior work [14, 135], which collects a new baseline signal per training sequence. We also resample Xela signals to a consistent 100Hz frequency. as the sensor data rate fluctuates between 80Hz to 100Hz, unlike prior work [70] that subsamples data to match modalities at lower frequencies.

We note that for representation learning, tactile data can be temporally correlated, and instantaneous signals cannot provide context for contact changes, therefore we choose to learn representations for chunks of 100ms of data. First, inputs to `Sparsh-skin` are formatted corresponding to a brief history of 0.1 seconds of the sensor signal $x_{1:10} \in \mathbb{R}^{10\times368\times3}$ concatenated with the history of sensor position $p_{1:10} \in \mathbb{R}^{10\times368\times3}$ computed from the forward kinematics of the Allegro hand. Inputs are then tokenized through a linear projection f_{linear} to the dimension d of the representation $z_i = f_{\text{linear}}(x_{1:10}|p_{1:10}) \in \mathbb{R}^{368\times d}$. Finally, a learnable token is added to each sensor according to the three types of Xela sensing pads (see 3.3.1) on the Allegro hand. We do not add additional positional embedding and instead rely on the sensor position to provide 3D positional information to the transformer network.

SSL prediction task

While it is common in the image domain to crop and subsequently resize images to perform signal corruption, this results in changing the shear profile for magnetic flux readings. Therefore to avoid any untoward data augmentation that changes the semantic meaning of the signal, we use block masking [5] to corrupt signals that are input to the encoding networks. Specifically, input data is masked after sensor tokenization in a cross-taxel manner i.e., given tokenized data from 368 sensors, we mask sensor data from local contiguous blocks including from sensors from distinct sensor islands by removing those sensors from the input (see Figure 3.2). The masked sensor tokens are subsequently transformed through the student and teacher network as $\mathbf{E}_\theta(\bar{z}_i)$ and $\mathbf{E}_{\hat{\theta}}(z_i^*)$ respectively.

For the prediction task, we use classification by defining a set of prototype classes as in [22, 131], which is robust to sensor noise compared to masked auto-reconstruction. The sensor tokens after transformation are converted into prototype logits through a classification head f_{class} as $\bar{p}_i$ and p_i^* respectively for the student and teacher networks. We use both the class token and the patch level cross entropy objective between the student and teacher logit predictions to enforce local-to-global correspondence learning in the sensor representation. Additional details about the model architecture, MAE reconstruction comparisons and training hyper-parameters are in the Appendix.

Online Probes

Unlike supervised learning (SL), where model performance is easily monitored through training and validation losses, in self-supervised learning (SSL), prediction task losses do not directly convey downstream task performance. In fact, in the presence of an EMA teacher network, which acts as a moving target, the prediction task loss can increase in tandem with the predictions of the teacher network. Therefore, we rely on online probes to monitor downstream performance. During training, we evaluate the tactile representation for a) reconstruction and b) the ability to identify objects used in play data.

Figure 3.3 provides a qualitative visualization of the reconstruction performance obtained by the decoder using representations computed by the student network $\mathbf{E}_\theta(\bar{z})$. In terms of object classification performance, we achieve approximately 95% accuracy

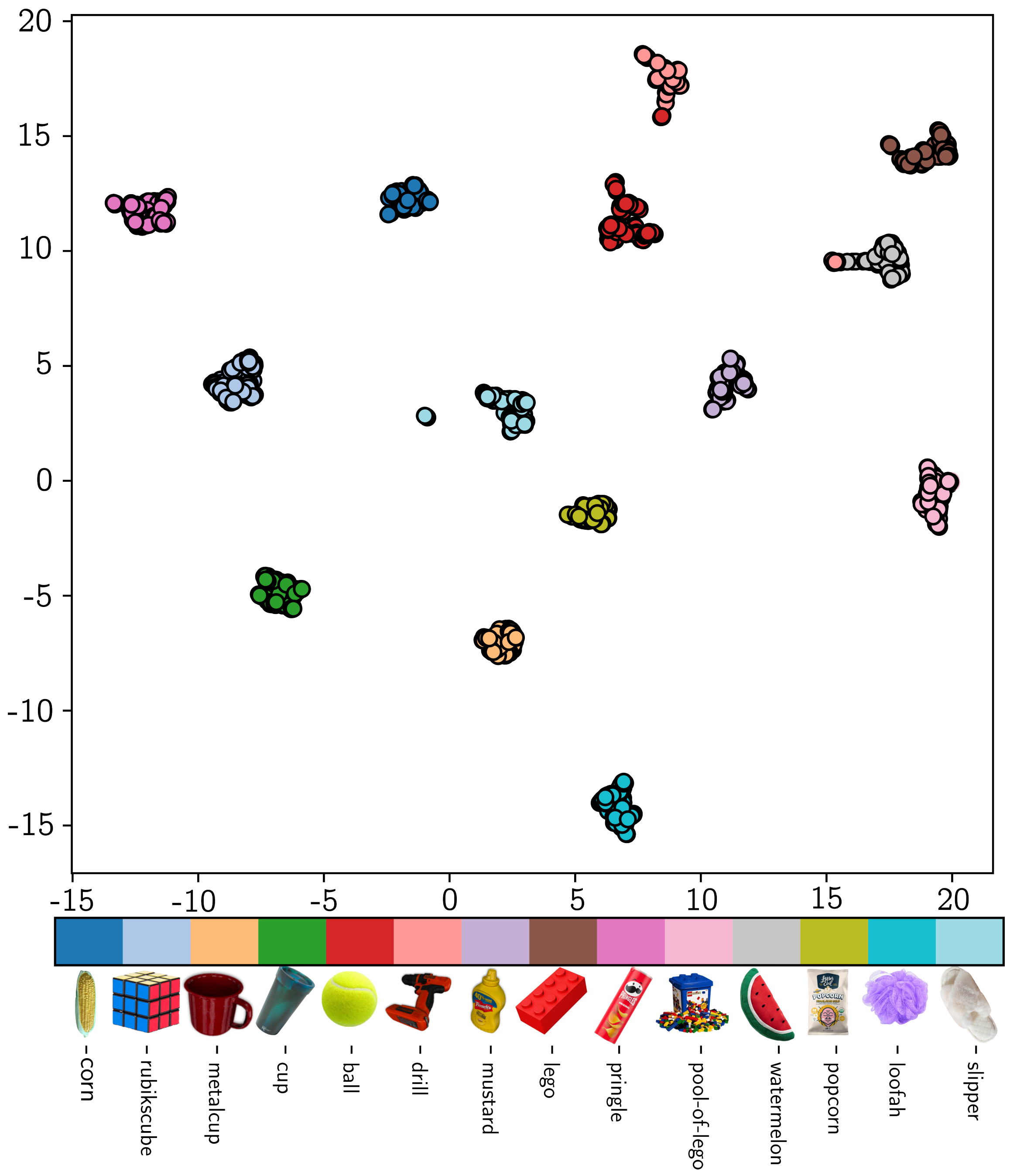


Figure 3.4. UMAP visualization of representations colored by object in robot hand.

across 14 classes, while both BYOL [70] and MAE [78] are limited to ∼ 81% accuracy. Additionally, Figure 3.4 presents a UMAP [122] visualization of the representations, where sequences from each object are mapped to distinct, non-overlapping clusters.

3.4 Sparsh-skin evaluation

In this section, we assess the ability of `Sparsh-skin` to comprehend tactile properties, enhance perception, and enable policy learning for manipulation through four downstream tasks spanning tasks studied in the tactile sensing literature: namely Force estimation, Joystick state estimation, Pose estimation, and policy learning via the Plug insertion task.

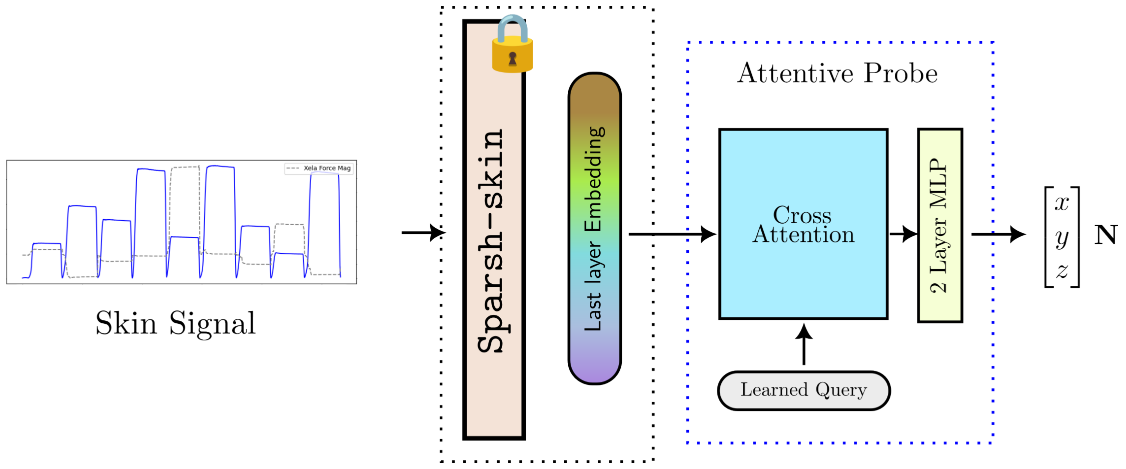


Figure 3.5. Attentive probe: Attentive pooling + small 2-layer MLP for regression tasks

3.4.1 Evaluation protocol

Downstream task decoders. The tasks we consider are of two types: a) requiring instantaneous prediction, and b) requiring temporal reasoning over tactile data. For tasks such as force estimation that require an instantaneous estimate, we use attentive pooling(see Figure 3.5). For tasks such as pose estimation and joystick state estimation, that require sequence reasoning, tactile observations are transformed into tokens at the output frequency through a cascaded application of the backbone network followed by attentive pooling as illustrated in Figure 3.6.

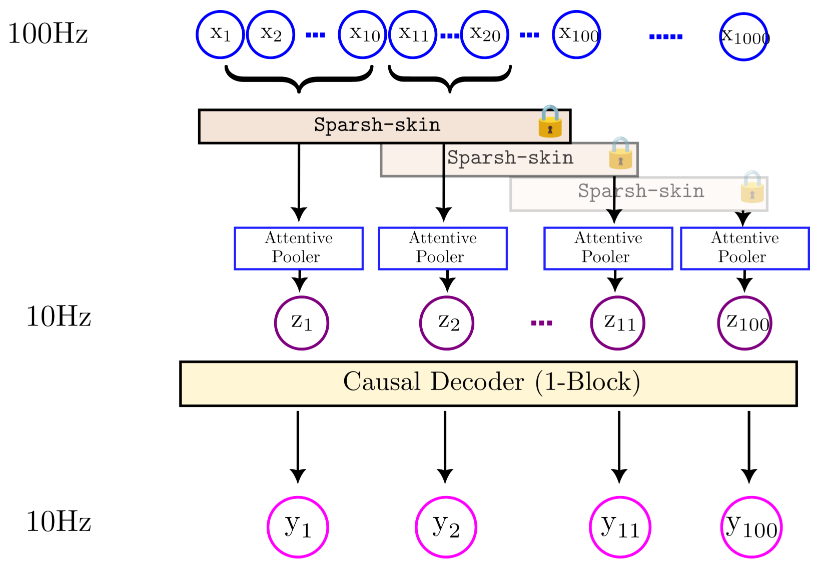


Figure 3.6. Decoder for sequence to sequence prediction tasks

Model comparisons. For each of the downstream tasks, we explore multiple variants of the `Sparsh-skin` encoder, along with additional baselines:

1. `Sparsh-skin` (frozen), pretrained representation that uses tactile and hand configuration history.
2. `Sparsh-skin` (MAE), pretrained representation that uses tactile and hand configuration history trained using MAE supervision instead of self-distillation.
3. `Sparsh-skin` (finetuned), finetuning the pretrained encoder network
4. BYOL*, our reproduction of the BYOL [67] approach using our collected play data and tactile data formatted as images, since the setup used in [70] does not contain palm sensing and uses an older variant of the tactile sensor.
5. End-to-end, training the entire encoder-decoder network with same capacity using only labeled task data

To measure performance, we generally use the average root mean squared error (RMSE) for all tasks, and measure success rate (SR) across trials for the policy learning task using plug insertion. Additionally for all supervised learning tasks, we evaluate each method for sample efficiency by reducing the downstream labeled data accessible during training.

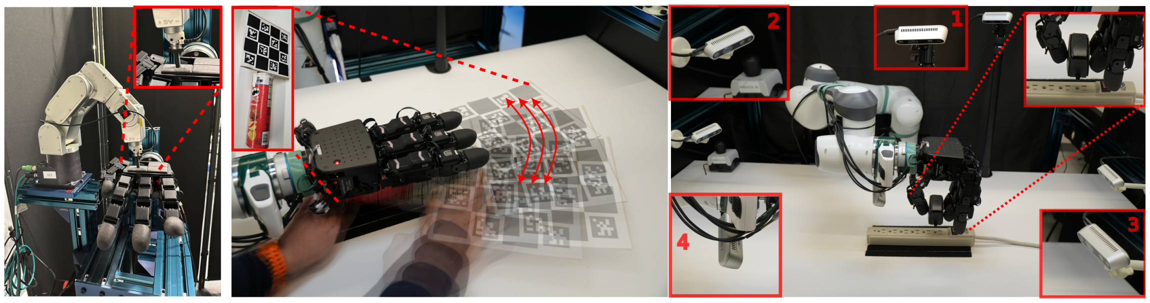


Figure 3.7. Hardware setup used for downstream tasks: **(Left)** shows the setup for force estimation. We use 3D printed probes attached to a F/T sensor to indent onto the Xela sensors. **(Middle)** shows the setup for pose estimation. We track an object mounted with an ArUco marker to obtain ground truth pose estimates while randomly moving it under the robot hand. **(Right)** shows the setup for plug insertion policy task. We collect tactile measurements and camera observations from three third-person view cameras and a wrist camera view.

3.4.2 Downstream task performance

Force Estimation. This task involves regressing tactile signals to 3-axis normal and shear forces on a robot hand's palm. We collected force-labeled data using a robot arm with an F/T probe to apply varying normal forces (0.25-5.0N) with hemispherical and flat indenters (see Figure 3.7 (left)). The probe's position was randomly sampled across the sensor pad, including locations both on and between magnetometers, differing from sensor characterization which only tests atop magnetometers.

Results (see Figure 3.8) While the end-to-end model is particularly worse at predicting forces throughout the spectrum, in low data regimes – 3.3% to 10% of the labeled data in this case – it is interesting to note that `Sparsh-skin` (finetuned) and `Sparsh-skin` (frozen) do not see any significant loss in performance. To this end, we test the models with even smaller number of downstream task data samples to find that `Sparsh-skin` is able to predict forces at a reasonable accuracy (350 mN in z) even with only $\sim$ 100 samples. Additionally, we find `Sparsh-skin` (MAE) is marginally worse at predicting forces highlighting that MAE may not be suitable for noisy magnetic flux signals. Furthermore, BYOL* is competitive albeit slightly worse with respect to `Sparsh-skin` (frozen) as this task tests for instantaneous force decoding. We present additional results in the appendix.

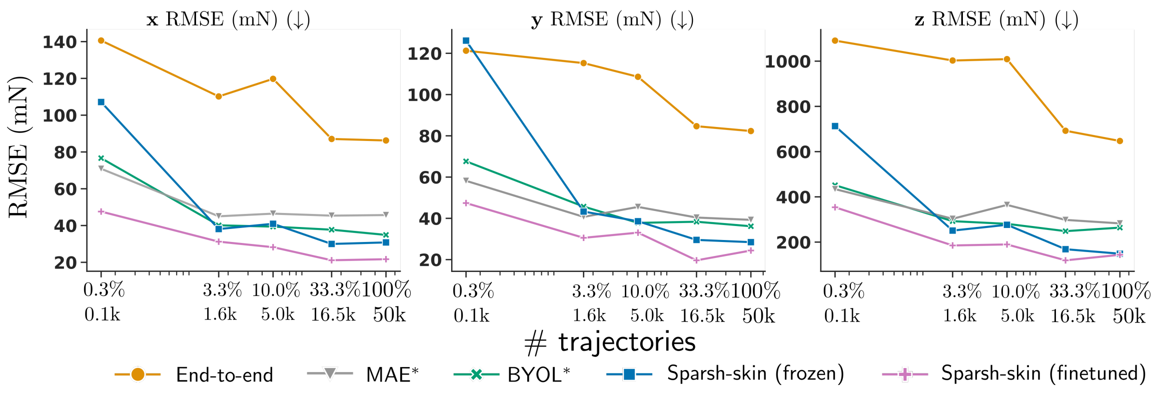


Figure 3.8. Force estimation (RMSE (↓)): BYOL pre-training is less accurate at predicting normal forces.

Joystick state estimation. We adapt this task from [16] (see Figure 5.1), as a study of *full-hand* object state estimation. The task is a sequential problem of predicting the joystick states (roll, pitch and yaw) given a short tactile history. In addition to the comparison of `Sparsh-skin` with an end-to-end approach, we also compute the RMSE results from the best reported model in HiSS [16] (denoted as HiSS* in Figure 3.9). Additional data and pre-processing details are in the appendix.

Results Despite challenges from jittery teleoperation such as inconsistent touch even with similar joystick maneuvers, our model (`Sparsh-skin`) matches baseline performance using full data. Notably, `Sparsh-skin` (frozen) achieves similar performance with only 3.3% of the data, demonstrating high sample efficiency. `Sparsh-skin` consistently shows lower prediction error across data budgets (Figure 3.9). Furthermore, `Sparsh-skin` (finetuned) drastically speeds up training, reaching comparable performance to an end-to-end approach in 12k optimization steps versus 220k (a 95% speedup) when using a 33% data budget. An illustration is provided in the Appendix.

Pose estimation. This task tests the ability to track and accumulate slip under the sensors to predict object pose changes ($\mathbf{t}_i^R \triangleq (x, y, \theta)) \in \mathbf{SE(2)}$ using the setup in Figure 3.7 (middle). We collect 120 trajectories ($\sim$ 30s each), by manually sliding/rotating an object in a range of $\sim (25\text{cm}, 25\text{cm}, 100°)$ under the Allegro hand, tracking ground truth object pose using ArUco tags. These poses in the camera frame are transformed into the robot hand frame and then projected into $\mathbf{SE}(2)$. We use the sequence decoder (Figure 3.6) which processes 1-second windows of raw tactile data (100Hz) and object pose (10Hz). In addition to RMSE, for this task,

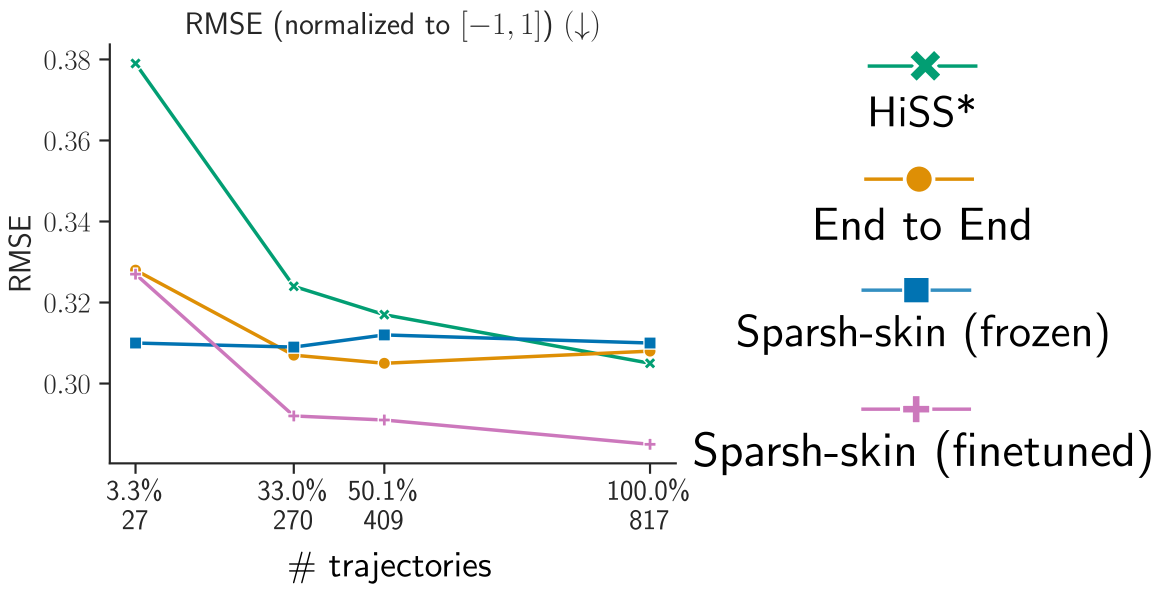


Figure 3.9. Joystick state estimation (↓): `Sparsh` outperforms end-to-end overall and is competitive with HiSS* even when it is given access to only 3.3% of dataset.

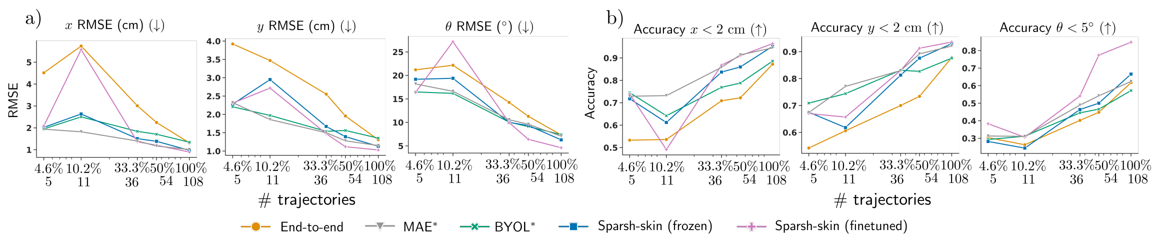


Figure 3.10. Pose estimation error (↓) and **(d)** Pose estimation accuracy (↑): `Sparsh` (finetuned) has a $\sim 10\%$ improvement over end-to-end for translation and $\sim 20\%$ improvement for rotation.

Model	Success Rate (↑)
VisuoSkin	0.66
V	0.20
V+T End-to-end	0.4
V+T Probing	0.75
V+T Fine-tuning	0.7

Table 3.1. Success rate percentage reported over 20 trials, while ensuring identical initial conditions during each trial for the tested policy variants. VisuoSkin results are obtained from [135]

performance is measured via pose accuracy (proportion of predictions within 2cm translation and 5° rotation error).

Results (see Figure 3.10) To evaluate sample efficiency, we train with data from sequences ranging between 5 to 100 trajectories. All pre-trained representations trained on play data achieve lower RMSE and higher pose prediction accuracy than the traditional end-to-end approach. In particular for `Sparsh-skin` (finetuned) we find a $\sim$ 10% improvement over the end-to-end model with the full dataset for translation, and $\sim$ 20% improvement for rotation. `Sparsh-skin` (frozen), the MAE baseline are on par with each other, while BYOL is worse at high-data regimes. Although all models suffer in accuracy in low data regimes, pretrained `Sparsh-skin` models maintain 70% accuracy rate for translation. This is explained by the correlation of the displacement of the object and the displacement of the magnetometers on the Xela sensors. However, tracking rotation is harder, as it involves torsion. Allowing in-domain data to fine-tune the `Sparsh-skin` representations is advantageous, especially for better tracking rotation of the object.

Policy learning (plug insertion). We train a transformer decoder policy predicting action chunks [217] with `Sparsh-skin` representations as input for this task. We adapt the insertion task [43, 135, 187] as it is fundamentally tactile requiring touch feedback to observe the alignment state of the plug. The task involves inserting a pre-grasped plug into a fixed socket using a 7-DOF Franka arm and Allegro hand (Figure 3.7 right) unlike [135] which used parallel jaw grippers. We collected 100 demonstrations via kinesthetic teleoperation, recording synchronized data: four camera views ($\mathbf{I}_t^{\text{left}}, \dots$), Allegro tactile readings (z_t), and robot joint states. The arm's initial position was randomized within a $0.05m \times 0.05m \times 0.02m$ volume $\sim$ 10cm above the socket, while the socket position is fixed. The policy predicts sequences of absolute end-effector poses (3D position + axis-angle orientation) $\mathbf{a} \triangleq (\mathbf{T}_t, \mathbf{T}_{t+1}, \dots)$, conditioned on visual and tactile observations but not proprioception (joint states). We evaluate average success rate over 20 trials with randomized start positions, comparing `Sparsh-skin` variants (V + `Sparsh-skin` (frozen), V + end-to-end, V + `Sparsh-skin` (finetuned)) against a vision-only (V) baseline to assess tactile contribution. Further details are in the Appendix

Results: (see Table 3.1) We find that policies conditioned on pretrained `Sparsh-skin` features outperform the end-to-end model. In the appendix, we present snapshots of real-world policy deployments for both vision-only and visuo-tactile `Sparsh-skin` (frozen) policies. Without tactile modality (vision only), we find that the policy is able to get close to the socket but indefinitely continues to search for it and does not push the plug in, even when it is directly above the socket. Further, we find that this policy tends to keep pushing the plug to the left of the socket. We note that this is due to perceptual aliasing, where the plug incorrectly appears to be right above the socket from the wrist camera. On the other hand, all model variants with access to the tactile modality observe respectable success rates. In qualitative inspection, we find that the policies using `Sparsh-skin` (V+T frozen) representations slides after making contact with the extension board, while `Sparsh-skin` (V+T end-to-end) and `Sparsh-skin` (V+T finetuned) tends to retry by lifting the plug, when mistakes occur. As noted earlier, in comparison with [15, 135] which trains end-to-end visuo-tactile policies, our setup uses a multifinger Allegro hand as the manipulator, where the plug is grasped using three fingers; nevertheless, we find that policies trained with tactile features from `Sparsh-skin` are competitive.

3.5 Conclusion

We present `Sparsh-skin`, a high-performance tactile representation model trained via self supervision for magnetic skins on dexterous hands. Our model learns contact state priors from an unlabeled dataset of contact-rich teleoperated hand interactions with various household objects. We demonstrate the efficacy of our supervision objective, tokenization, and masking strategies through evaluation across various tactile centric tasks spanning force estimation, pose estimation, object state estimation and policy learning, which indicate that our model is highly performant.

3.6 Appendix

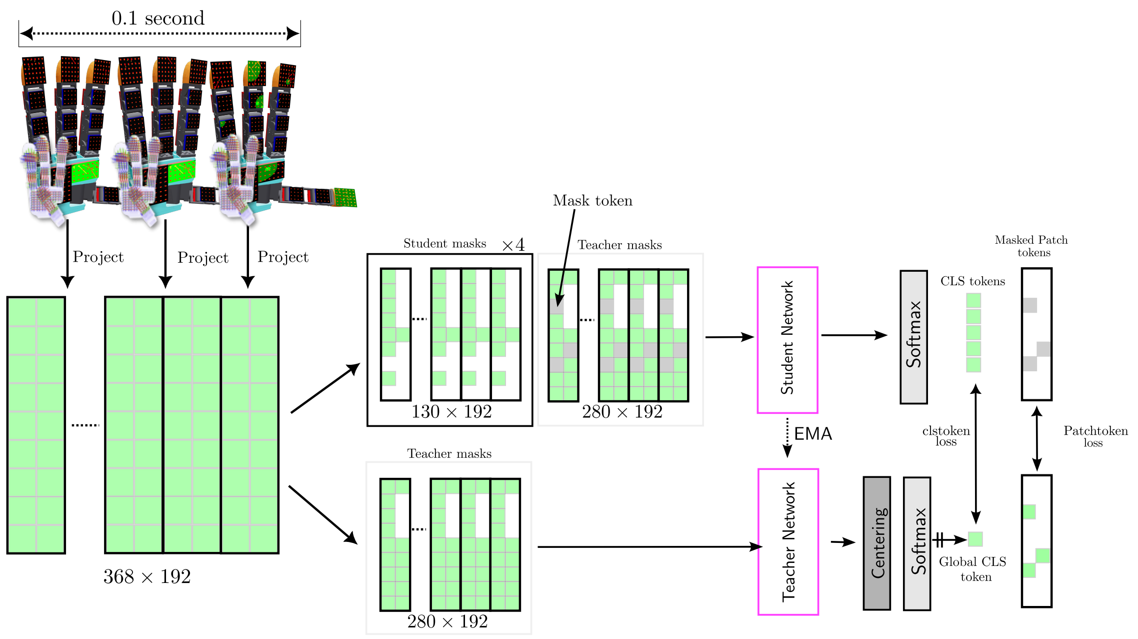


Figure 3.11. **Sparsh-skin block diagram for self-supervised learning of skin representations.** Our approach follows the student-teacher framework and loss functions used in self-distillation. However, we adapt the transformer input tokenization to accommodate time-series Xela data.

3.7 Sparsh-skin self-supervision details

3.7.1 Training details

We train Sparsh-skin on 8 Nvidia A-100 (80G) GPUs. To monitor learning, we use reconstruction online probe and classification via linear probing. We use AdamW optimizer and use a linear rampup followed by a cosine schedule as the learning scheduler. Further, we find that tuning momentum value as well as the weight decay factor was important in observing training convergence. Additional information of hyperparameters is detailed in Table 3.2.

Architecture	ViT-Tiny (adapted)
Embedding dim	192
EMA decay	[0.994, 1.0]
LR	1e-4
Batch size	64

Table 3.2. **Training hyperparameters for `Sparsh-skin`.** All models run for 500 epochs with optimizer AdamW, a weight decay cosine schedule from 0.04 to 0.4, and a learning rate warmup of 30 epochs.).

3.7.2 Architecture details

Our encoder model is a modified version of Vision Transformers [46]. Specifically, we adapt the tokenization of the time-series Xela with sensor pose data. After flattening the 3D-axis magnetic reading per magnetometer (368) and concatenating their corresponding pose in chunks of 0.1 second, the inputs $x \in \mathbb{R}^{10\times368\times6}$ are tokenized through a linear projection to the dimension d of the representation $f_{linear}(x) \in \mathbb{R}^{368\times d}$. We use a tiny model with $d = 192$. We add a learnable embedding to identify different types of xela pads (palm, phalanges and fingertips). Then, we construct different cropped view of the data, two global views and eight local views. We mask sensor data from contiguous blocks by removing those sensors from the input. For the local view we retain between 10% and 40% of the tactile signal, whereas for the global views we retain 40% to 100%. An illustration of the masking and diagram block of the pipeline for self-supervised learning of Xela representations is shown in Figure 3.11.

The student and teacher share the same encoder and projector head architecture, both initialized with the same weights. The projector head corresponds to a 3-layer MLP with an output dimension of $k = 65536$. We use the projection head for the proxy prediction task to distill knowledge to match output distributions over k dimensions between student and teacher networks. The student network is updated via back-propagation, while the teacher network is updated at a lower frequency via exponential moving average (EMA) on the student weights. We pass the global and local views to the student encoder, while the teacher only has access to the global views. The register tokens from global/local views are passed through the projection head. For the teacher only, the output is also centered and sharpened via softmax normalization.

3.8 Additional task details

We provide additional information about the decoder architectures for each task, as well as additional results to highlight the performance on downstream tasks when using frozen or fine-tuned `Sparsh-skin` representations. Also, please refer to Table 3.3 for details on labeled data curation for evaluation tasks.

Task	Dataset	Size	Collector	Label
Force estimation	Normal load (indenter: sphere, flat)	50k datapoints	Robot	3-axis force
Pose estimation	Object sliding	108 trajectories	Human	Object pose $\mathbf{SE}(2)$
Joystick state estimation	Joystick motion	817 trajectories	Human	Normalized roll, pitch, yaw
Plug insertion	Demonstrations	100 trajectories	Human	Absolute EE pose

Table 3.3. Datasets for evaluating `Sparsh` representations on downstream tasks.

3.8.1 Force estimation

`Sparsh-skin` features are pooled via attentive pooling to obtain a full-hand representation $z_{hand} \in \mathbb{R}^d$. The force decoder consist of shallow 2-layer MLP with 3 outputs regressing to normalized force for each axis.

In Figure 3.14 we illustrate the data protocol followed for force estimation, which we note is different from the protocol that is usually followed for force characterization of tactile sensors. We note that we indent the tactile sensor pads at both, positions on top of the sensor as well as positions in between magnetometer locations, while choosing these positions randomly. This results in cases where the probe may slide and present slightly uneven force outputs. Specifically, in figure 3.14(b) we note that `Sparsh-skin` predicts the correct normal forces, while accumulation (mean) of normal forces from the magnetometers over the sensor pad results in inconsistent force outputs compared to ground truth.

In Figure 3.15, we present the correlation metrics between ground truth and predicted forces on test data for decoders trained with a 33% data budget. The results show that end-to-end training leads to overfitting, resulting in poor generalization to unseen strokes and essentially random normal force predictions. In contrast, using `Sparsh-skin` (frozen) representations yields better fitting, which can be further improved by adapting these representations to in-domain data.

In Figure 3.16, we present a comparison between ground-truth testing strokes (normal loading sequences) and their reconstructed counterparts, obtained by passing Xela data through the frozen force decoder to recreate the sequences. The forces estimated via `Sparsh-skin` (frozen) are able to capture increasing/ decreasing changes in the normal loading, as opposed to the end-to-end model. Shear from skin representations is not as accurate as normal force prediction, but the trend of the tangential forces matches the ground truth.

3.8.2 Joystick state estimation

For this task, we highlight that when we train decoders using pretrained representations as the input, the convergence rate of the validation RMSE is significantly higher (see Figure 3.17) than training the decoder using raw observations through uninitialized models. Specifically observe that `Sparsh-skin` (fine-tuned) is able to reach performance on par with end-to-end pretrained model within 12.9k optimization steps.

3.8.3 Pose estimation

In this task, we aim to predict the object pose over 1-second trajectories. Xela observations at 100Hz are converted into tactile representation tokens at the output frequency using `Sparsh-skin` in a cascaded manner. Following attentive pooling, a single-layer transformer block is applied to reason about the 1-second context window of full-hand tactile features.

Figure 3.18 compares ground-truth test pose sequences with their reconstructed counterparts, obtained from task models trained on 100% and 33% of the available data. The results show that fine-tuning `Sparsh-skin` on the full dataset yields higher accuracy in estimating object pose changes over time compared to traditional end-to-end approaches. Moreover, even with a drastic reduction in labeled samples (to 33%), the model still achieves relatively good performance, particularly in tracking translation changes. Furthermore, for this task, we also visualize that this tasks requires *full-hand* sensing. For instance, in Figure 3.19, we observe that when we use `Sparsh-skin` by removing palm sensing on the Xela hand results in $\geq 10\%$ drop in pose tracking performance.

3.8.4 Policy learning (plug insertion)

For this task, we use a transformer decoder to predict action sequences given camera and tactile observations. Figure 3.20 illustrates the architecture of the transformer decoder used in this work. Images are encoded using a Resnet18 CNN, which are trained from scratch to produce image features, while the tactile observations are processed through `Sparsh-skin`. Further, a learnable token (CLS / action) token is also concatenated with the observation tokens. After processing through the transformer, we extract the action token, which is then passed into a small 2-layer MLP to predict a sequence of actions. For this task, follow an *receding-horizon* control approach, where we choose a prediction action sequence length of 16, of which 8 actions are executed, given only the observations from the current timestep.

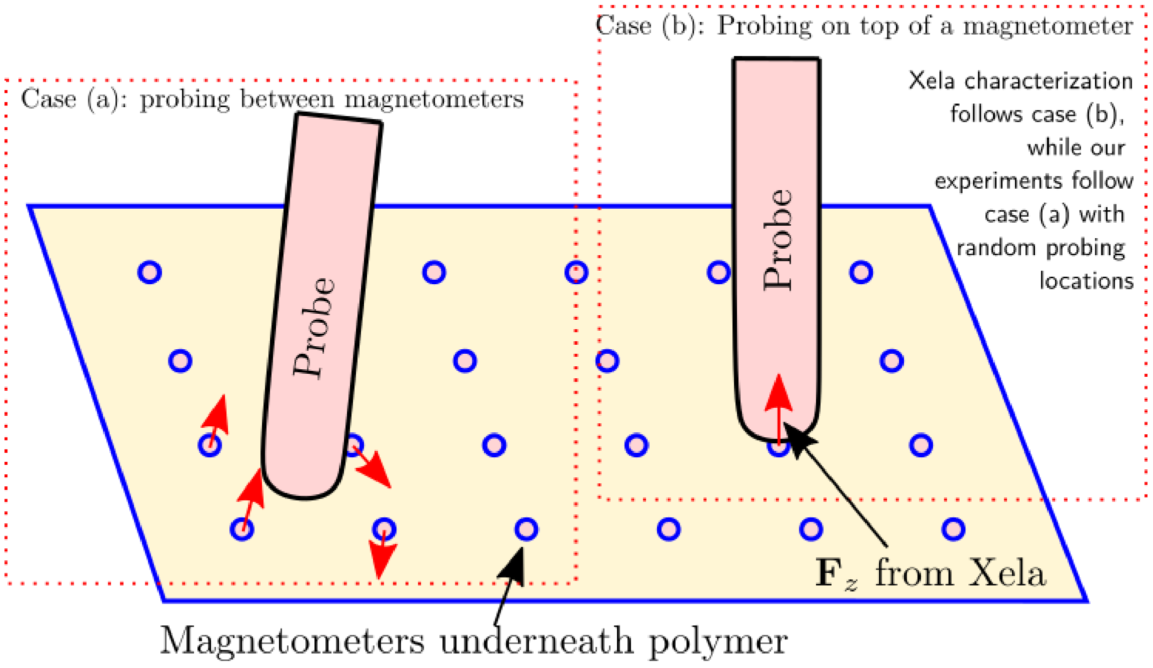

Case (b): Probing on top of a magnetometer
Xela characterization follows case (b), while our experiments follow case (a) with random probing locations
Case (a): probing between magnetometers
Probe
Probe
$\mathbf{F}_z$ from Xela
Magnetometers underneath polymer

Figure 3.12. Illustration of data collection procedure followed in our setup vs procedure followed during Xela force calibration

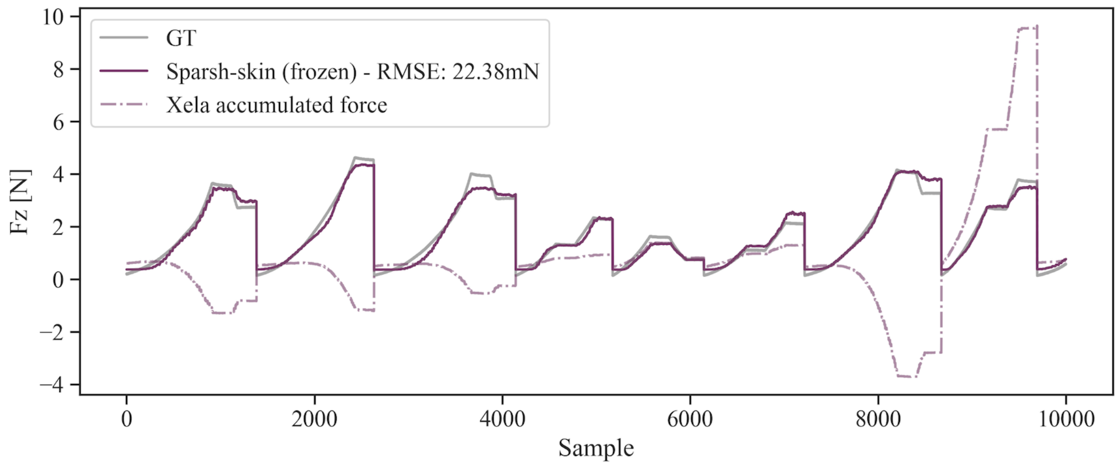


Figure 3.13. Illustration of how the Normal Xela force output is inconsistent with GT force obtained from F/T probe, and the corresponding prediction from `Sparsh-skin`

Figure 3.14. Illustration of data collection protocol follwed for Force estimation with Xela sensors

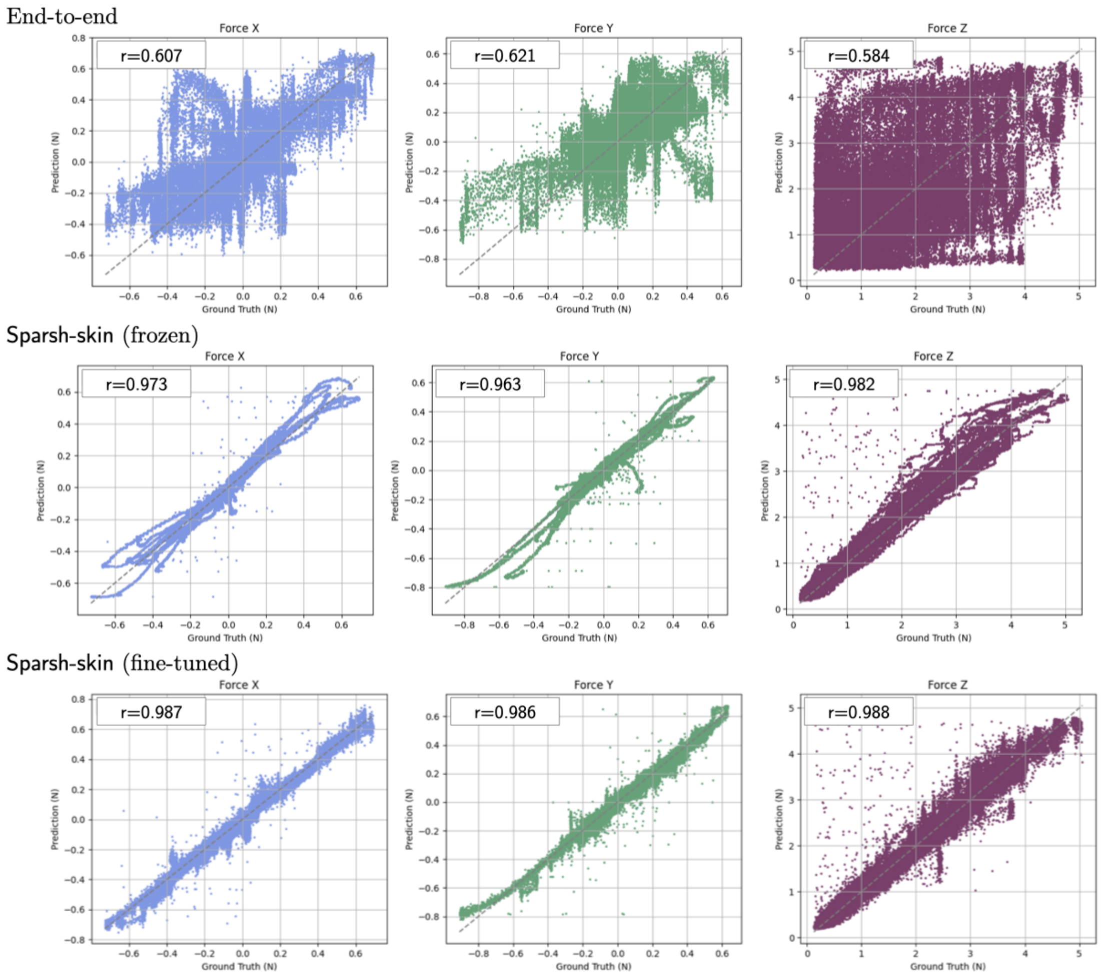


Figure 3.15. Correlation between ground truth and predicted forces on unseen normal loading with an indenter on Xela sensors.

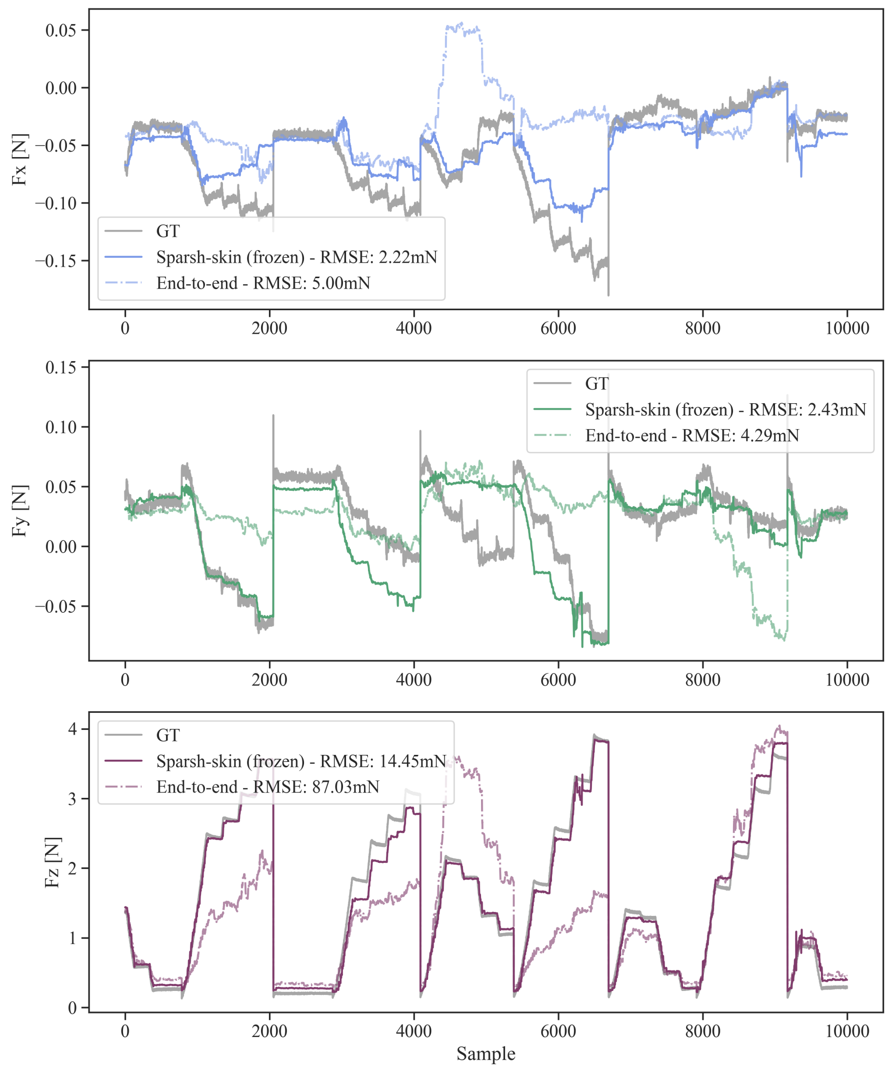


Figure 3.16. Ground truth tangential and normal force from test strokes with flat indenter (gray) and force sequence reconstruction from `Sparsh-skin` (frozen) and end-to-end model.

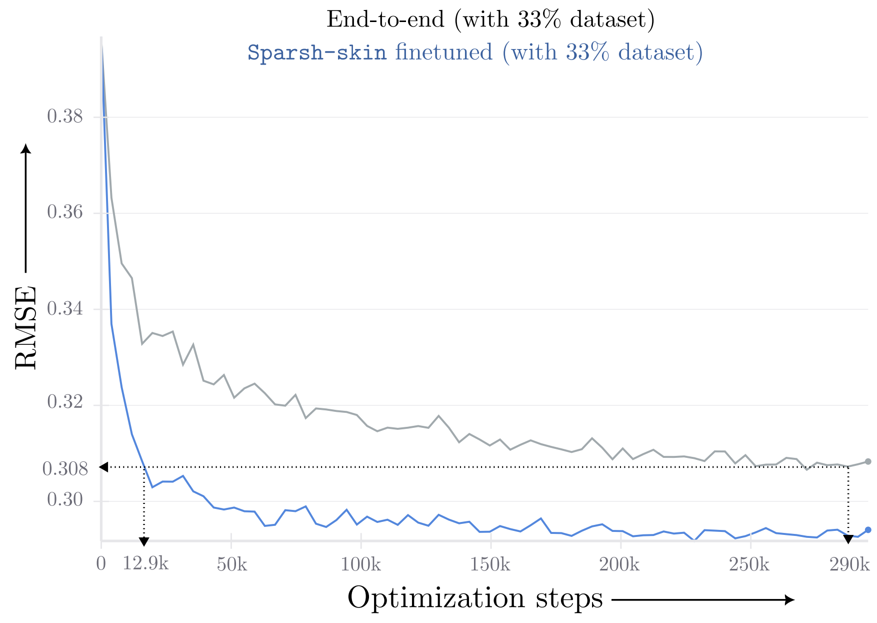


Figure 3.17. Validation RMSE convergence rates between `Sparsh-skin` fine-tuned and `Sparsh-skin` end-to-end: We find that `Sparsh-skin` fine-tuned allows the model to generalize and learn the patterns required to infer joystick states significantly faster during training.

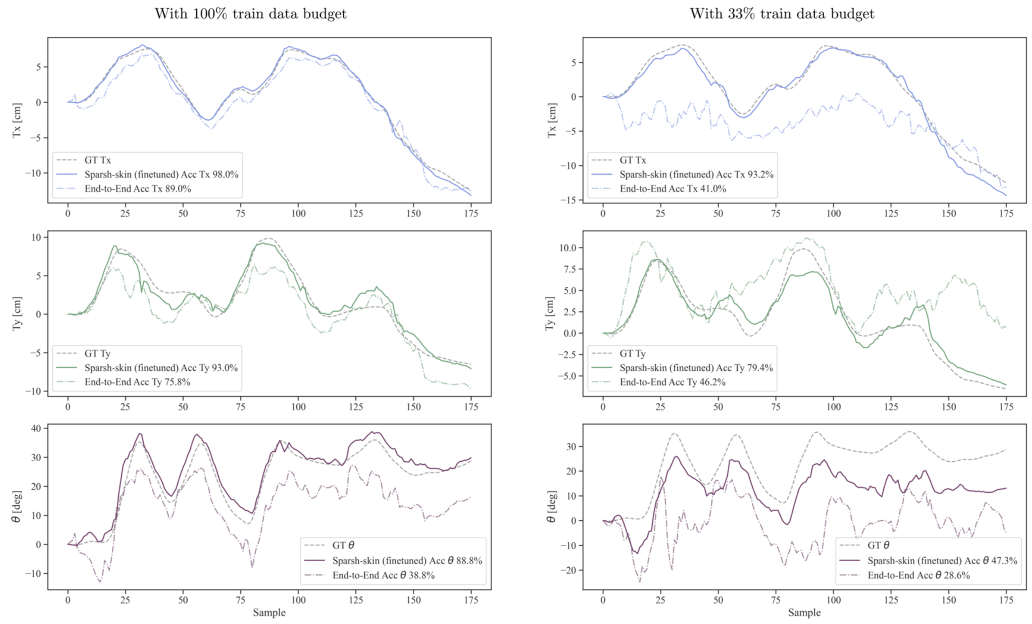


Figure 3.18. Ground truth pose sequence for object in test set and reconstructed trajectory via end-to-end and `Sparsh-skin` (finetuned) representations. (left) Task decoders trained with 100% of train data budget, corresponding to 108 sequences. (right) Task decoders trained with 33% of train sequences.

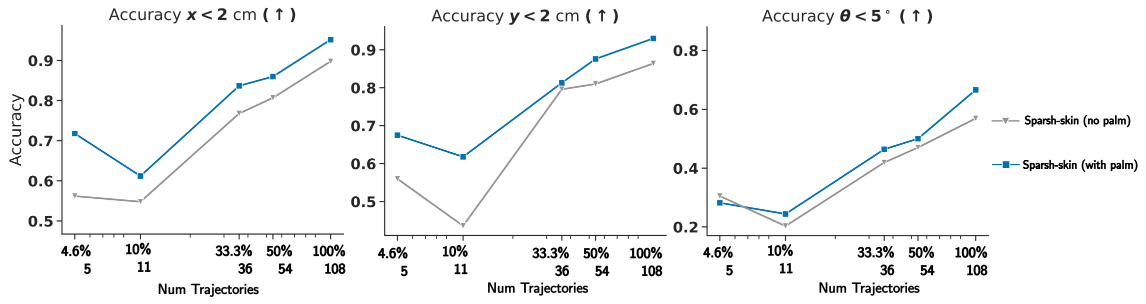


Figure 3.19. Comparison of pose estimation accuracy of `Sparsh-skin` with and without palm sensing.

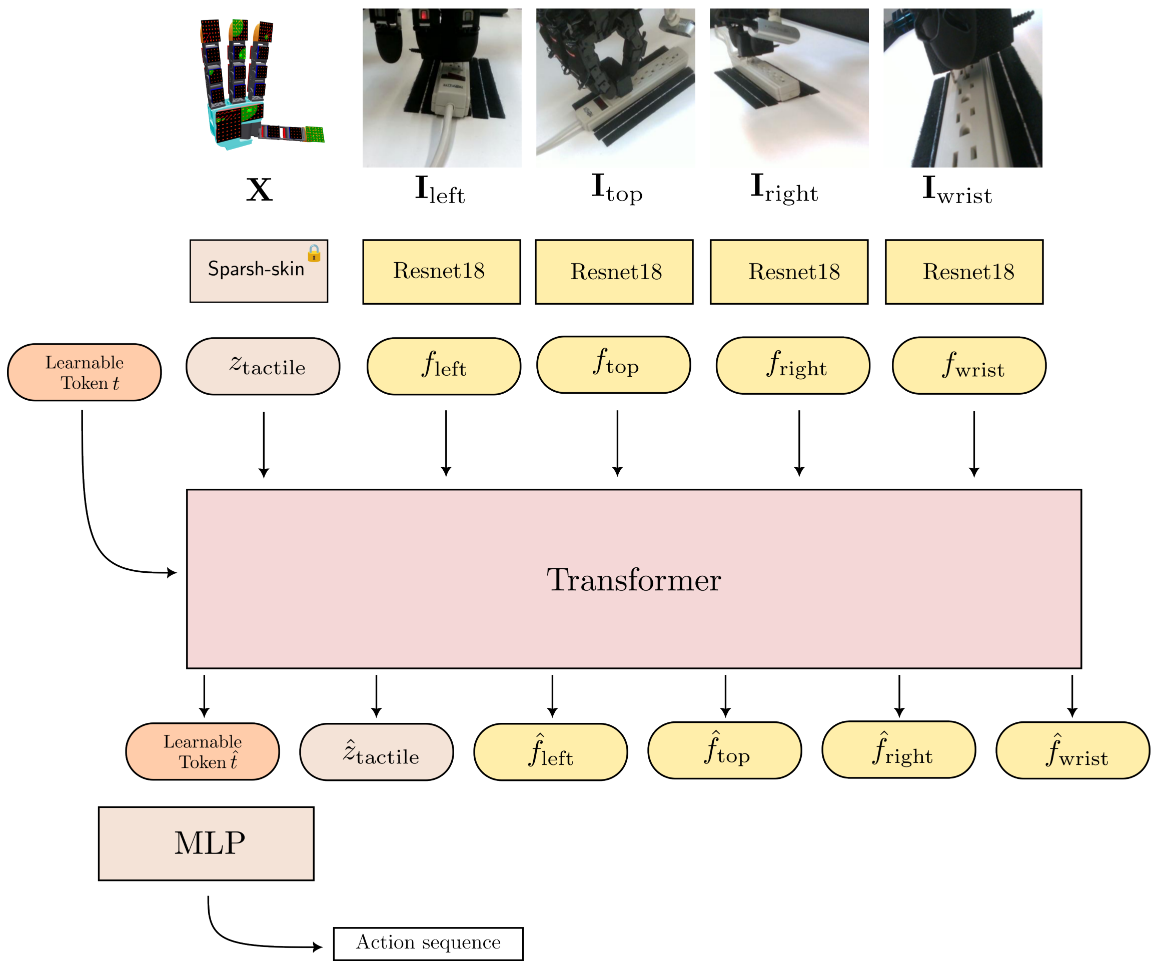


Figure 3.20. Illustration of the policy architecture: We use a transformer to fuse information from visual and tactile modalities, through the use of a learnable action token, which is then used to subsequently predict action sequences.

Chapter 4

Sparsh-X: Multisensory touch representations for robot manipulation

We present `Sparsh-X`, the first multisensory touch representations across four tactile modalities: image, audio, motion, and pressure. Trained on $\sim$1M contact-rich interactions collected with the Digit 360 sensor, `Sparsh-X` captures complementary touch signals at diverse temporal and spatial scales. By leveraging self-supervised learning, `Sparsh-X` fuses these modalities into a unified representation that captures physical properties useful for robot manipulation tasks. We study how to effectively integrate real-world touch representations for both imitation learning and tactile adaptation of sim-trained policies, showing that `Sparsh-X` boosts policy success rates by 63% over an end-to-end model using tactile images and improves robustness by 90% in recovering object states from touch. Finally, we benchmark `Sparsh-X`'s ability to make inferences about physical properties, such as object-action identification, material-quantity estimation, and force estimation. `Sparsh-X` improves accuracy in characterizing physical properties by 48% compared to end-to-end approaches, demonstrating the advantages of multisensory pretraining for capturing features essential for dexterous manipulation.

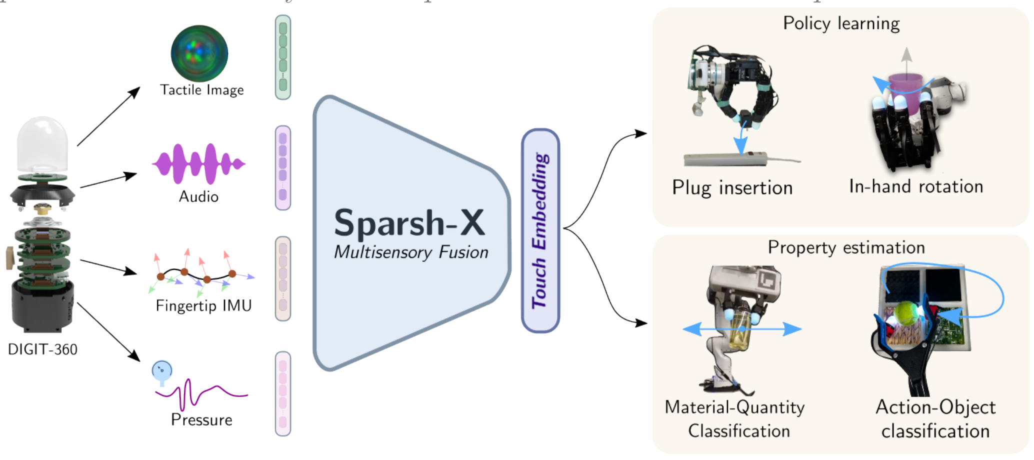


Figure 4.1. **Sparsh, Multisensory Touch Fusion Transformer for General-Purpose Representations.** Touch in robotics can be sensed through multiple modalities, including tactile images, vibrations, motion, and pressure. Sparsh is a transformer-based backbone that fuses these modalities from the Digit 360 sensor. We show its versatility across diverse downstream tasks: manipulation via imitation learning (plug insertion), tactile adaptation (in-hand object rotation), and benchmark tasks to probe the understanding of physical properties.

4.1 Introduction

Touch is a rich and multifaceted sense that plays a central role in human dexterity. Humans fluidly adapt their interactions to the physical properties of objects by integrating a wide spectrum of touch signals that include skin deformation, vibrations, motion, and pressure. This multisensory feedback enables us to distinguish between a plastic and paper cup, twirl a pen between fingers with ease, and manipulate tools under severe visual occlusion. Leveraging the multisensory nature of touch is desirable for robust, fine-grained robot manipulation.

Despite its importance, multisensory touch remains significantly underutilized in robotics. Most approaches rely on unimodal tactile sensing, such as GelSight-like sensors [100, 104, 207], due to standardized hardware availability. However, advances like Digit 360 [102] now enable capturing high-resolution images, vibrations, motion, and pressure in a compact form, making multisensory touch accessible. Prior work on using tactile modalities independently shows promise [123, 204], but a unified, scalable, and easily integrable method to take advantage of these modalities is still lacking. Representation learning offers a viable solution to integrating heterogeneous sensory inputs from sensors like Digit 360 by fusing complementary contact information from all modalities into a shared latent space. In addition, as has been shown for vision-based tactile sensors [69, 82, 194, 215] representation learning allows a downstream task training to be more data-efficient and robust to noise or irrelevant variations [11, 80, 94, 131].

This paper introduces `Sparsh-X`, the first self-supervised backbone for multisensory touch representation learning across four key tactile modalities: image, audio, motion, and pressure. Trained on unlabeled data from diverse manipulation behaviors, such as sliding, tapping, rotating, picking up, and dropping, `Sparsh-X` learns to fuse heterogeneous tactile signals into compact and expressive contact embeddings. Beyond `Sparsh-X`, this unlabeled dataset also facilitates research in representation learning and benchmarking. Through supervised tasks, we show that `Sparsh-X` captures a wide range of physical properties, including object-level characteristics (e.g., type and mass), static contact properties (e.g., force and material), and dynamic interaction cues (e.g., motion and impact). Representations that encode these physical properties at the fingertip level are especially valuable for dexterous manipulation, as they enable feedback of object and contact state directly in latent space.

Ultimately, tactile signals are valuable only when effectively used in policy learning. This remains a challenge, particularly in reinforcement learning due to the *sim-to-real* gap [111]. We demonstrate that `Sparsh-X` can be effectively applied in policy training via two examples: (1) imitation learning, and (2) tactile adaptation of policies trained in simulation with privileged access to contact information. Experiments across manipulation tasks, including insertion and in-hand rotation, demonstrate that integrating `Sparsh-X` leads to significantly improved real-world performance over end-to-end tactile image baselines. By unifying multisensory touch in a shared latent space, `Sparsh-X` takes a step toward foundation models for touch, enabling scalable and data-efficient learning for fine-grained robotic manipulation. Our key contributions are:

1. `Sparsh-X`, the first unified backbone for multisensory touch: fusing image, audio, motion, and pressure signals into a general-purpose representation. Trained on $\sim$ 1M unlabeled samples from Digit 360, `Sparsh-X` enables scalable and transferable touch perception.
2. The first Digit 360 dataset curated for benchmarking multisensory touch representations, allowing interpretability analysis in terms of contact dynamics and physical properties of the object.

3. An empirical demonstration of `Sparsh-X` enhancing real-world policy learning performance and robustness enabled by *tactile adaptation* for fine-grained manipulation skills like insertion and in-hand rotation.

4.2 Related Work

Vision-based tactile sensors [44, 99, 104, 208] have been widely used in contact-rich tasks, including material and volume prediction [68, 210], shape inference [58, 166], localization [88, 165], insertion [43], and contour-following [3, 195] among others. While tactile sensing has been primarily vision-based, other modalities such as audio have also been used independently for capturing object properties [37, 55] and dynamic behaviors [171], though audio alone may be insufficient for perceiving continuous interactions such as forces, deformations, and motion. Subsequently, prior work has also explored audio-visual learning [113, 123] as a natural extension. While audio and vision modalities augment the tactile state complimentarily, additional modalities such as fingertip motion and accumulated pressure can provide additional information for detecting shear forces, recognizing object properties, and predicting object slip and pose changes.

Self-supervised learning (SSL) has been effective for developing tactile image representations [52, 69, 82, 194, 215], allowing better performance and data-efficiency in downstream tasks. Other approaches that use additional tactile modalities such as audio, often rely on using task-specific data to fine-tune pre-trained encoders (e.g. AST [66] and BYOL-A [129] for audio) or joint audio-visual encoders [124]. MULSA [106] introduced multimodal transformers integrating vision, tactile image, and audio from contact microphones by treating all signals as RGB images, but suffers from quadratic complexity of pairwise attention since it simply concatenates the tokens from all modalities. MimicTouch [204] proposed unimodal SSL for tactile images and audio separately, without explicit cross-modal fusion. In contrast, we propose `Sparsh-X`, a multisensory framework that fuses image, audio, motion, and pressure signals via bottleneck self-attention [127] into a shared latent space. This enables to capture contact properties for downstream tasks, while also reducing computational complexity compared to vanilla transformer-based concatenation strategies.

The Digit 360 sensor. Digit 360 [102] offers multisensory touch sensing in a compact fingertip form factor, making it well-suited for dexterous robotic hardware. Inside a dome-shaped elastomer, it integrates a hyper-fisheye camera, contact microphones, IMU, static pressure sensors, and more. While it marks a significant step toward standardizing touch sensing, its unique form factor introduces new challenges. For example, the soft hemispherical dome deforms and moves upon contact, complicating shear force estimation. Additionally, the use of a hyper-fisheye lens and directional lighting limits the applicability of photometric surface reconstruction methods like Poisson integration [208]. Despite these limitations, learning touch representations through scale pretraining offers a promising path to overcome such challenges.

4.3 Learning Multisensory Touch Representations

Fusing tactile modalities is crucial to discovering correlation between modes. For instance high-frequency audio and tactile image can both indicate making/breaking contact. However, traditional tactile sensing work has largely relied only on unimodal approaches, i.e., on tactile images of elastomer deformations [99, 208]. In cases, where additional modalities are considered [106, 204], they have been treated independently. In this section, we introduce `Sparsh-X`, a backbone for general multimodal touch representations for the Digit 360 [102] sensor. Our model integrates four tactile modalities: image, audio, accelerometer, and pressure. Through self-supervision on $\sim$ 1M unlabeled contact interactions, `Sparsh-X` compresses contact information into a unified multimodal representation.

Inputs and Model Architecture. `Sparsh-X` is a transformer-based backbone [47] (see Figure 4.2) where each input signal is first processed independently for L_f layers through self-attention. Thereafter, we allow cross-modal information flow via attention bottlenecks, as in [127]. Specifically, we concatenate B bottleneck fusion tokens to each modality's embedding for the subsequent L_b blocks. After each cross-modal update, the fusion tokens are averaged across modalities to promote information sharing. Intuitively, the bottleneck tokens act as multimodal summarizers, distilling and exchanging information between tactile modalities within each transformer block. Following experimental insights from [127], we set the total number of transformer layers to $L = L_f + L_b = 12$, with $L_f = 8$ layers for unimodal processing

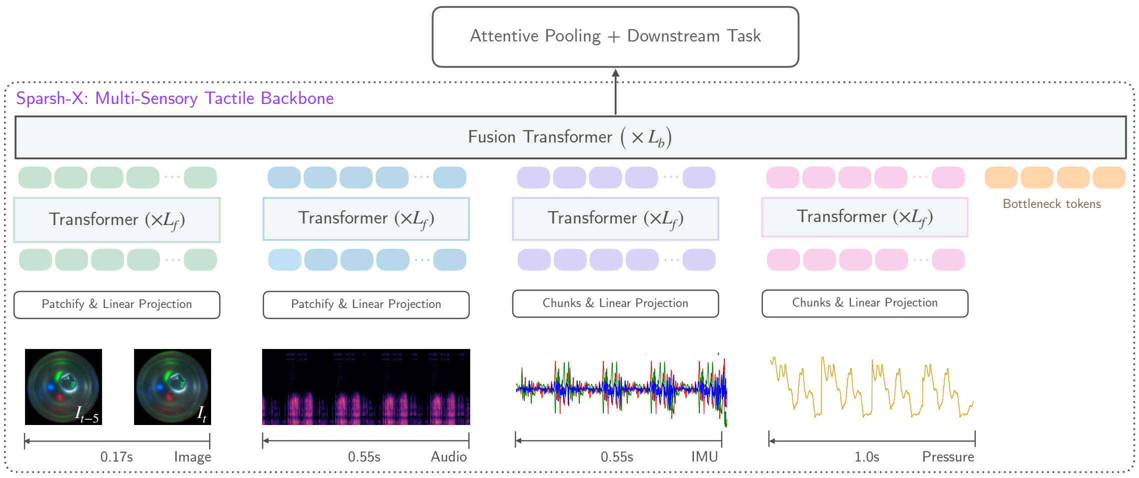

Attentive Pooling + Downstream Task
Sparsh-X: Multi-Sensory Tactile Backbone
Fusion Transformer $(\times L_b)$
Transformer $(\times L_f)$
Transformer $(\times L_f)$
Transformer $(\times L_f)$
Transformer $(\times L_f)$
Bottleneck tokens
Patchify & Linear Projection
Patchify & Linear Projection
Chunks & Linear Projection
Chunks & Linear Projection
I_{t-5}
I_t
0.17s
Image
0.55s
Audio
0.55s
IMU
1.0s
Pressure

Figure 4.2. `Sparsh-X`, a multisensory touch transformer for general-purpose representations, integrates four tactile inputs: image, audio, accelerometer, and pressure. Each modality is processed independently in the first L_f layers, then fused using bottleneck tokens for cross-modal attention in the final L_b layers.

and $L_b = 4$ fusion layers with $B = 4$ bottleneck tokens.

The inputs to `Sparsh-X` are image, audio, accelerometer, and pressure recorded by the Digit 360 sensor. Since all modalities have different sampling frequencies and data structures, we describe the steps for preprocessing and tokenization. *Tactile images* are sampled at 30fps [82] and passed to the model with a temporal stride of 5 concatenated along the channel dimension. We crop to zoom-in the fish-eye image and resize to $224 \times 224 \times 3$. Image patches (16×16) are then tokenized to embeddings of 768 dimensions through a linear projection layer. *Audio* comes from two contact microphones sampled at 48kHz. A 0.55s window of audio signal is converted into a log-mel spectogram of 128 channels computed from a 5ms Hamming window with hop length 2.5ms. We concatenate the spectograms from both microphones, resulting into an audio input of 224×256 which is further tokenized with a patch size of 16. *IMU* data from the 3-axis accelerometer is sampled at 400Hz and combined in a window of 0.55s. The *pressure* signal is sampled at 200Hz and combined in a window of 1.1s window. Both signals are tokenized resulting in 224×3 and 224×1 temporal signals.

SSL Training Pipeline. We train `Sparsh-X` using Self-Supervised Learning (SSL) which offers several benefits, including the ability to learn general representations,

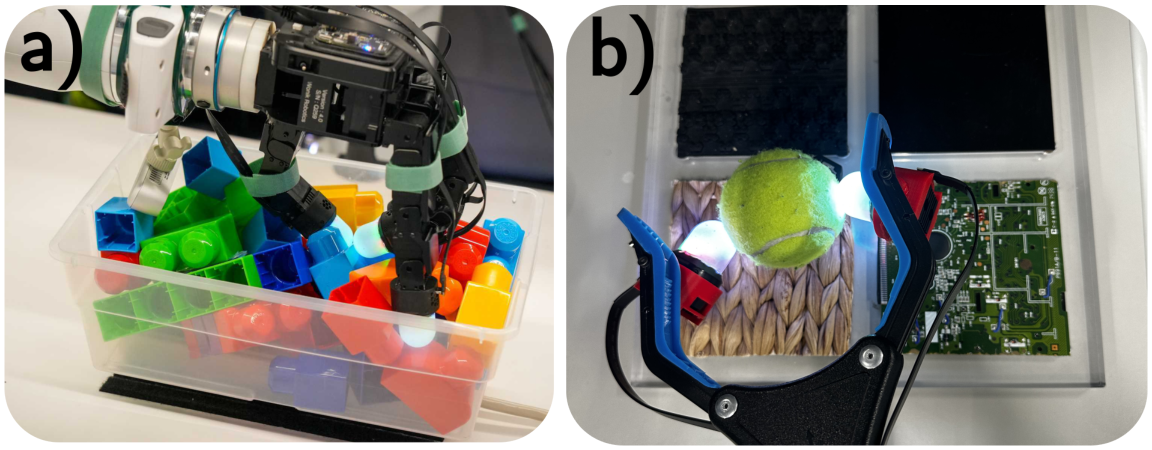


Figure 4.3. Pretraining data collection setup using (a) Digit 360 and Allegro hand (b) two-fingered manual picker.

robustness to distractors, and independence from labeled data. Our SSL training dataset consists of $\sim$ 1M samples generated from two primary sources: an Allegro hand with Digit 360 sensors on the fingertips that performs random motions with objects such as dipping into a tray filled with various items; and a manual picker [33, 150, 203] with the same sensor adapted to the gripping mechanism, used to execute atomic manipulation actions such as picking up, sliding, tapping, placing, and dropping objects against diverse surfaces that vary in roughness, hardness, softness, friction, and texture properties. We employ a teacher-student self-distillation approach [22, 131], where both branches consist of an encoder and a predictor head. After tokenizing each multisensory touch input, appending a register token, and adding sinusoidal positional embeddings, we apply masking to the student input tokens per modality, retaining 10-50% of the signal for local masks and 50-100% for global masks. We concatenate the register tokens from global and local masks and pass them through their corresponding prediction heads. As in [22], the prediction task involves clustering, where teacher tokens serve as pseudo-labels for the student network, with centroids that adapt over time as the model learns. We use cross-entropy between the softmax outputs of the teacher and student networks as optimization objective. We train `Sparsh-X` for 200 epochs on 16 A-100 GPUs, with 128 batch size and AdamW optimizer with linear rampup followed by a cosine schedule as the learning scheduler. Please refer to the Appendix for further pretraining details.

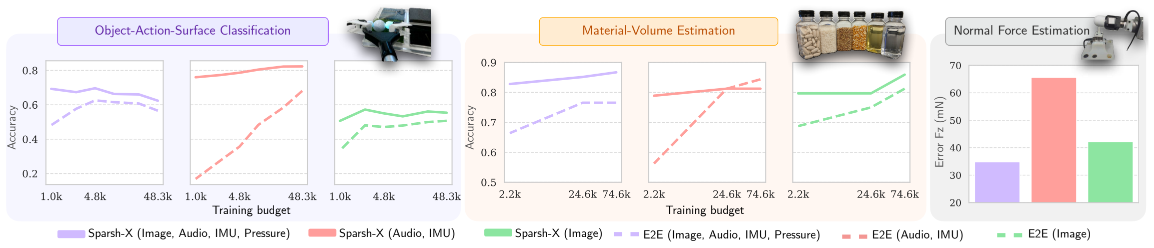


Figure 4.4. Performance of frozen `Sparsh-X` representations with different tactile inputs. The synergy of multiple modalities improves object-action-surface identification (*left*) and material-quantity estimation (*middle*), outperforming tactile image alone and showing data efficiency over E2E. Combined modalities also enhance normal force estimation (*right*), a task typically addressed with vision-based tactile sensing.

4.4 Integrating Multisensory Touch Representations in Downstream Tasks

We propose a set of downstream tasks for evaluating the capability and generalization of our representations on touch-centric tasks. Our study is driven by two central research questions, first, *what physical properties do our representations capture from contact interactions?*, and second *how can real-world touch representations be leveraged for manipulation policy learning?*.

4.4.1 Inferring physical properties with Sparsh-X

We design supervised tasks to evaluate `Sparsh-X` representations' ability to capture physical properties for robotic manipulation. These tasks cover (1) object-level characteristics (e.g., type and quantity), (2) static contact properties (e.g., force and material), and (3) dynamic interaction cues (e.g., sliding, tapping). For each task, we train a task-specific attentive decoder [28] in a supervised manner, using as input `Sparsh-X` representations. Importantly, the encoder weights of `Sparsh-X` are frozen to isolate and assess the quality of the representations learned from self-supervised pretraining.

Object-Action-Surface Classification. We train the decoder on the subset of the pre-training dataset that uses the manual picker to jointly classify the object being grasped (golf ball, LEGO, or wood block), the action being performed (planar sliding, circular sliding, or tapping), and the extrinsic surface in contact with the object (plastic, fabric, grass, or formwork). This task probes whether the representations encode static contact properties like friction, stiffness, and roughness, as well as dynamic cues that distinguish between different motion patterns. To assess the benefits of multisensory touch, we ablate the input modalities to `Sparsh-X` and compare performance against traditional end-to-end (encoder-decoder) models trained from scratch. As shown in Figure 4.4 (left), combining tactile modalities significantly boosts accuracy. For instance, pairing audio with IMU yields a 32% improvement, while using all modalities together provides a 13% gain compared to using tactile images alone. Pre-training further enhances performance when using all modalities, consistently outperforming E2E models with task-specific embeddings, with a particularly notable 10% margin under the lowest data regime.

Material-Quantity Estimation. We evaluate the capacity of our model to distinguish materials (both solids and liquids) and to provide a coarse mass estimation through shaking motions [36, 87, 110, 121]. We train an attentive decoder to jointly classify material type and quantity by shaking 8oz bottles with a parallel gripper equipped with Digit 360 sensors (see Figure 4.1). The dataset includes four solids (corn kernels, lentils, vitamin pills, rice), two liquids with distinct viscosity (water, oil), and three fill levels (full, half, quarter). As shown in Figure 4.4 (middle), `Sparsh-X` representations from all modalities achieve the highest accuracy across all training data budgets, outperforming end-to-end models trained on tactile images alone by 20.5%. `Sparsh-X` representations outperform end-to-end models across all sensory inputs, demonstrating superior data efficiency and generalization.

Normal Force Estimation. Following the protocol from [82], we use a hemispherical probe to apply normal forces of up to 3.5N to the Digit 360 sensor by indenting perpendicularly into the elastomer surface. An attentive regression head is trained to estimate the applied normal force from frozen `Sparsh-X` representations across various tactile sensory inputs. Evaluation is performed on samples with randomized force magnitudes and indentation locations, using the same probe geometry. As shown in Figure 4.4 (right), combining all tactile modalities leads to improvement in force estimation accuracy, achieving an average error of 35mN, a 17% improvement over using only tactile image.

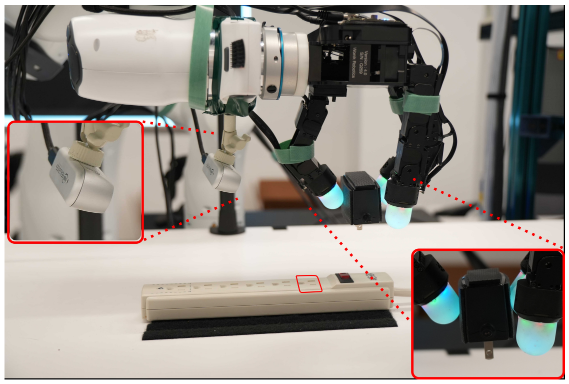
4.4.2 Sparsh-X for Policy learning

We investigate how manipulation policies can benefit from touch representations that capture physical properties like friction, mass, and forces. We demonstrate their effectiveness in real-world (a) imitation learning and (b) tactile adaptation of sim-trained policies, evaluating on two contact-rich tasks: plug insertion and in-hand rotation.

Plug-Insertion via Imitation Learning

Insertion is a fundamental skill in robot manipulation and has long served as a benchmark task in the literature [43, 152, 168, 187]. We evaluate the utility of multisensory touch representations in a plug-insertion task, where a robot equipped with an Allegro hand and Digit 360 sensors must insert a pre-grasped plug into a fixed socket. Using kinesthetic teleoperation, we collect 100 demonstrations with randomized initial arm poses, recording joint states, wrist poses, camera images, and tactile data.

Our policy architecture is adapted from ACT [218]. The inputs include wrist camera embeddings, obtained by training a vision encoder, and Sparsh-X representations for the thumb, index, and middle fingers. These tactile features are aggregated using an attentive pooling layer [28]. The model predicts a trajectory of absolute end-effector poses over a horizon of $H = 8$.



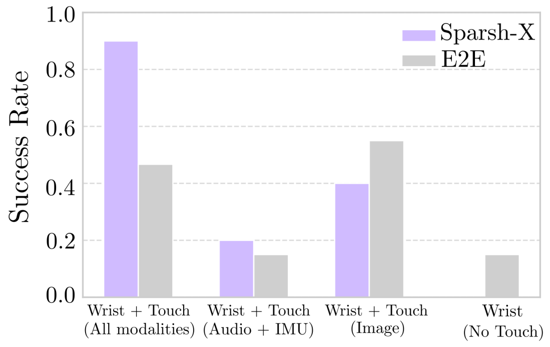


Figure 4.5. **Left.** Experimental setup for plug-insertion. **Right.** Success rate over 20 trials using different tactile sensory modes. Leveraging multimodal touch with Sparsh-X improves performance by 500% over external-vision-only and 63% over E2E tactile-vision-only policies.

Evaluation. To evaluate the contribution of multisensory touch, we ablate the policy by varying the combination of tactile sensory inputs used, always in conjunction with the wrist camera. We also include a vision-only baseline without touch. Each policy is evaluated over 20 trials, with randomized wrist starting positions.

As shown in Figure 4.5, multisensory touch is key for achieving high performance in a tight-tolerance insertion task, with a 90% success rate. Pretraining plays a crucial role, yielding a 90% performance boost compared to training representations with all modalities from scratch jointly with the policy model on task-specific data. These results highlight the benefits of both multimodality and pretraining. Access to multiple sensing modalities enables better discrimination of subtle contact cues. For example, audio can signal initial contact or collisions, while tactile images and pressure provide information about normal and shear forces that are critical for alignment and insertion.

Policies using tactile images outperform those relying solely on audio and motion cues. Interestingly, for tactile images, end-to-end training outperforms using frozen pretrained representations. The tactile image signal varies little between trials, allowing a specialized encoder to focus on subtle changes in contact patch location and size. It is worth noting, however, that these encoders are evaluated in-distribution, suggesting that pretrained representations may still benefit from increased data diversity and scale or fine-tuning. Notably, our results show a 63% improvement in performance when using Sparsh-X with all modalities compared to an end-to-end policy trained with tactile image alone. While pretrained tactile image encoders may benefit from broader data diversity, combining touch modalities helps mitigate such limitations, given their complementary nature. As a final remark, incorporating touch significantly improves performance on this task. In particular, the wrist-only policy often fails due to visual aliasing, where insufficient camera parallax results in plug pegs incorrectly appearing directly on top of socket openings, leading to failed insertions.

In-Hand rotation with sim-to-real tactile adaption

A common strategy for learning dexterous manipulation policies is to first train a base policy using privileged information typically available only in simulation (e.g., object physical properties like mass and friction, or contact signals like location and forces)

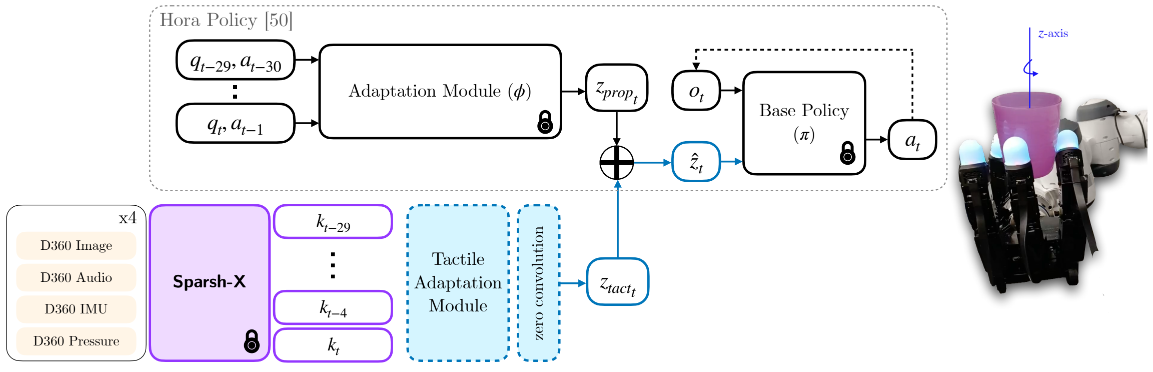


Figure 4.6. We introduce real-world tactile adaptation of sim-trained policies via ControlNet [214], where the zero-convolution layer enables gradual fine-tuning of the embedding $\hat{z}_t$ using `Sparsh-X` representations.

and later distill the information into another model using inputs only accessible in the real-world such as proprioception [98]. This raises a natural question: *when richer information becomes available in the real world, how can we bring policies trained in simulation closer to the privileged information setting?*

We explore sim-to-real tactile adaptation in the context of in-hand rotation. As our base policy, we use Hora [140], a proprioception-only policy for rotating objects along the z-axis. Hora is trained in simulation via rapid motor adaptation [98], leveraging privileged information such as object pose, shape, mass, friction, and other properties that can be perceived through touch at the fingertips. Since `Sparsh-X` captures physical properties, our goal is to do *tactile adaptation* on top of Hora, improving stability during rotation by reducing slip.

We propose *tactile adaptation* via ControlNet [214], which allows the integration of new control modalities without retraining the base model. Applied to policy learning, ControlNet ensures that performance does not degrade below that of the original Hora policy. As shown in Figure 4.6, we learn a tactile adaptation module that connects to the frozen base policy through a zero-initialized convolutional layer, enabling progressive integration of tactile information. Specifically, we feed to the tactile adaptation module `Sparsh-X` representations of all four fingertips over the past 1.5 seconds, aligning with the proprioceptive history window used by the base policy.

Training. We rollout the baseline Hora policy (using the open-sourced policy checkpoint) to collect real-world sequences, capturing proprioceptive joint states, target actions, and tactile data related to the in-hand rotation of cup-like objects. For training, we select 50 successful trajectories in which the object remains stably rotating along the z-axis for at least 30 seconds. The tactile adaptation module is then optimized to minimize the L2 loss between the real-world joint angles and the target action output from Hora with the ControlNet.

Evaluation. We compare the baseline Hora policy against two tactile adaptation variants: Hora+ControlNet(`Sparsh-X`), using pre-trained `Sparsh-X` representations with different combinations of tactile modalities, and Hora+ControlNet(E2E), trained end-to-end. To isolate the impact of tactile feedback over improvements from rejection sampling, we also evaluate against Hora fine-tuned on real-world data and a proprioception-only imitation learning baseline. The primary goal is to improve stability during in-hand rotation by reducing slip, where the object either shifts into the palm or is completely dropped. We evaluate each policy based on vertical drift and time-to-fall, performing 10 trials per policy with a maximum episode duration of 60 seconds.

Our ControlNet approach with `Sparsh-X` representations, reduced vertical translation by 90% (see Figure 4.7 (top)), using either all modalities or just tactile images. While Hora+ControlNet(E2E) also improved stability a bit, it slowed rotation. Critically, our method outperformed both finetuned Hora and the proprioception-only imitation learning baseline, demonstrating that the benefit stems from tactile feedback, not just good demonstrations. The imitation learning baseline was unreliable, frequently failing due to out-of-distribution states as reflected in the lowest time-to-fall metric.

We also evaluate model robustness to altering the physical properties of the object, specifically friction and mass as shown in Figure 4.7 (middle). When friction is reduced, Hora+ControlNet(`Sparsh-X`) outperforms all other policy variants. By using a synergy of all tactile modes, it can maintain object stability without losing grasp. In contrast, Hora+ControlNet(image) struggles, possibly indicating that changes in the contact patch are too subtle to be captured by `Sparsh-X` from tactile images alone. This result underscores the complementary strengths of multisensory touch. When the object's mass is increased, Hora+ControlNet(`Sparsh-X`) and Hora+ControlNet(image) allow the baseline policy to adapt its finger gaiting to compensate for the added weight. This adaptation is possible because the tactile properties of the object can

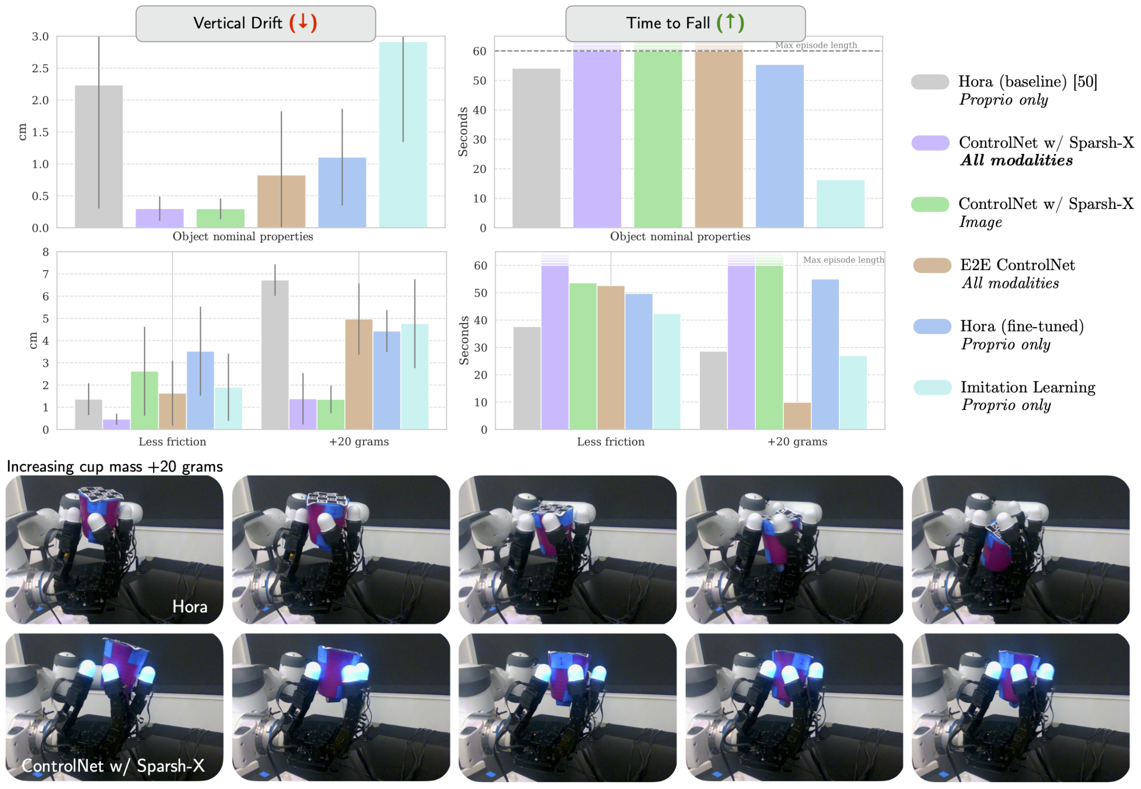


Figure 4.7. **Top.** For object nominal properties, tactile adaptation with `Sparsh-X` reduces vertical drift by 90% compared to Hora. Fine-tuning with successful rollouts does not yield same performance, highlighting the effectiveness of tactile adaptation. Metrics from 10 trials (60s episodes). **Middle.** Under dynamical changes, tactile adaptation shows superior stability than Hora variants. Metrics from 5 trials (60s episodes). **Bottom.** Snapshots of policy rollouts with and without multisensory touch input as object mass increases.

be captured by `Sparsh-X` and transferred to the privilege information embedding in the latent space.

4.5 Discussion and Conclusion

We present `Sparsh-X`, a self-supervised backbone for general multisensory touch representations. Through both policy learning and supervised tactile experiments, we demonstrate that incorporating multisensory touch from Digit 360 and scale pretraining over $\sim 1M$ samples significantly enhances task performance compared to using end-to-end approaches with tactile images alone.

Our study is driven by two central research questions. First, *how can real-world touch representations be leveraged for manipulation policy learning?* `Sparsh-X` pretraining enables better and more robust policies. We explore two approaches: imitation learning (IL) and tactile adaptation of sim-trained policies. IL is naturally suited as demonstrations capture rich tactile signals from all modalities provided by Digit 360. For sim-policies, we propose tactile adaptation via ControlNet, enabling the propagation of tactile information previously accessible only in simulation as privileged information. We validate these approaches on two fundamental manipulation tasks: plug insertion and in-hand object rotation. Notably, `Sparsh-X` enhances policy performance by 63% over policies using tactile images alone and improves robustness by 90% by using touch to recover object state during manipulation.

Second, *what tactile properties do our representations capture?* We find that `Sparsh-X` representations effectively captures physical properties that allows to identify objects, actions, surfaces, estimate intrinsics properties and forces from multisensory touch signals. We evaluate `Sparsh-X` on a suite of supervised benchmark tasks common in the literature: object-action recognition, material-quantity estimation, and force prediction. We perform ablations over tactile input modalities and training data budgets to assess the impact of pretraining and multisensory fusion. Our results show that a synergy of touch from images, audio, IMU, and pressure, leads to higher accuracy across all tasks even in low-data regimes. Compared to training end-to-end with tactile images only, `Sparsh-X` achieves an average improvement of 48% across all tasks, demonstrating the benefits of both pretraining and touch sensory fusion.

Limitations

While our study highlights the benefits of multisensory touch and self-supervised pretraining, some limitations remain. Each modality introduces its own challenges in capturing a broad and diverse set of contact interactions at scale. In our pretraining dataset, the tactile image modality from the Digit 360 sensor exhibits the lowest diversity in terms of number of different devices used with their own optical artifacts, potentially limiting its generalization in downstream performance. We believe that as the community increasingly adopts multisensory tactile sensors like the Digit 360, collaborative efforts can help build larger and more diverse datasets to support scalable pretraining. Additionally, our experiments focus exclusively on frozen `Sparsh-X` representations to understand the pure impact of pretraining on generalization. However, allowing fine-tuning with task-specific data could further improve performance and help compensate for modality-specific data limitations. Finally, our evaluation of force sensing is limited to normal force estimation under controlled contact conditions. Generalizing to varying contact geometries and multiple simultaneous contacts remains an open area for future work. Moreover, we do not consider shear force estimation in this study, as separating the effects of extrinsic forces from the internal deformation of the elastomer presents non-trivial modeling challenges.

4.6 Appendix

4.6.1 Datasets

Dataset for `Sparsh-X` SSL Pretraining

Our self-supervised learning (SSL) dataset is sourced from two platforms: an Allegro hand equipped with Digit 360 sensors mounted on each fingertip, and a custom mobile picker tool with sensors integrated into its gripping mechanism. Since SSL representation learning does not require labeled data, we collect tactile data by having the Allegro hand interact rummaging freely with a tray filled with LEGO blocks and marbles. This setup enables the capture of rich, multi-contact interactions with

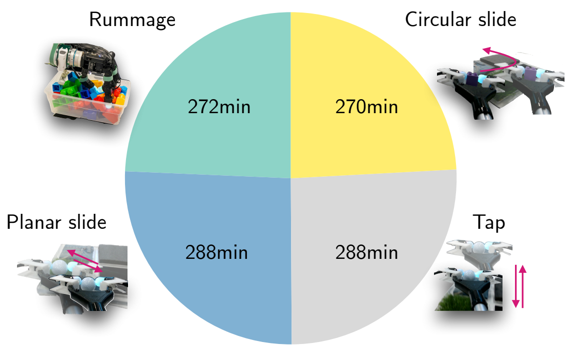


Figure 4.8. Distribution of recorded Digit 360 data by platform. The dataset includes 18.6 hours of data collected using two platforms: the Allegro hand (4.5 hours) and the mobile picker tool (14.1 hours).

objects that feature distinctive geometries, such as spherical shapes and sharp edges.

The mobile picker is used to gather tactile data from everyday manipulation actions, including tapping and sliding, across surfaces with varying friction and stiffness. This allows us to record both intrinsic and extrinsic contact interactions. We collected eight sequences with the Allegro hand, each lasting approximately 8.5 minutes, recording data from all four fingers. From the mobile picker, we gathered 104 sequences with an average duration of 3 minutes, logging data from both sensors on the gripper. For the subset of the dataset collected with the mobile picker, we provide annotations indicating the object in grasp, the action performed, and the surface in contact, to support downstream evaluation tasks. In total (see Figure 4.8), our dataset spans 18.6 hours of tactile data collected from six different Digit 360 sensors.

Sparsh-X processes temporal windows of data from each tactile modality. A visualization of the input data is shown in Figure 4.9.

Images. We input pairs of tactile images sampled with a temporal stride of 5, concatenated along the channel dimension. These images are captured using a hyperfisheye lens in Digit 360, allowing us to view the entire dome-shaped elastomer surface. Unlike planar GelSight-like sensors, these images include reflections from the surrounding LED light sources, visible near the center of the dome. While these reflections act as useful markers, encoding meaningful information about gel deformation upon contact, they also pose challenges for standard preprocessing

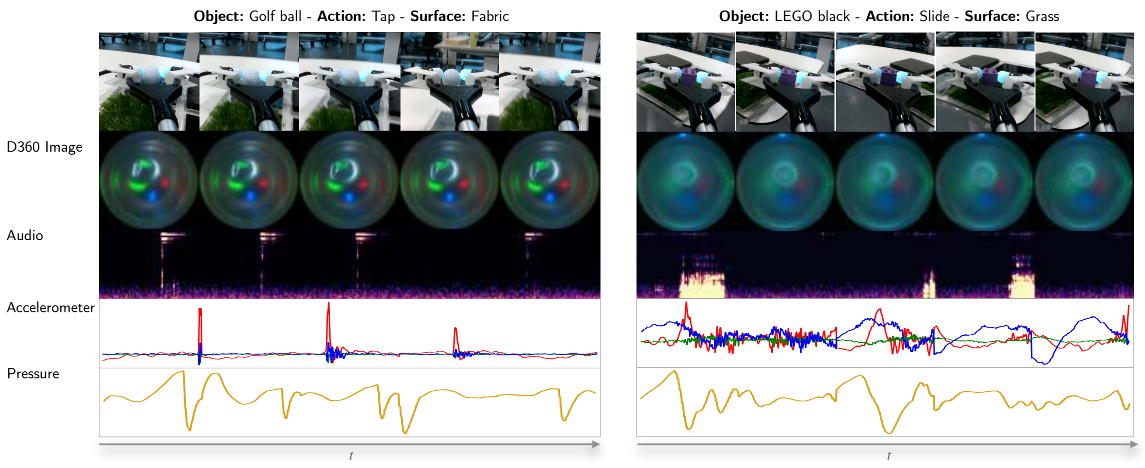


Figure 4.9. Visualization of each of the tactile input modalities to `Sparsh-X`. Samples from pretraining dataset.

techniques such as background subtraction or lighting augmentations, which risk corrupting the contact signal.

Audio. Digit 360 sensor is equipped with two contact microphones that capture vibrations, sampled at 48kHz. This signal is especially informative for detecting changes in contact state, such as making and breaking contact. `Sparsh-X` processes 0.5sec windows of audio data from each microphone in the frequency domain. After standardization and conversion to log-mel spectrograms, the audio is treated as a single-channel image input to the model.

IMU and Pressure. We extract 0.5 second and 1 second windows of data from the 3-axis accelerometer and the static pressure sensor embedded in the Digit 360. Each window is standardized using the mean and standard deviation computed per sequence to ensure consistency across variations in sensor signal amplitude.

Datasets for Downstream Tasks

For each experiment related to estimating physical properties with `Sparsh-X` (see Section 4.4.1), we designed custom setups to collect training data tailored to each task.

For **Object-Action-Surface Classification**, we repurpose the SSL pretraining dataset collected with the manual picker. We annotate each data point with metadata specifying the object being held (golf ball, wood block, LEGO block), the action

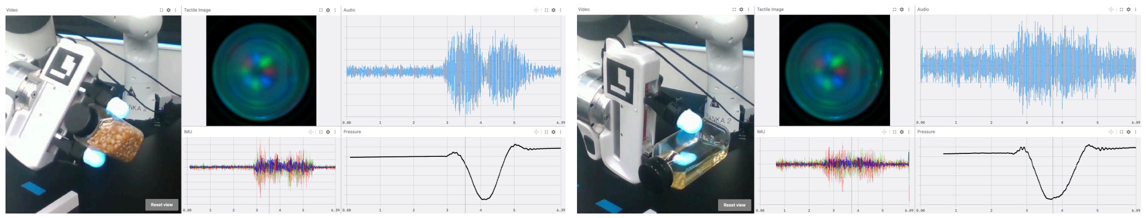


Figure 4.10. Visualization of the experimental setup and tactile sensory inputs for the material-quantity classification dataset. The setup involves shaking bottles filled with different materials and quantities using the Franka's gripper equipped with Digit 360 sensors.

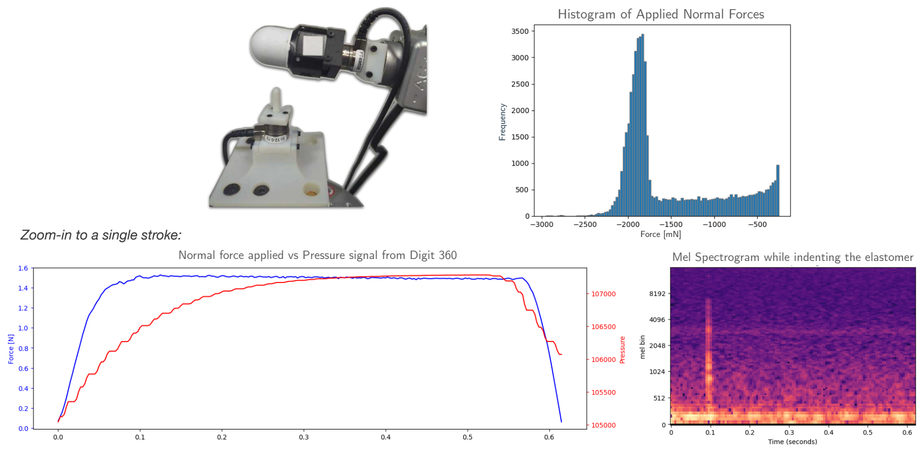


Figure 4.11. **Top:** Experimental setup and data distribution for the normal force regression experiment. **Bottom:** Zoom-in on a single indentation stroke. Note that the pressure signal from the Digit 360 sensor correlates well with the ground-truth normal force measured by the force/torque sensor beneath the hemispherical probe. The mel spectrogram also reveals the moment of contact between the probe and the elastomer.

performed (tap, linear slide, circular slide), and the external surface in contact (grass, fabric, plastic, foamwork). From the 104 available sequences, 69 are used for training and the remaining 35 for testing the performance of the classifier. The dataset is balanced in number of samples per label.

For **Material-Quantity Estimation**, we design a 3D-printed gripper attachment to mount the Digit 360 sensors onto the Franka arm. The data collection protocol involves shaking six different 8oz bottles containing various materials (lentils, rice, corn kernels, vitamin pills, water, and oil) at different fill levels (full, half, quarter). An illustration of this setup is provided in Figure 4.10. For shaking the bottles to create variation in the tactile signal, the Franka's gripper is rotated left and right in randomized motion patterns, including variation in the initial angle. We collect 20 trajectories for each material-quantity combination, using 15 sequences for training and reserving the remaining 5 for evaluating the classifier.

For **Normal Force Estimation**, we fix a hemispherical probe to a force/torque sensor and mount a Digit 360 sensor on the Meca arm, which is used to indent the elastomer surface perpendicularly, applying controlled normal forces of up to 3.5N. Figure 4.11 illustrates the experimental setup and the distribution of the collected data. We observe that the pressure modality correlates strongly with both the magnitude of the applied normal force and the location of the resulting deformation on the elastomer. The audio modality captures discrete events, such as the initial contact between the probe and the sensor surface.

4.6.2 Benchmarking `Sparsh-X` for physical properties comprehension

Object-Action-Surface Classification. This task evaluates whether `Sparsh-X` can capture tactile cues that enable the identification of objects through both intrinsic and extrinsic contact interactions. The goal is to jointly classify the object being grasped, the action performed, and the surface in contact. The selected objects and surfaces span a range of properties, including texture, hardness, and friction.

We use representations from `Sparsh-X` to train a downstream classifier on the dataset described in Appendix 4.6.1. Figure 4.12 shows confusion matrices on the test set for two classifiers: one trained using frozen `Sparsh-X` representations with all tactile modalities as input, and another trained end-to-end (E2E) using only tactile images. Note that the classification task involves 36 classes, representing all combinations of object, action, and surface. The results shown in the figure correspond to training with 50% of the labeled training set.

The `Sparsh-X`-based classifier shows stronger diagonal alignment, indicating more accurate predictions across the 36 joint object-action-surface classes. In contrast, the E2E model suffers from greater confusion among similar classes, particularly those with overlapping surface or action components (e.g., misclassifying "Tap-Foamwork" as "Tap-Fabric" or "Slide-Plastic"). These results highlight the benefit of multimodal

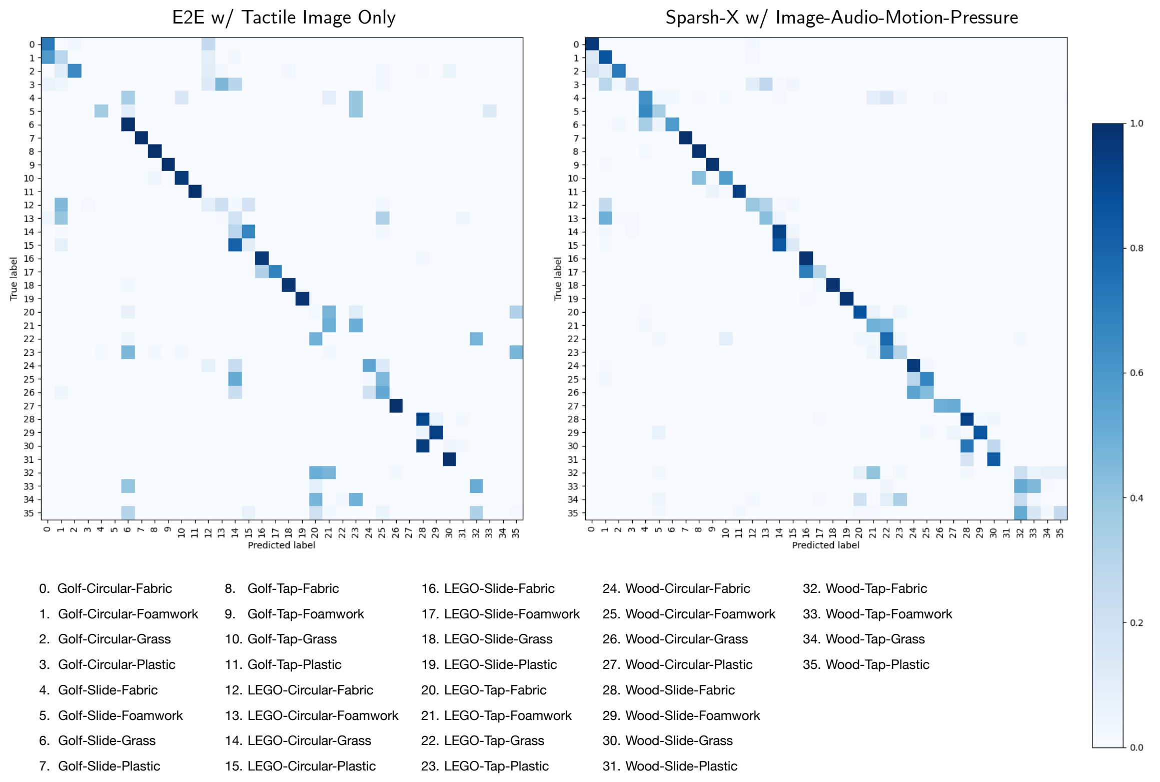
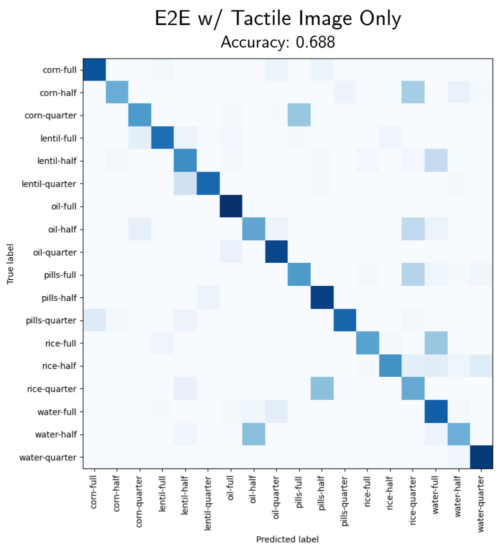


Figure 4.12. Confusion matrix for object-action-surface classification. We compare an end-to-end classifier trained solely on tactile images with a classifier trained on frozen `Sparsh-X` representations, under a 50% training data budget.



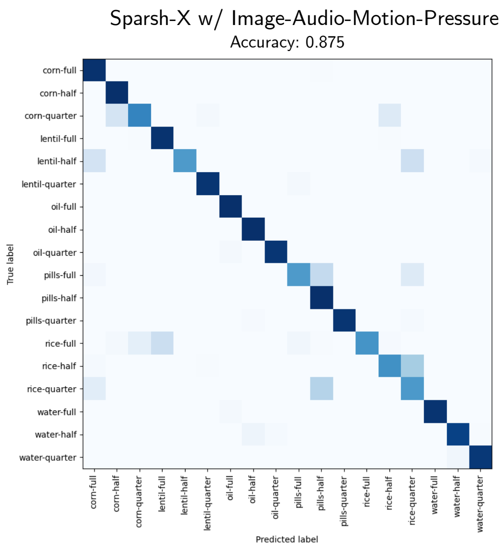


Figure 4.13. Confusion matrix for material-quantity estimation. We compare an end-to-end classifier trained solely on tactile images with a classifier trained on frozen `Sparsh-X` representations, under a 33% training data budget.

tactile representations: incorporating audio, motion (IMU), and pressure modalities helps disambiguate fine-grained contact dynamics that are challenging to capture with images alone.

Material-Quantity Estimation. This task further evaluates `Sparsh-X` 's ability to comprehend physical properties. Specifically, we focus on distinguishing materials based on their granularity and viscosity (e.g., solids and liquids), as well as estimating mass through coarse volume classification. We train a classifier to predict one of 18 joint classes, each representing a unique combination of material type and quantity level. The classifier is trained either using frozen `Sparsh-X` representations or end-to-end (E2E) from tactile images alone.

Figure 4.13 shows the confusion matrices for the material-quantity classification task when trained with 33% of labeled data, comparing an end-to-end (E2E) classifier trained solely on tactile images with a classifier trained on frozen `Sparsh-X` representations. The E2E model achieves 68.8% accuracy, while the `Sparsh-X`-based classifier reaches 87.5%, highlighting the benefit of multimodal tactile representations. In the E2E setting, we observe frequent confusion between different fill levels of the same material and between visually similar liquids such as oil and water. In contrast, the TacX-based classifier exhibits strong diagonal alignment, suggesting accurate identification across the 18 material-quantity classes.

4.6.3 Sparsh-X and Policy Learning

Real-World Sparsh-X and Policy Deployment

For real-world deployment of Sparsh-X, we use ROS2. We maintain circular buffers of 5 seconds for each tactile modality per Digit 360 fingertip. For synchronization, we use the timestamp of the image modality as the reference, selecting the closest-in-time sample from the other modalities (audio, motion from accelerometer, and pressure). Once inputs are processed, Sparsh-X can run inference at 50Hz on a GPU RTX 4090. However, the construction of log-mel spectrograms remains the main computational bottleneck for real-time processing. When processing all four Digit 360 sensors on the Allegro hand, end-to-end inference with Sparsh-X runs at approximately 20Hz. Figure 4.14 shows the deployment pipeline for policy experiments.

Plug-Insertion via Imitation Learning

Training details. The robot, equipped with an Allegro hand and sensorized with Digit 360 fingertips, is tasked with inserting a pre-grasped plug into the first socket of an extension power strip. In our experimental setup, the socket position remains fixed, while the starting position of the robot arm is randomized within a 3D cuboid of $(5, 5, 2)$ *cm* around the nominal starting pose.

The model inputs include an embedding of the wrist camera image and Sparsh-X representations for thumb, index, and middle finger sensors, processed through an attentive pooling layer [28]. We train a ResNet18 [77] in an end-to-end (E2E) fashion to learn the wrist image embeddings. We append learnable action tokens, which are processed by the transformer and subsequently decoded into a sequence of actions. In our setup, the model predicts a sequence of absolute robot end-effector poses i.e., $\mathbf{a} \triangleq (\mathbf{T}_t, \mathbf{T}_{t+1}, \dots \mathbf{T}_{t+H})$ with a prediction horizon of $H = 8$. An illustration of the plug insertion architecture is shown in Figure 4.15.

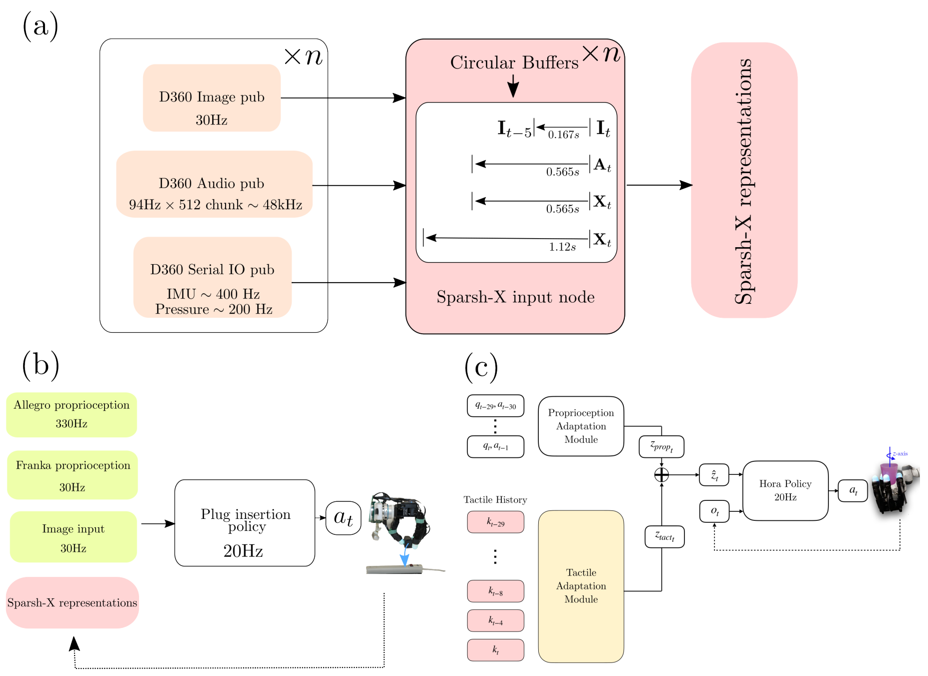


Figure 4.14. Real-world policy deployment architecture: We use ROS2 middleware for policy deployment, and PyTorch for deep learning modules. In addition to the proprioceptive states of the robot and optional third-person vision modality, downstream policies take as input `Sparsh-X` representations from upto 4 fingertips of the Allegro hand. (a) illustrates how the inputs are constructed for TacX, (b) illustrates policy deployment for the plug insertion policy, and (c) illustrates the policy deployment for the in-hand rotation (Hora) policy.

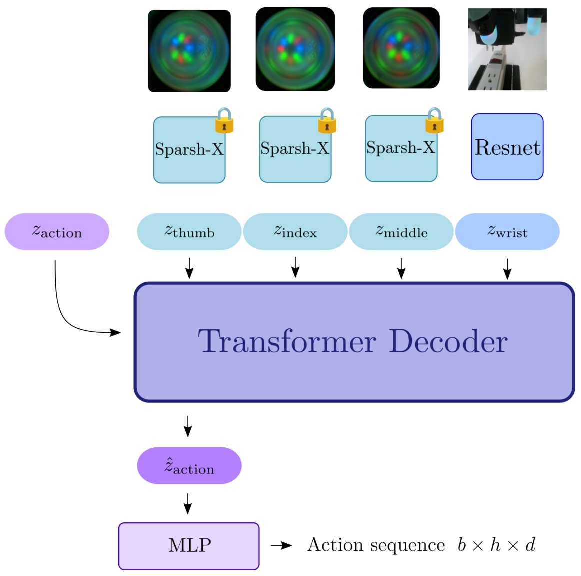


Figure 4.15. Architecture overview of the plug insertion policy. A transformer decoder is trained to generate action sequences based on `Sparsh-X` representations from three fingertips, a wrist camera image capturing the current robot state, and a learnable latent action code. All embeddings are concatenated and processed by a lightweight MLP to decode the next end-effector action.

4.6.4 In-Hand rotation with sim-to-real tactile adaption

Training details. Hora [140] is a two-stage policy that rotates objects along the z-axis. The first stage trains the policy using privileged information, which includes the object's state or pose, local shape, mass, friction, and other physical properties that can be perceived by the fingertips. The second stage trains an adaptation module to approximate the latent space of the privileged information from the discrepancy between observed proprioception history and commanded actions, which implicitly informs about contact.

Although the approximation of the privileged vector from proprioceptions transfers to the real setup, it operates with incomplete information about the object's state. With multisensory touch sensing at the fingertip level, privileged information such as changes in object pose, slip, and friction are now accessible in real-world scenarios, albeit not directly. We can leverage `Sparsh-X` representations to fine-tune the real-world approximation of the privileged information embedding. The goal is to do *tactile adaptation* on top of the baseline policy to enhance stability during object rotation.

We pass to the tactile adaptation module frozen `Sparsh-X` representations for each of the four fingers in the Allegro-hand, with a temporal stride of 0.19s, equating to 8 touch representations per finger over a 1.5s window, which matches the proprioception state history consumed by the baseline Hora. Features for each finger are pooled using attentive pooling to create a global representation, which is then concatenated along the temporal dimension, resulting in a $(t \times n) \times 768$ input embeddings. The tactile adaptation model to be trained is a shallow MLP followed by the zero-convolution layer.

Our dataset consists of successful rollouts of the Hora policy, where the object keeps rotating without touching the palm for at least 30 seconds. The data is serialized into the lerobot dataset format [18], sampled at a control frequency of 20Hz. For training the tactile adaptation module, our objective is to minimize the L2 loss between the real-world hand joint angles and the target action given by the frozen Hora policy under the tactile-informed privileged embedding.

Chapter 5

PTLD: Sim-to-real Privileged Tactile Latent Distillation for dexterous manipulation

Tactile dexterous manipulation is key to automating complex household tasks, yet acquiring effective control policies remains a challenge. While recent work has relied on imitation learning, obtaining high quality demonstrations for multi-fingered hands via robot teleoperation or kinesthetic teaching is prohibitive. Alternatively, with reinforcement we can learn skills in simulation, but fast and realistic simulation of tactile observations is challenging. To bridge this gap, we introduce PTLD: sim-to-real Privileged Tactile Latent Distillation, a novel approach to learning tactile manipulation skills without requiring tactile simulation. Instead of simulating tactile sensors or relying purely on proprioceptive policies to transfer zero-shot sim-to-real, our key idea is that to leverage *privileged sensors* in the real world to collect real-world tactile policy data. This data is then used to distill a robust state estimator that operates solely on tactile input. We demonstrate from our experiments that PTLD, can be used to improve proprioceptive manipulation policies trained in simulation significantly by incorporating tactile sensing. We also show that PTLD enables learning the challenging task of tactile in-hand reorientation where we see a 57% improvement in number of goals reached over using proprioception alone.

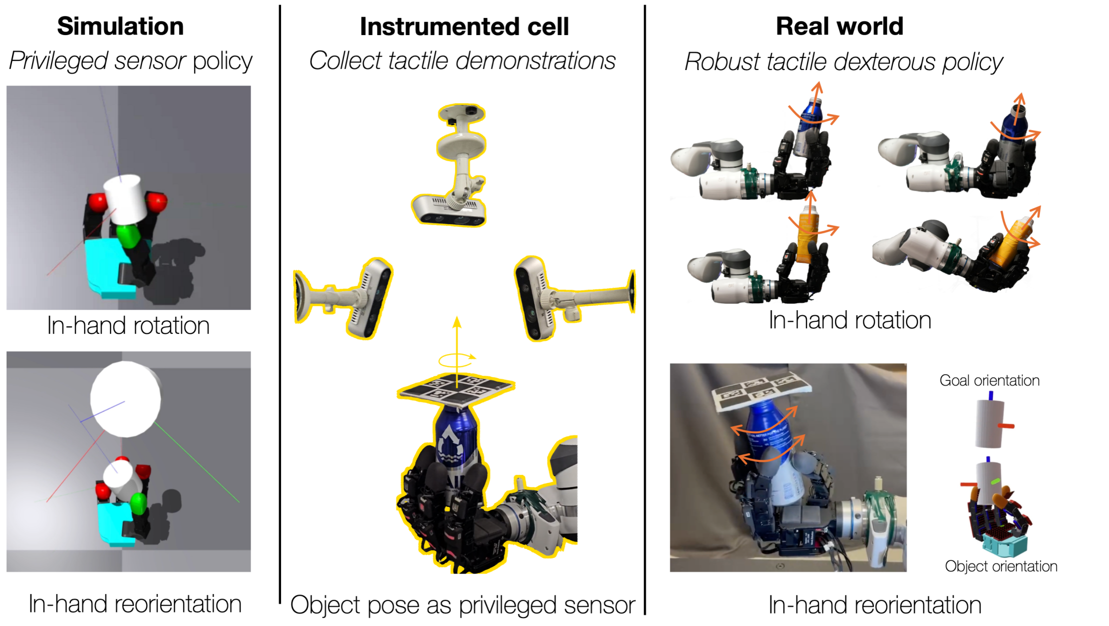


Figure 5.1. PTLD: sim-to-real Privileged Tactile Latent Distillation is an approach to learn tactile dexterous policies without simulating tactile sensors. First, *Privileged sensor* policies are trained in simulation using Reinforcement learning which produces strong policies. These policies are deployed in instrumented setups to collect tactile demonstrations. Finally, a tactile state estimator is trained from tactile demonstrations to obtain robust real-world deployable tactile policies. With PTLD, we demonstrate that **in-hand rotation** can be robust to object property changes such as slip, mass, and wrist orientation changes, and that performance for the challenging task of **in-hand reorientation** can be significantly improved by over 57% with tactile sensing

5.1 Introduction

Contact-rich dexterous manipulation with multi-fingered robot hands has remained a grand goal in robotics for several decades. The potential to solve tasks with human-like dexterity and use tools designed for humans paves a path towards physical intelligence in areas like healthcare and household tasks. Recent work in learning from demonstrations [32, 33, 218] provides a scalable recipe for learning new policies by collecting a large set of on demonstration data obtained via robot teleoperation, using hand-held grippers or kinesthetic teaching. However, such an approach is impractical for multi-fingered dexterous hands due to a difficulty in teleoperating robot hardware reliably for dexterous tasks like using a screwdriver, a wrench, or turn a door knob. Kinesthetic teaching is equally challenging when more than two fingers are required for a task such as reorienting an object in hand [23]. While hand-held grippers are promising [33, 192], they require designing an exoskeleton structure that balances the flexibility (DoFs) and stability required for dexterous tasks, and recent successes have largely been limited to simple tasks.

Sim-to-Real reinforcement learning (RL) offers an alternative to learn dexterous tasks and has tremendous success in learning robot locomotion [137, 167]. However, most existing approaches focus on blind proprioceptive only policies in both locomotion [79, 108, 211] and manipulation [139, 140]. While a few works have shown success with perceptive policies [1, 74, 159], training visual perceptive policies in simulation is relatively slow due to the added step of image rendering for simulating the image modality. Furthermore, these policies are often challenged by a large sim-to-real gap.

Our focus in this work is on *tactile dexterous manipulation*, encompassing dynamic tasks such as in-hand rotation, in-hand reorientation and pinch-to-power grasp transitions [2, 83, 140, 198, 202]. The standard approach for these tasks today is to learn policies through RL. However, akin to challenges in training visually perceptive policies there exist several challenges for training tactile perceptive/sensorimotor policies in simulation. First, simulating tactile sensors accurately is difficult. Therefore, most existing works [198, 201] resort to simplified tactile sensing models such as single point of contact or binary contact models. Second, even when one uses approximate soft-body simulation for training the policies, there exists a large sim-to-real gap which prevents straightforward deployment.

In this paper, we present a new approach to learn tactile manipulation policies without paying the cost of simulating tactile sensors. Our method takes inspiration from *privileged latent distillation* – where the idea is to first train an oracle policy with access to privileged state information (available in simulation) and then imitating the oracle policy (in simulation) to a zero-shot deployable policy using a perceptive state estimator that only has access to partially observations such as vision or proprioception. This idea has been presented in the literature with several names such as learning by cheating [24], data driven planning via imitation [34], or rapid motor adaptation [98] and has achieved tremendous empirical success in robotics.

Here, we extend privileged latent distillation in two distinct ways: First, we extend policy distillation in simulation to latent distillation in real world. Now to achieve this latent distillation, one requires an executable *oracle* policy in the real world. To this end, second, we treat properties such as object poses, object shape and the like as *privileged sensors*, and deploy the *privileged sensor* policies in the real world by instrumenting a real-world robot cell. Once we are able to execute the *privileged sensor* policy in the real world, we distill its latent into a tactile policy through supervised learning, by collecting a paired dataset of tactile sensor observations and latents produced by the *privileged sensor* policy.

Privileged latent distillation [24, 34, 98] is typically implemented with a teacher-student setup in simulation, requiring two stages of training. Stage 1, for oracle policy training, and Stage 2, for distillation of the oracle policy into a proprioceptive policy. Specifically, our method relaxes stage 2 distillation to leverage a few privileged quantities, requiring another round of real-world distillation. This can be laborious. Therefore, in this paper, we also present architectural advancements, demonstrating that the two stage approach [98] can very well be replaced with an asymmetric actor critic [138] training step requiring only a single round of training.

Finally, one might reasonably say that instrumenting a real-world robot cell for sim-to-real tactile distillation is too much hassle only to learn deployable sensorimotor tactile policies for the same task. Therefore, in addition to sim-to-real tactile distillation for a deployable proprioception policy, in this paper we also show successful results from a significantly harder task of tactile in-hand reorientation which cannot be accomplished in simulation purely with proprioception history only.

In summary, our contributions in this paper are threefold:

- We present a novel approach to learn sensorimotor tactile dexterous manipulation policies without paying the cost of simulating tactile sensors. We use privileged sensors as the interface between simulation and reality to perform *privileged latent distillation* using real world data.
- We present architectural advances for training manipulation policies, simplifying the two-stage distillation step in simulation into a single training step.
- Through our experiments we demonstrate that tactile policies trained through our sim-to-real latent distillation approach consistently outperform proprioception policies as well as adaptation based tactile policies in both robustness and performance.

5.2 Related work

5.2.1 Dexterous In-hand Manipulation

Dexterous in-hand manipulation has been an active area of research for decades [1, 2, 72, 126, 130, 147]. It features the cooperative use of multiple fingers on a multi-fingered hand to grasp and manipulate objects. While classical approaches need a physical model of the object and robot geometry to plan robot finger motions [50, 125], recent approaches have had success with using RL directly to learn policies in a model free manner [1, 2, 25, 140, 201]. However, RL approaches face the *sim-to-real* gap i.e., it is challenging to reproduce real world sensor observation and physics in simulation. Even for modalities such as vision where it is feasible to simulate the sensor, the simulation model is physically inaccurate and does not describe the real world sensors, therefore extensive visual domain randomization [74, 172] is crucial. Our method, on the other hand trains in simulation with observations such as object poses and object shape that do not suffer a large sim-to-real gap, but requires one to forgo the zero-shot sim-to-real deployment assumption.

5.2.2 Privileged distillation

Partial observability is a significant challenge in RL. For dexterous in hand manipulation of objects estimating precise contact dynamics is quite important as it determines the object motion and subsequently the requisite action. In the absence of sensors to accurately estimate these properties, most existing approaches in the literature have prominently used privileged distillation [24, 34, 98, 140, 160, 222] to learn dexterous manipulation policies. As described in 5.3, first one trains an oracle policy that has access to privileged information only available in simulation, then one distills it into a deployable policy that produces probabilistic estimates of the privileged state using a small history of sensor observations. Concretely, this is usually a history of proprioceptive observations or visual observations.

5.2.3 Tactile sensing and Representation learning

The tactile modality has long been promised to be imperative for such contact-rich dexterous manipulation. The last decade has seen a plethora of tactile sensors introduced for manipulation ranging from vision-based tactile sensors such as the GelSight [207] and DIGIT [100], magnetic-skin tactile sensors such as ReSkin [14] and Xela [175], to resistive and capacitive sensing sensors [86]. Consequently, there also exists a rich body of work that leverages tactile sensors for perception tasks and simple manipulation tasks such as peg-insertion [42, 152], cable manipulation [153] and planar pushing [162]. However, these tactile manipulation tasks in the literature are chosen carefully to satisfy the following features: a) the task is often quasi-static and b) the tasks are simple to demonstrate to make them amenable to behavior cloning methods [32, 218]. Recently, self supervised tactile representations [82, 83, 152, 215] address the lack of standardization in tactile sensors in robotics, demonstrating their use in several manipulation tasks, although the tasks chosen are largely quasi-static. Of these, [83] notably proposed a tactile adaptation algorithm to adapt RL trained dexterous manipulation policies in the real world to use tactile sensing, along the lines of policy finetuning [180] using real world data. However, these methods are fundamentally limited section 5.6.3 and only gain from rejection sampling of successful real-world trajectories, as the performance ceiling of the proprioceptive teacher policies are limited. In contrast, `PTLD` produces quantitatively more robust policy behaviours.

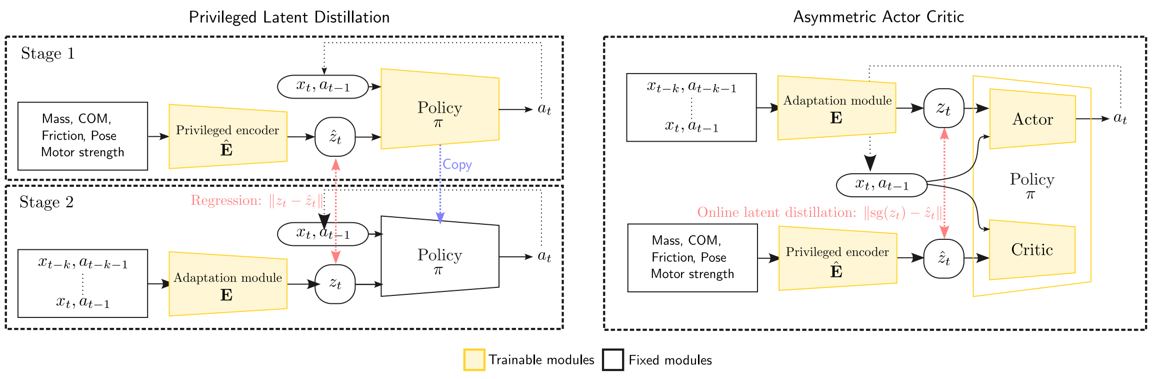

Privileged Latent Distillation
Stage 1
Mass, COM, Friction, Pose Motor strength
Privileged encoder $\hat{\mathbf{E}}$
$\hat{z}_t$
x_t, a_{t-1}
Policy π
a_t
Copy
Stage 2
Regression: $\|z_t - \hat{z}_t\|$
x_{t-k}, a_{t-k-1}
x_t, a_{t-1}
Adaptation module $\mathbf{E}$
z_t
Policy π
a_t
Asymmetric Actor Critic
Adaptation module $\mathbf{E}$
Actor
Policy π
Critic
Online latent distillation: $\|\text{sg}(z_t) - \hat{z}_t\|$
Trainable modules
Fixed modules

Figure 5.2. **(left)** *Privileged latent distillation* is a two stage approach to training policies in simulation. An *oracle* policy with privileged information is trained in stage 1, then it is distilled into a deployable policy in stage 2 (in simulation). **(right)** Asymmetric Actor Critic is a single stage approach where two networks actor and critic respectively are trained simultaneously. The critic is provided with privileged information and learns the value function, while the actor is only given deployable partial sensor information

5.3 Background

5.3.1 Notation

We model the dexterous manipulation tasks we discuss in the paper as finite horizon ($N \in \mathbb{N}$) Partially Observable Markov Decision Processes (POMDPs) $\mathbf{M} \triangleq (\mathcal{S}, \mathcal{A}, \mathcal{X}, \mathcal{P}, \mathcal{R})$ where $(\mathcal{S}, \mathcal{A}, \mathcal{X}) \in \{\mathcal{S}_t, \mathcal{A}_t, \mathcal{X}_t\}_{t=1}^{N}$ denote the state, action and observation spaces over the finite horizon N respectively. $\mathcal{P} = \{\mathcal{P}_t : \mathcal{S}_{t-1} \times \mathcal{A}_{t-1} \rightarrow \mathcal{S}_t\}$ denotes the transition dynamics, and $\mathcal{R} = \{\mathcal{R}_t : \mathcal{S}_t \times \mathcal{A}_t \rightarrow [0, 1]\}$ denotes the reward function. Our goal is to learn policies $\pi : \mathcal{X}_{t-k+1:t} \times \mathcal{A}_{t-k:t-1} \rightarrow \mathcal{A}_t$ via Reinforcement Learning (RL) to maximize the expected return $G(\tau) = \sum_{t=0}^{N} \gamma^t R_t$ over the horizon as $\pi^* = \text{argmax}_{\tau \sim \mathcal{P}^\pi} \mathbb{E}\left[G(\tau)\right]$.

Specifically, we choose to parameterize the policy with a neural network that is a combination of an encoder $\mathbf{E}$ which encodes the observations $\mathcal{X}$ into a latent space $\mathcal{L}$, which is then consumed by the policy π to produce actions $\mathcal{A}$. Typically, we employ two encoders during training. First, we have $\hat{\mathbf{E}}$ the privileged encoder which has access to privileged observations in simulation, and $\mathbf{E}$ the adaptation encoder which only has access to deployable observations. Then we have the policy as follows:

$$\mathcal{A}_t \sim \pi(\mathbf{E}(\mathcal{X}_{t-k+1:t}), \mathcal{A}_{t-k:t-1}) \tag{5.1}$$

5.3.2 Privileged latent distillation

Privileged latent distillation is a two stage approach to learning deployable policies in the real-world (see fig. 5.2). This requires that the simulation environment for the task supports a) privileged state observations $\mathcal{X}^{\text{priv}}$ which are typically low dimensional quantities such as object, robot and contact states and b) sensor observations $\mathcal{X}^{\text{sensor}}$ which can be realized in the real world such as proprioception and rendered camera. First, an oracle policy is trained that is allowed to 'cheat' and observe the full privileged state $\mathcal{X}^{\text{priv}}$ that describes the environment wholly. Then, a deployable student policy is trained to imitate the oracle policy given only the sensor observations $\mathcal{X}^{\text{sensor}}$. Since the sensor observations only observe the state partially, most approaches employ *frame stacking* where a history of sensor observations, and previous actions are used as the observation.

The oracle policy is trained in simulation using an RL algorithm such as PPO [149], while distillation is usually implemented as supervised learning. Implementations typically use a) action imitation where the actions between the oracle and student policies are matched or b) latent imitation where an encoded latent between the oracle and student policies is supervised. Specifically, we have

$$\mathcal{L}_{\text{action}} = \left\| \hat{\pi}(\hat{\mathbf{E}}(\mathcal{X}^{\text{priv}}), \mathcal{A}) - \pi(\mathbf{E}(\mathcal{X}^{\text{sensor}}), \mathcal{A}) \right\| \tag{5.2}$$

$$\mathcal{L}_{\text{latent}} = \left\| \hat{\mathbf{E}}(\mathcal{X}^{\text{priv}}) - \mathbf{E}(\mathcal{X}^{\text{sensor}}) \right\| . \tag{5.3}$$

It must be noted that experience or observations collected for distillation is collected by the deployable (student) policy, while the supervision signal comes from the oracle encoder. This is importantly distinct from traditional offline imitation learning as student distillation implements an on policy variant of DAgger [146]. Specifically, since the student policy is supervised by the teacher on experience collected by the student, the student observes a wider observation space during training, resulting in a robust policy.

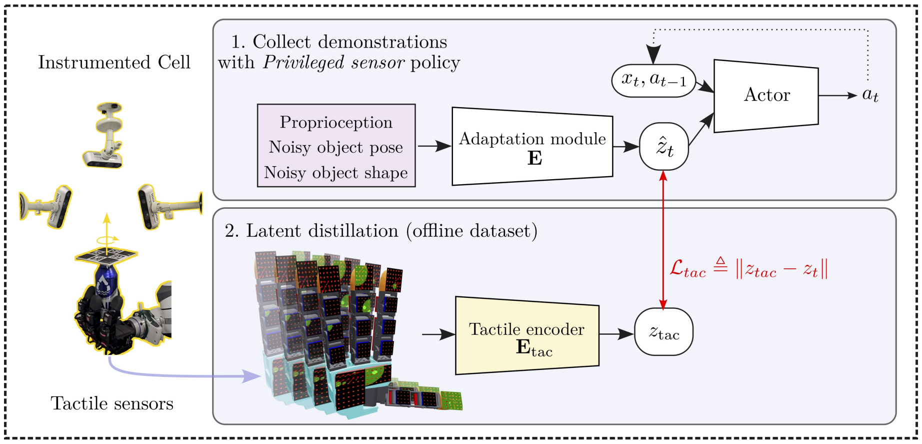

Instrumented Cell
1. Collect demonstrations with *Privileged sensor* policy
x_t, a_{t-1}
Actor
a_t
Proprioception
Noisy object pose
Noisy object shape
Adaptation module **E**
$\hat{z}_t$
2. Latent distillation (offline dataset)
$\mathcal{L}_{tac} \triangleq \|z_{tac} - z_t\|$
Tactile encoder $\mathbf{E}_{\text{tac}}$
z_{tac}
Tactile sensors

Figure 5.3. A simplified illustration of PTLD. Once we have a *privileged sensor* policy trained in simulation using AAC, first we collect demonstrations in the real world by deploying the policy, and additionally collect deployment sensor observations. Then, we train a deployment encoder (tactile encoder in this case) to recover the latents from the *privileged sensor* policy using an offline dataset.

5.3.3 Asymmetric Actor Critic

Asymmetric Actor Critic (AAC) [138] is another approach which aims to learn robust deployable policies by taking advantage of full-state observability in simulation. Specifically, it employs an actor-critic framework, where the critic is provided with the privileged state $\mathcal{X}^{\text{priv}}$, while the actor is provided with $\mathcal{X}^{\text{sensor}}$ sensor observations (see fig. 5.2). In our approach, we employ learning policies with AAC, as opposed to RMA [98] as it simplifies policy learning in simulation into a single training step.

5.4 PTLD: Privileged Tactile Latent Distillation

We now describe our method to train tactile manipulation policies using sim-to-real tactile distillation along with architectural improvements that simplify training in simulation.

5.4.1 Online distillation with Asymmetric Actor Critic

We employ an actor-critic framework for training manipulation policies in our work as opposed to privileged latent distillation. As alluded to before, this simplifies training into a single stage in simulation. Crucially, we find that our policy parameterization which separates the actor into an observation encoder $\mathbf{E}$ and policy π is beneficial. A separate encoder allows one to learn general state representations. Therefore inspired by self-distillation in representation learning [6, 67, 83, 152] approaches, we also employ a self-distillation representation loss, between the latent representations learnt by the critic (privileged) encoder and the actor (student) encoder:

$$\mathcal{L}_{\text{latent}} \triangleq \left\| \mathbf{E}(\mathcal{X}^{\text{sensor}}) - \text{sg}(\hat{\mathbf{E}}(\mathcal{X}^{\text{priv}})) \right\| \tag{5.4}$$

where sg denotes stop gradient. In our experiments (see section 5.6.2), we find that this online latent distillation loss improves both reward achieved by the policy in simulation. Futhermore, in simulation evaluation, this simple distillation loss results in similar policy performance to privileged latent distillation, motivating the simplification from two stage training in simulation to single stage training in simulation.

As shown in fig. 5.2, these networks are trained in simulation simultaneously in simulation, and we use the clip variant of proximal policy optimization (PPO) [149] augmented with the online latent distillation loss:

$$\mathcal{L}_{\text{PPO}} \triangleq \mathcal{L}_{\pi}^{\text{CLIP}}(\mathbf{E}, \pi) + c_V \mathcal{L}_V(\hat{\mathbf{E}}, V) + \mathcal{L}_{\text{entropy}}(\mathbf{E}, \pi) \tag{5.5}$$

$$\mathcal{L} \triangleq \mathcal{L}_{\text{PPO}} + c_{\text{latent}} \mathcal{L}_{\text{latent}} \tag{5.6}$$

where c_{latent} is a weighting factor. We optimize the total loss $\mathcal{L}$ via backpropagation.

5.4.2 Privileged sensors for Sim-to-Real Tactile Distillation

The essence of our method relies on the observation that directly deploying policies that take as input partial observations $\mathcal{X}^{\text{sensor}}$ is suboptimal. However, most existing works use simplified sensor observations for two primary reasons. First, they rely on simple sensor observations such as proprioception, scan lines, or binary contact for tactile sensor to simplify simulation training. Second, many methods aim for zero-shot sim-to-real deployment, which necessitates using these simplified input sensor models.

In this work, first we note that when we allow the actor to access *privileged sensors* such as object pose and object shape that provide higher observability into the state, the trained policy can qualitatively learn different behaviors in addition to improved performance in simulation. Now that the upper performance bound of the *privileged sensor* policy is higher, we wish to distill this policy into a final policy. However, such a policy cannot be directly deployed in the real world. Therefore, we instrument a real world cell, for instance with multiple cameras and object markers to provide (noisy) object poses $\mathbf{T}_t^W \in \mathbf{SE}(3)$ as the real world *privileged sensor*, and additionally also sensorize the multi-fingered robot hand with tactile sensors. Then we deploy the *privileged sensor* policy in the cell and collect an offline dataset of *on policy demonstrations* and record both the latent representations produced by the policy as well as the associated tactile sensor observations (fig. 5.3). Finally, we train an observation encoder that takes as input both tactile sensor data and proprioception data to match the latents from the privileged sensor policy. Formally, as illustrated in fig. 5.3 we distill the *privileged sensor* policy into a tactile policy and use the MSE loss for supervision.

Since we perform distillation using real world data, and do not have access to a simulator to effectively train the student encoder using its own experience, we employ DAgger [146], where we iteratively train the student tactile encoder with an aggregated dataset where experience is collected by the policy using intermediate trained tactile encoders.

5.4.3 Self supervised learning with PTLD

In addition to the online latent distillation loss, we use self supervised objectives to regularize the latent representation to capture the requisite information and improve sample efficiency during RL training. Specifically, we implement a reconstruction decoder $\hat{D}$ which takes in the privileged encoder latent and reconstructs the privileged sensor information, which in our instantiation is relative object orientation $\mathbf{R}_t^{t-N} \in \mathbf{SO}(3)$ denoting the orientation of the object from the current timestep t to the first timestep of the horizon $t-N$. We use the MSE loss between the 6D representation [221] of object orientation and the predicted object orientation as $\mathcal{L}_{\text{pose}} \triangleq \|\hat{D}(\hat{z}_t) - \mathbf{R}_t^{t-N}\|$.

Then our final RL training loss is:

$$\mathcal{L} \triangleq \mathcal{L}_{\text{PPO}} + c_{\text{latent}}\mathcal{L}_{\text{latent}} + c_{\text{pose}}\mathcal{L}_{\text{pose}} \tag{5.7}$$

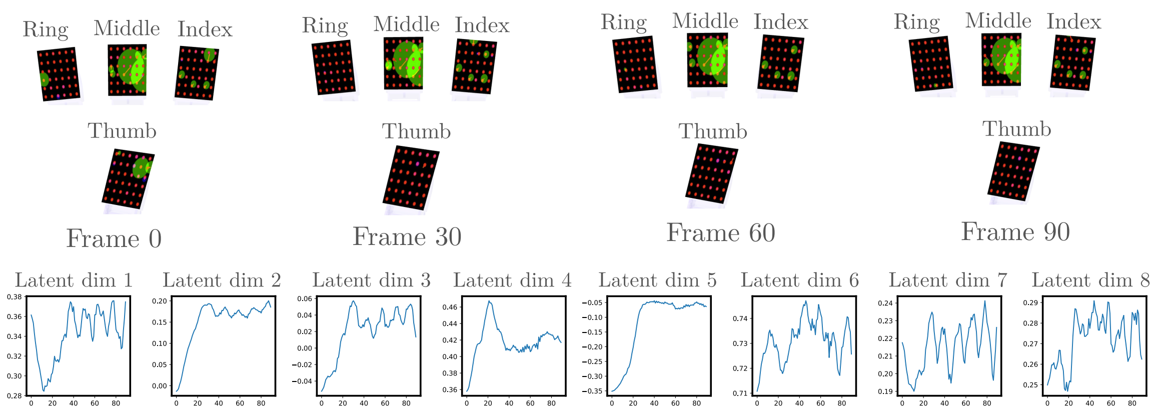


Figure 5.4. Visualization of tactile observations and the latents changing over the first 1 second of *privileged sensor* policy deployment. Here we visualize only the tactile data at the robot fingertip for simplicity, however the tactile encoder takes as input all observations from the hand.

5.5 The privileged tactile manipulation system

5.5.1 Real world privileged sensor cell

Robot Setup For all experiments, we use the Allegro hand sensorized with Xela uSkin [175], attached to a Franka Panda robot arm. There are a total of 18 Xela uSkin sensing pads on the Allegro hand amounting to a total of 368 individual sensors. We use the continuous 3-axis raw tactile sensor measurements from the Xela sensor, over the processed force measurements as we find that those measurements contain significant hysteresis and lag. A baseline signal with no contact is additionally collected (over 2 minutes) and subtracted from the raw tactile measurements before data collection each time.

Privileged Sensors Setup To deploy the privileged sensor policies for latent dataset collection, we instrument the real world robot cell with 4 Realsense D435i/D435 RGBD cameras which view the in-hand manipulation area. These cameras are calibrated jointly and track an Aruco marker attached to the object being manipulated to produce reliable multi-view pose estimate that is refined via Pose Graph Optimization (PGO) [40]. We assume known shapes of the objects that are being deployed.

5.5.2 Manipulation task I: In hand rotation

We choose the task of in-hand rotation [140, 141, 198] about the z-axis for our experiments. In this task the robot hand is required to rotate an object along a specified axis, ensuring that the object does not drop and remains held by the robot fingers. We demonstrate that task performance on this task can be vastly improved by incorporating rich tactile information as part of the policy observation. For state observation and reward details for this task, we refer the reader to [140].

Tactile encoder For the tactile encoder (fig. 5.5 (a)), we concatenate the tactile observations $\mathcal{X}^{\text{tactile}} \in \mathbb{R}^{368\times3}$, and the tactile sensor positions computed from the Allegro joint states using forward kinematics $\mathcal{X}^{\text{sensor\_pos}} \in \mathbb{R}^{368\times3}$ as the input $\mathcal{X} = \text{cat}(\mathcal{X}^{\text{tactile}}, \mathcal{X}^{\text{sensor\_pos}}) \in \mathbb{R}^{368\times6}$. The tactile observations are produced at 100Hz, and we use a history of 0.5s of tactile data as input to the encoder. Specifically, our tactile encoder uses a combination of MLP and 1D temporal convolution as follows: ($\mathcal{X} \rightarrow$ MLP $\rightarrow$ Temporal conv $\rightarrow$ MLP $\rightarrow z$), where z is the predicted latent.

5.5.3 Manipulation task II: In hand reorientation

We also develop an in-hand general purpose re-orientation task designed as a goal orientation reaching problem. We wish to demonstrate that PTLD can not only be used to improve existing policies that can be deployed in the real world with proprioception, but that we can also learn more difficult policies which require additional information to be encoded. Specifically, in this task, we randomly sample goal orientations $\mathbf{R}_t^{\text{goal}} \in \mathbf{SO}(3)$ of an object, and the robot is tasked to manipulate the object held by the fingers, such that it's pose $\mathbf{R}_t^{\text{object}}$ reaches within a set threshold angular distance ($\delta \triangleq 0.25$ rad $\simeq 14.5°$) of the goal orientation (see fig. 5.1). However, due to limitations in our hardware in the real robot cell (i.e., the four cameras are placed above the manipulation area), we sample goal orientations in the upper hemisphere within $\theta = 40°$ about the z-axis. We use the asymmetric actor critic framework trained with PPO to solve this task as well.

State The state observation input to the adaptation module $\mathbf{E}$ in simulation includes the Allegro hand joint states $q_t \in \mathbb{R}^{16}$, previous joint targets $\tilde{q}_t \in \mathbb{R}^{16}$, noisy

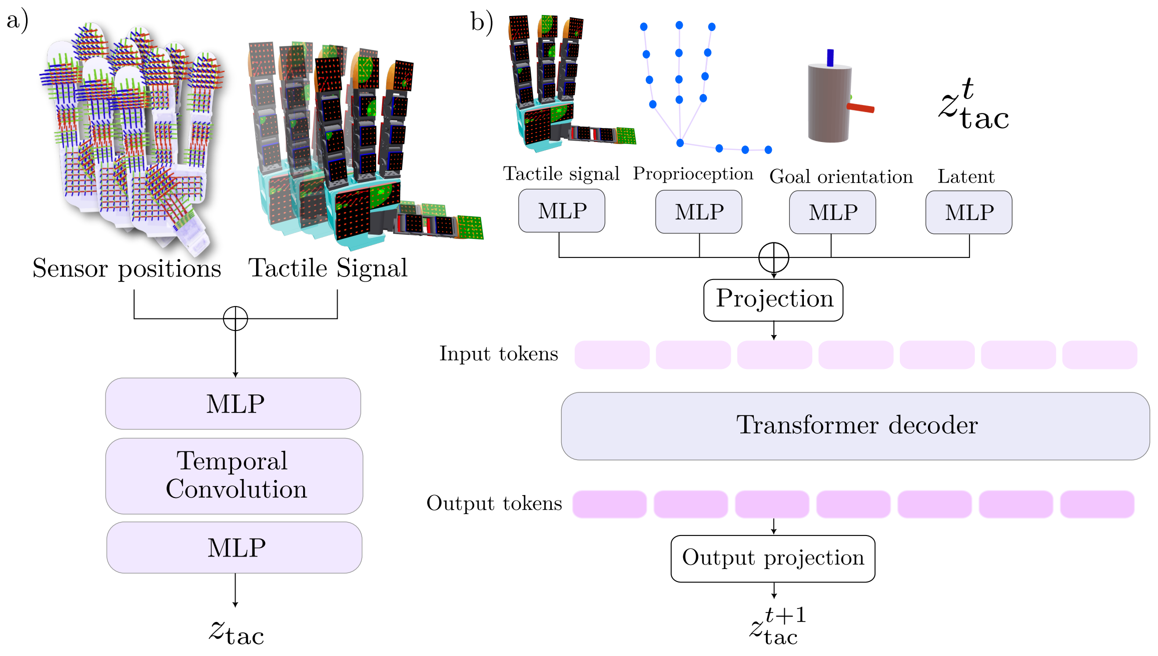


Figure 5.5. Tactile encoders for manipulation tasks: a) For in-hand rotation, we concatenate a history of tactile signals and sensor positions, and encode them using a 1D temporal convolution network to predict the tactile latents. b) For in-hand reorientation, we concatenate, tactile signals, proprioception, goal orientations and past latents and embed them with a causal transformer to produce future tactile latents.

object position $p_t \in \mathbb{R}^3$, noisy object orientation $\mathbf{R}_t \in \mathbf{SO}(3)$, and goal orientation $\mathbf{R}_t^{\text{goal}} \in \mathbf{SO}(3)$. All orientations use the the 6-D [221] rotation representation. We also concatenate the relative difference between the goal and current object pose to the goal pose, to facilitate learning. Since a single observation maybe insufficient, we employ *frame stacking* and provide the actor with a 30-step history ($\sim 1.5s$ for a control rate of 20Hz) of these observations. On the other hand, the privileged encoder $\hat{\mathbf{E}}$ is also provided with object linear and angular velocity, fingertip states (position, orientation and velocities.) in addition to the inputs of $\mathbf{E}$.

Reward We train the policy with a mixture of rewards to ensure that the robot not only succeeds in reaching goal poses, but also employs natural realizable finger gaits. The reward (table 5.1) contains three main terms ($r_{\text{goal}}, r_{\text{success}}, r_{\text{streak}}$), which includes the rotational distance reward, success bonus and a streak bonus to encourage the agent to reach multiple goals one after another. We also encourage the policy to maintain contact with the fingertips and keep the object in the center using

($r_{\text{contact}}, r_{\text{position}}$). In addition to the positive rewards, we also use motion penalties and energy penalties to obtain smooth motions. Crucially, we find that penalizing $r_{\text{finger\_pose}}$ the robot finger motion from the initial finger pose is useful in generating a finger gait. Without this penalty, the policy learns to curl the fingers along with the object, resulting in unrecoverable hand configurations. We detail other motion and energy penalties for reward shaping in the Appendix.

Reset Since we expect the policy to reach several goals in an episode, we design the reset strategy accordingly. We start the episode with a stable grasp sampled from a grasp set as commonly implemented, and reset the goal orientations multiple times within an episode when the object orientation reaches the goal orientation ($\leq \delta$). Additionally, we use a z-height threshold to reset the episode when the policy drops the object from its fingers. Finally, we also use the standard episode reset after a fixed number of simulation steps.

Reward	**Scale**
$r_{\text{goal}} \triangleq \frac{1}{d(\mathbf{R}_t^{\text{object}}, \mathbf{R}_t^{\text{goal}}) + \epsilon}$	2.0
$r_{\text{success}} \triangleq (1 \text{ if } d(\mathbf{R}_t^{\text{object}}, \mathbf{R}_t^{\text{goal}}) \leq \delta \text{ else } 0)$	5.0
$r_{\text{streak}} \triangleq \frac{N_{\text{success}}}{N_{\text{max\_success}}}$	2.0
$r_{\text{contact}} \triangleq \sum_i (C_i > \delta_{\text{contact}})$	0.1
$r_{\text{position}} \triangleq \lVert p_t - p_0 \rVert$	0.05
$r_{\text{finger\_pose}} \triangleq \lVert q_t - q_0 \rVert$	-1.0
$r_{\text{fingertip\_object}} \triangleq \sum_i \lVert p_{\text{fingertip}_i} - p_t \rVert$	-0.2

Table 5.1. Reward function for any target in-hand reorientation

Tactile encoder Since the in-hand reorientation task is significantly more complex, and requires the policy to change gait according to the relative difference between current object pose $\mathbf{R}_t^{\text{object}}$ and the goal pose $\mathbf{R}_t^{\text{goal}}$, for this task we use a *recursive state estimator* as the tactile encoder. This estimator reasons about the full sequence of tactile observations, proprioception, and goal orientations of the object. To this end, we use a Transformer network (see fig. 5.5) to auto-regressively predict the current latent given the history of observations and latents. Specifically, we embed each input, tactile observations $\mathcal{X}_t^{\text{tactile}} \in \mathbb{R}^{368\times 3}$, proprioception input $\mathcal{X}_t^{\text{proprio}} \in \mathbb{R}^{32}$, the goal orientations $\mathbf{R}_t^{\text{goal}} \in \mathbf{SO}(3)$ and the previous latent $z_t \in \mathbb{R}^8$, individually using 2-layer MLP networks. We subsequently concatenate all the embeddings, linearly project it and add positional encodings to produce the input embedding to the transformer. Finally, the embeddings are processed by a decoder (causal) transformer, from which we select the next predicted latent z_{t+1} during inference.

5.6 Experiments

In this section, we first demonstrate in section 5.6.2 that the in-hand rotation task can be improved in simulation using *privileged sensors* and additionally that the in-hand rotation task typically trained using a two stage RMA [98, 140] approach can very well be replaced by AAC by using our online latent distillation loss. Then, in section 5.6.3, we present results of in-hand-rotation now improved by `PTLD` to incorporate tactile information, compared against other baseline methods. We then investigate decoding object orientations from tactile latents in section 5.6.4, and finally, we showcase the in-hand reorientation task in section 5.6.5.

5.6.1 Implementation details

We train the policies using PPO [149], and use IsaacGym [119] as our simulator. Once an offline dataset is collected in the real world cell, we train the tactile encoder via supervised learning. For optimizing the tactile encoder, we use AdamW optimizer, with a learning rate of 1e$-$4. We use ROS2 for communication between the different robot processes, and achieve a real-time policy deployment rate of $\sim$ 20Hz.

5.6.2 Privileged sensors improve performance in simulation

In fig. 5.6, we show that reward achieved by the policy in the distillation step is significantly higher when compared to the policy that only has access to proprioception to recover the latents from the stage 1 oracle policy. Further as shown in table 5.2, here we also evaluate the policy performance in simulation where we randomize object properties, and evaluate policy quality metrics introduced in [140] such as (1) *rotation*

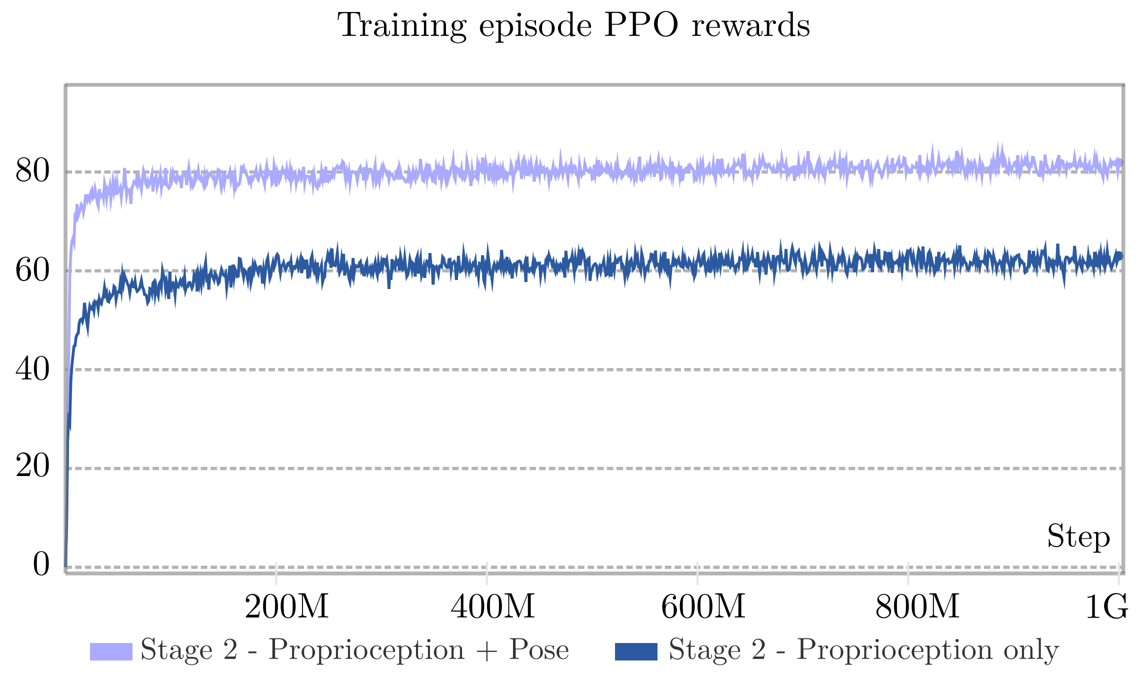


Figure 5.6. Policy performance for stage 2 distillation step in simulation improves significantly when object pose information is provided in addition to proprioception input

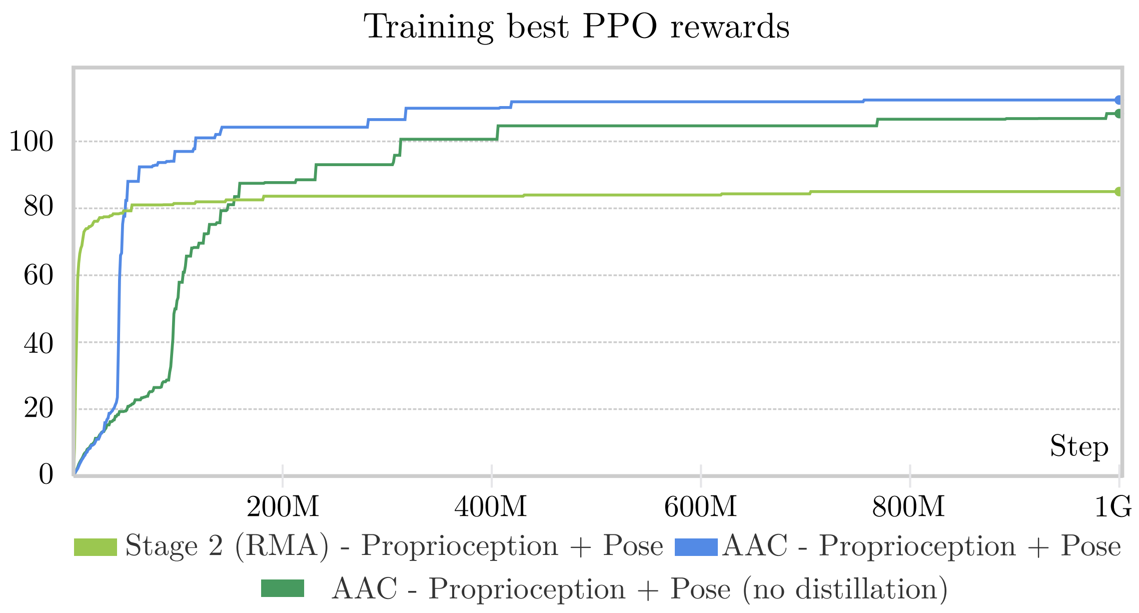


Figure 5.7. Asymmetric actor critic (blue) trained in a single stage in simulation outperforms the RMA distillation approach which requires two stage training in simulation

			z-axis	
Method	Input modalities	RotR ↑	TTF ↑	RotP ↓
Oracle		**159.4**	**0.89**	**31.86**
Latent distillation (RMA)	Proprioception	139.0	0.79	28.53
Latent distillation (RMA)	+ Pose	153.1	0.86	31.09
AAC	Proprioception	129.89	0.76	27.83
AAC	+ Pose	151.24	0.83	26.51
AAC (no distillation)	+ Pose	134	0.78	29.74

Table 5.2. We compare the performance improvement over various baselines on z-axis in hand rotation, under the same training setting *in simulation*. Specifically, compared to [140], we first demonstrate that Asymmetric Actor critic with latent supervision (AAC) improves performance in simulation significantly. Further, we demonstrate that additional input modalities such as object shape and pose produce significant improvements in simulation, which is a pre-requisite for tactile distillation to outperform the baselines.

reward: which measures the reward achieved by the policy for rotating the object, (2) *time to fall*, which measures the amount of time as a fraction of the episode that the policy rotates the object before dropping it, and (3) *undesired rotation penalty*, which measures any off-axis rotation. Specifically, as expected, the stage 2 policy with object pose information improves in all metrics in simulation.

Similarly, in fig. 5.7, we observe that while the RMA stage 2 distillation step learns much quicker at the beginning of training, the AAC approach, and more specifically the AAC variant which uses the online latent distillation (see section 5.4.1) outperforms RMA eventually and leads to policies with better behaviors in the real world. Finally, in table 5.2 we also see that the AAC policy with pose as the privileged sensor given as input to the actor encoder, performs similarly to the RMA policy. This motivates our choice to simplify training in simulation to a single AAC training stage.

5.6.3 PTLD improves policy robustness in the real world

We compare PTLD against several baselines that include proprioception, as well as both proprioception and tactile information, to demonstrate the effectiveness of our method. First, for proprioception baselines, we compare against RMA [140] which implements the two stage distillation approach in simulation and AAC, which is our single stage variant where the encoder gets only a small history proprioception and action. We also implement our privileged distillation approach but purely with proprioception as a third baseline, to demonstrate that tactile signals do indeed help in recovering information from the *privileged sensor*, and that PTLD does not simply finetune the encoders with more data. Finally, we also implement a baseline taking tactile information as input. Here, we implement the *tactile adaptation* approach presented in [83], but modify it for the Xela sensors, and use the Sparsh-Skin [152] representation.

We compute metrics over 10 trials and run trials with three different cylinder shaped objects. Specifically, we compute *total rotation* which measures the rotation about the z-axis in radians, the *time to fall*, which measures the time in seconds the policy rotates the object before it is dropped, and *vertical drift* which measures slip along the z-axis, and indirectly the instability of rotating an object. Here, from fig. 5.8 we see that PTLD outperforms all baselines by an order of magnitude. Specifically, in total rotation and time to fall, PTLD is significantly better than both privileged distillation with proprioception and tactile adaptation. Here, we note that our approach overcomes the main limitation of *tactile adaptation*, because with PTLD the teacher data is sourced from a significantly better privileged sensor policy, which results in significantly improved metrics. Similarly, when compared against privileged distillation with proprioception only, we note that in the absence of additional deployment sensors such as tactile sensors, distillation in the real world manifests as policy finetuning, which can only result in modest policy improvements.

5.6.4 Tactile information improves object state estimation

So far, we have demonstrated that PTLD with tactile information improves policy robustness. In this section, we investigate the reason. Specifically, we evaluate the ability of the learned encoders to recover object orientation information. We train an

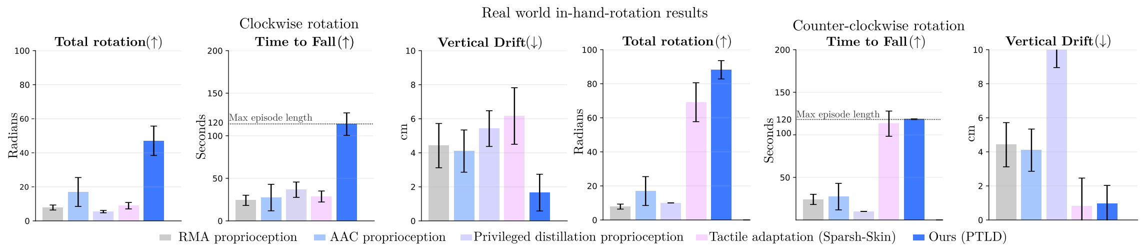


Figure 5.8. Real world comparison of in-hand rotation policy performance over 10 trials with three cylinder like objects. `PTLD` can

object pose decoder from the distilled real world encoder. We use the 6D rotation representation and use the MSE loss to supervise the object pose decoder using real world tactile policy rollouts as the dataset. We evaluate the performance of object orientation estimation on held out rollout trajectories. In table 5.3, we compare the object pose estimation error accumulated over 100 prediction between encoders observing only proprioception or proprioception and tactile. We consider different pose parameterization as input and output of the tactile encoder: a) Absolute object pose $\mathbf{R}_t^W$ denotes the pose of the object at time t with respect to the hand coordinate frame, b) Relative object pose $\mathbf{R}_t^{t-H}$ denotes the pose of the object with respect to the start of the temporal window. Here in both cases, we find that the addition of tactile information improves the rotation prediction errors. For object rotation, the decoder only needs to predict the angular rotation about the axis (z-axis). While we expect that proprioception and tactile only capture relative information, we hypothesize that with absolute object pose, the decoder is able to learn a fixed transform, given sufficient data, as the hand is kept fixed during policy deployment for this dataset. Further, the inferior performance of proprioception with relative object pose input and output, can be explained as multi-modal problem, since the input proprioception patterns for a finger gait can be similar, while the outputs can be substantially different due to slippage or sliding. A qualitative visualization of object pose estimation from the tactile encoder (with absolute object pose parameterization) is also visualized in fig. 5.9

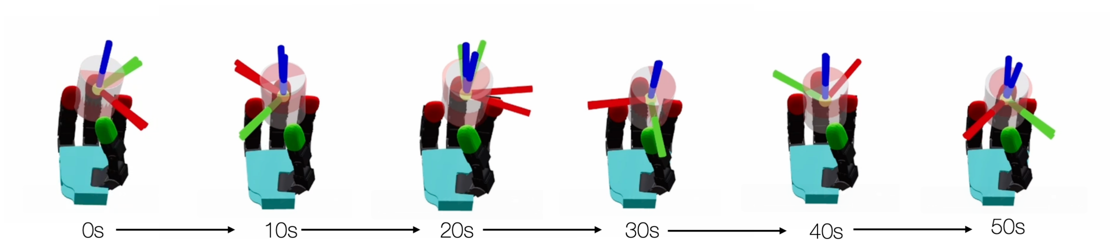


Figure 5.9. Visualization of object pose reconstruction from tactile latent decoder: The red transparent cylinder denotes the predicted object pose prediction, while the gray translucent cylinder denotes the true object pose recorded from the instrumented cell during deployment. Specifically, we compose the predictions over each second to visualize the cumulative tactile object pose reconstruction over time.

		Avg. Rotation error over 100 steps
Pose parameterization	Distillation modality	Rad (↓)
Absolute object pose	Proprioception	$0.28_{\pm 0.06}$
Absolute object pose	Proprioception + Tactile	$\mathbf{0.25}_{\pm \mathbf{0.09}}$
Relative object pose	Proprioception	$0.45_{\pm 0.04}$
Relative object pose	Proprioception + Tactile	$\mathbf{0.22}_{\pm \mathbf{0.04}}$

Table 5.3. Average cumulative rotation error over 20 seconds (100 inference steps) for a decoder trained to recover object orientation from latents learned after real-world distillation. We freeze the proprioception / tactile encoder and only train the decoder.

5.6.5 Tactile object In-hand reorientation

As described in section 5.5.3, we train an in-hand reorientation task with privileged information access to object pose and goal poses in simulation, which is then deployed in the real world in the instrumented cell. We present this task as a showcase task to demonstrate that PTLD can be used to train difficult in-hand reorientation policies which require more precise object state estimation to achieve success in this task. Furthermore, as alluded to before, we note that for this task, a simple temporal convolution based tactile encoder does not suffice. Specifically, we find that with such an encoder, the encoder latches onto a single finger gaiting mode resulting in rotation of the object in hand in one direction. Therefore, in table 5.4 we measure the performance of in-hand reorientation using an Autoregressive transformer based

	In-hand reorientation	
Method	$N_{\text{goals reached}}$ (↑)	TTF (s) (↑)
Autoregressive Transformer **(proprio)**	$2.1_{\pm 2.14}$	$10.99_{\pm 10.15}$
Autoregressive Transformer **(tactile+proprio)**	$\mathbf{3.3}_{\pm 1.55}$	$\mathbf{13.42}_{\pm 10.18}$

Table 5.4. In-hand reorientation performance significantly drops in the absence of tactile information. Metrics are computed over 10 trials, and each episode is run until the object is dropped out of the robot hand.

encoder. We use number of goals reached ($N_{\text{goals reached}}$) and Time to Fall (TTF (s)) as the metric. As expected, we find that when we remove the tactile information from the transformer encoder, the number of goals reached by the policy on average significantly drops. Qualitatively we find that tactile information helps the policy to be robust when there are object slippages during reorientation.

5.7 Conclusion

In this paper, we presented PTLD as a method to learn dexterous policies with tactile sensing, without having to simulate tactile sensors. Our key idea was to disregard the zero-shot sim-to-real requirement in deploying policies, and instead rollout policies in instrumented setups with *privileged sensors*. Then the key idea was to learn tactile policies from these policy rollouts by matching implicit state latents between the *privileged sensor* policy and tactile policies. We demonstrated PTLD on in-hand rotation and in-hand reorientation, demonstrating performance improvements in both tasks. We note that while we use tactile sensing as the deployment sensor, our method is general and can be used to learn perception policies with other modalities such as vision as well.

5.8 Limitations

While PTLD provides a straightforward and novel way to train dexterous policies with sensing modalities that are otherwise difficult or infeasible to simulate, we recognize certain limitations in our work that need to be considered.

- **Modality overlap**: Since `PTLD` relies on distillation from a privileged sensor policy deployed in the real world, one must consider the overlap in information between the deployment and privileged sensor. For instance, noisy object pose as a privileged sensor only recovers the kinematic information from the simulator, while the deployment sensor (tactile sensor) can ideally leverage additional dynamics information such as contact forces and contact direction. This implies that the choice of privileged sensor plays a role in minimizing the information lost in sim-to-real distillation
- **Noise in privileged sensors** We note that sensing observations inherently contain noise in the real world, and that the simulation policy must be trained accounting for such noise. Noise in object pose estimation limits the upper ceiling of the policy on real-world deployment. In the case of object pose estimation, one could use a motion capture setup to get more accurate observations.

Chapter 6

Conclusions

In this thesis, we developed techniques to efficiently leverage tactile sensing for robot perception and dexterous manipulation.

In particular, we first explored using self-supervised representation learning to unify several visuo-tactile sensors that have been introduced in the community. With `Sparsh`, we first evaluated self-supervision algorithms for visuo-tactile sensors and developed a recipe to learn general purpose tactile representations that work across three different visuo-tactile sensors. When compared against end-to-end training for downstream tasks, we demonstrated that self-supervised representations offer increased sample efficiency thereby making downstream labeled data collection cheaper.

Then, with `Sparsh-skin`, we demonstrated that learning self-supervised representations for magnetic skin-based tactile sensors is also equally useful, even though it might appear that the signals are low-dimensional. For `Sparsh-skin` we considered tactile skins that cover the full multi-fingered dexterous hand, and learned tactile representations that are also conditioned on the hand configurations. `Sparsh-skin` improved both performance and sample efficiency in downstream tasks requiring labeled data.

Third, with `Sparsh-X`, we present the first multisensory tactile representations that unifies four different tactile modalities including tactile image, tactile audio, IMU and pressure signals available from the DIGIT360 sensor. With Sparsh-X we demonstrated not only the sample efficiency, but also the utility of unifying the multiple tactile sensing modalities.

Finally, we also presented `PTLD` in this thesis, where we explored learning improved dexterous policies in the real world that leverage tactile sensing, by distilling from *privileged sensor* policies in the real world, and therefore avoiding the difficulty of simulating tactile sensors.

This thesis presents several promising approaches toward incorporating tactile sensing for robot dexterous manipulation more efficiently. The next step toward general robotic manipulation would be to incorporate tactile sensing more tightly for several downstream tasks. There are several open challenges as well as avenues that can benefit with using tactile sensing as well as self-supervised tactile representations.

First, learning dexterous manipulation from human demonstration is particularly exciting. Most work that tackles learning from human demonstrations does so from the viewpoint of re-targeting human demonstrations to the robot embodiment kinematically [71, 73, 120]. This is sufficient for simple robotics tasks such as pick and place, however with dexterous tasks such as in-hand manipulation of objects, kinematic retargeting can result in implausible object and robot motions [133]. Several works have tackled the challenges of kinematic retargeting for dexterous manipulation via rejection sampling based filtering using simulation [133] or by learning manipulation priors in simulation for real-world dexterous tele-operation [202]. Nevertheless, all of the approaches cite the lack of tactile sensing as a major drawback. Primarily, tactile sensing can provide additional observability into the human-object interaction system, to provide actual contact state with objects, specific contact patches with object surfaces, as well as forces applied on these objects, which can be retargeted to the robot embodiment to produce smooth realizable robot trajectories. These realizable trajectories can then be used to learn policies using *learning from demonstration* techniques. In fact recent advances in hardware such as OSMO [200] can be used to collect such human demonstrations with tactile sensing. An auxiliary question that presents itself with the advent of such hardware is, what changes to data collection for robot manipulation is needed?

A second such avenue is multi-modal world models for robotics. With `PTLD` we enabled learning tactile latent representations specific to certain dexterous tasks. Similarly with `Sparsh`, `Sparsh-skin`, and `Sparsh-X` we learned tactile latent representations using self-supervised learning to improve several downstream tasks. Can we learn general purpose environment encoders from multiple sensing modalities jointly?

Much recent progress in robotics has been enabled by vision-language models [89] and more recently video models [9] as the backbone for robotics models. These models provide strong visual priors enabling visual generalization with robot policies. Standardizing tactile sensing across robot hardware (and tactile sensing human gloves) and subsequently learning dynamics models using human demonstrations, holds promise in enabling general dexterous intelligence in robots. There are several plausible approaches toward incorporating tactile sensing via human demonstrations: a) Tactile sensors can be directly incorporated via reconstruction and next-step latent prediction objectives in existing vision-language or video models and b) Tactile sensing can be used to learn *value* functions for dexterous tasks, where we can learn the 'goodness' of certain hand-object states from tactile observations.

Bibliography

[1] Ilge Akkaya, Marcin Andrychowicz, Maciek Chociej, Mateusz Litwin, Bob McGrew, Arthur Petron, Alex Paino, Matthias Plappert, Glenn Powell, Raphael Ribas, et al. Solving rubik's cube with a robot hand. *arXiv preprint arXiv:1910.07113*, 2019. 5.1, 5.2.1

[2] OpenAI: Marcin Andrychowicz, Bowen Baker, Maciek Chociej, Rafal Jozefowicz, Bob McGrew, Jakub Pachocki, Arthur Petron, Matthias Plappert, Glenn Powell, Alex Ray, et al. Learning dexterous in-hand manipulation. *The International Journal of Robotics Research*, 39(1):3–20, 2020. 5.1, 5.2.1

[3] Kirsty Aquilina, David AW Barton, and Nathan F Lepora. Tactile control for object tracking and dynamic contour following. *Robotics and Autonomous Systems*, 178:104710, 2024. 4.2

[4] Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, Yann LeCun, and Nicolas Ballas. Self-supervised learning from images with a joint-embedding predictive architecture. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15619–15629, June 2023. (document), 2.2, 2.3, 2.10.1, 2.10.2

[5] Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, Yann LeCun, and Nicolas Ballas. Self-supervised learning from images with a joint-embedding predictive architecture. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15619–15629, 2023. URL `https://openaccess.thecvf.com/content/CVPR2023/papers/Assran_Self-Supervised_Learning_From_Images_With_a_Joint-Embedding_Predictive_Architecture_CVPR_2023_paper.pdf`. 3.3.2

[6] Alexei Baevski, Wei-Ning Hsu, Qiantong Xu, Arun Babu, Jiatao Gu, and Michael Auli. data2vec: A general framework for self-supervised learning in speech, vision and language. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 1298–1312. PMLR, 17–23 Jul 2022. URL `https://proceedings.mlr.press/v162/baevski22a.html`. 3.3.1, 5.4.1

[7] Alexei Baevski, Wei-Ning Hsu, Qiantong Xu, Arun Babu, Jiatao Gu, and Michael Auli. data2vec: A general framework for self-supervised learning in speech, vision and language. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 1298–1312. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/baevski22a.html. 2.10.1

[8] Randall Balestriero, Mark Ibrahim, Vlad Sobal, Ari Morcos, Shashank Shekhar, Tom Goldstein, Florian Bordes, Adrien Bardes, Gregoire Mialon, Yuandong Tian, Avi Schwarzschild, Andrew Gordon Wilson, Jonas Geiping, Quentin Garrido, Pierre Fernandez, Amir Bar, Hamed Pirsiavash, Yann LeCun, and Micah Goldblum. A cookbook of self-supervised learning, 2023. 2.2

[9] Philip J. Ball, Jakob Bauer, Frank Belletti, Bethanie Brownfield, Ariel Ephrat, Shlomi Fruchter, Agrim Gupta, Kristian Holsheimer, Aleksander Holynski, Jiri Hron, Christos Kaplanis, Marjorie Limont, Matt McGill, Yanko Oliveira, Jack Parker-Holder, Frank Perbet, Guy Scully, Jeremy Shar, Stephen Spencer, Omer Tov, Ruben Villegas, Emma Wang, Jessica Yung, Cip Baetu, Jordi Berbel, David Bridson, Jake Bruce, Gavin Buttimore, Sarah Chakera, Bilva Chandra, Paul Collins, Alex Cullum, Bogdan Damoc, Vibha Dasagi, Maxime Gazeau, Charles Gbadamosi, Woohyun Han, Ed Hirst, Ashyana Kachra, Lucie Kerley, Kristian Kjems, Eva Knoepfel, Vika Koriakin, Jessica Lo, Cong Lu, Zeb Mehring, Alex Moufarek, Henna Nandwani, Valeria Oliveira, Fabio Pardo, Jane Park, Andrew Pierson, Ben Poole, Helen Ran, Tim Salimans, Manuel Sanchez, Igor Saprykin, Amy Shen, Sailesh Sidhwani, Duncan Smith, Joe Stanton, Hamish Tomlinson, Dimple Vijaykumar, Luyu Wang, Piers Wingfield, Nat Wong, Keyang Xu, Christopher Yew, Nick Young, Vadim Zubov, Douglas Eck, Dumitru Erhan, Koray Kavukcuoglu, Demis Hassabis, Zoubin Gharamani, Raia Hadsell, Aäron van den Oord, Inbar Mosseri, Adrian Bolton, Satinder Singh, and Tim Rocktäschel. Genie 3: A new frontier for world models. 2025. 6

[10] Hangbo Bao, Li Dong, Songhao Piao, and Furu Wei. Beit: Bert pre-training of image transformers. *arXiv preprint arXiv:2106.08254*, 2021. 2.10.1

[11] Adrien Bardes, Quentin Garrido, Jean Ponce, Xinlei Chen, Michael Rabbat, Yann LeCun, Mido Assran, and Nicolas Ballas. V-JEPA: Latent video prediction for visual representation learning. 2023. (document), 2.2, 2.4, 2.10.1, 2.10.3, 4.1

[12] Adrien Bardes, Jean Ponce, and Yann LeCun. MC-JEPA: A joint-embedding predictive architecture for self-supervised learning of motion and content features, 2023. 2.10.1

[13] Maria Bauza, Antonia Bronars, and Alberto Rodriguez. Tac2Pose: Tactile object pose estimation from the first touch. *The International Journal of*

Robotics Research, 42(13):1185–1209, 2023. doi: 10.1177/02783649231196925. URL `https://doi.org/10.1177/02783649231196925`. 2.1, 2.4, 2.6.1

[14] Raunaq Bhirangi, Tess Hellebrekers, Carmel Majidi, and Abhinav Gupta. Reskin: versatile, replaceable, lasting tactile skins. In *5th Annual Conference on Robot Learning*, 2021. URL `https://proceedings.mlr.press/v164/bhirangi22a/bhirangi22a.pdf`. 3.1, 3.2.1, 3.3.2, 5.2.3

[15] Raunaq Bhirangi, Venkatesh Pattabiraman, Enes Erciyes, Yifeng Cao, Tess Hellebrekers, and Lerrel Pinto. Anyskin: Plug-and-play skin sensing for robotic touch. *arXiv preprint arXiv:2409.08276*, 2024. URL `https://arxiv.org/abs/2409.08276`. 3.1, 3.2.1, 3.4.2

[16] Raunaq Bhirangi, Chenyu Wang, Venkatesh Pattabiraman, Carmel Majidi, Abhinav Gupta, Tess Hellebrekers, and Lerrel Pinto. Hierarchical state space models for continuous sequence-to-sequence modeling. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2025. URL `https://dl.acm.org/doi/10.5555/3692070.3692223`. 3.4.2

[17] David Córdova Bulens, Nathan F. Lepora, Stephen J. Redmond, and Benjamin Ward-Cherrier. Incipient slip detection with a biomimetic skin morphology. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 8972–8978, 2023. doi: 10.1109/IROS55552.2023.10341807. 2.1, 2.4

[18] Remi Cadene, Simon Alibert, Alexander Soare, Quentin Gallouedec, Adil Zouitine, and Thomas Wolf. Lerobot: State-of-the-art machine learning for real-world robotics in pytorch. `https://github.com/huggingface/lerobot`, 2024. 4.6.4

[19] Roberto Calandra, Andrew Owens, Manu Upadhyaya, Wenzhen Yuan, Justin Lin, Edward H Adelson, and Sergey Levine. The feeling of success: Does touch sensing help predict grasp outcomes? *arXiv preprint arXiv:1710.05512*, 2017. 2.1, 2.2, 2.4, 2.6.2, 2.10.3

[20] Roberto Calandra, Andrew Owens, Dinesh Jayaraman, Justin Lin, Wenzhen Yuan, Jitendra Malik, Edward H. Adelson, and Sergey Levine. More Than a Feeling: Learning to Grasp and Regrasp Using Vision and Touch. *IEEE Robotics and Automation Letters*, 3(4):3300–3307, 2018. doi: 10.1109/LRA.2018.2852779. URL `https://arxiv.org/abs/1805.11085`. 1

[21] Guanqun Cao, Jiaqi Jiang, Danushka Bollegala, and Shan Luo. Learn from incomplete tactile data: Tactile representation learning with masked autoencoders. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 10800–10805. IEEE, 2023. 2.2, 2.3, 3.2.2

[22] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal,

Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9650–9660, 2021. URL https://openaccess.thecvf.com/content/ICCV2021/papers/Caron_Emerging_Properties_in_Self-Supervised_Vision_Transformers_ICCV_2021_paper.pdf. (document), 2.2, 2.3, 2.10.1, 3.3.2, 4.3

[23] Claire Chen, Zhongchun Yu, Hojung Choi, Mark Cutkosky, and Jeannette Bohg. Dexforce: Extracting force-informed actions from kinesthetic demonstrations for dexterous manipulation. *IEEE Robotics and Automation Letters*, 10(6): 6416–6423, 2025. doi: 10.1109/LRA.2025.3568318. 5.1

[24] Dian Chen, Brady Zhou, Vladlen Koltun, and Philipp Krähenbühl. Learning by cheating. In *Conference on robot learning*, pages 66–75. PMLR, 2020. 5.1, 5.2.2

[25] Tao Chen, Megha Tippur, Siyang Wu, Vikash Kumar, Edward Adelson, and Pulkit Agrawal. Visual dexterity: In-hand reorientation of novel and complex object shapes. *Science Robotics*, 8(84):eadc9244, 2023. 1, 5.2.1

[26] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 1597–1607. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/chen20j.html. 2.2

[27] Weihang Chen, Jing Xu, Fanbo Xiang, Xiaodi Yuan, Hao Su, and Rui Chen. General-purpose sim2real protocol for learning contact-rich manipulation with marker-based visuotactile sensors. *IEEE Transactions on Robotics*, 40:1509–1526, 2024. doi: 10.1109/TRO.2024.3352969. 2.10.1

[28] Xiaokang Chen, Mingyu Ding, Xiaodi Wang, Ying Xin, Shentong Mo, Yunhao Wang, Shumin Han, Ping Luo, Gang Zeng, and Jingdong Wang. Context autoencoder for self-supervised representation learning. *International Journal of Computer Vision*, 132(1):208–223, 2024. 2.4, 2.10.3, 4.4.1, 4.4.2, 4.6.3

[29] Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 15750–15758, 2021. 2.2, 2.10.1

[30] Yizhou Chen, Mark Van der Merwe, Andrea Sipos, and Nima Fazeli. Visuotactile transformers for manipulation. In Karen Liu, Dana Kulic, and Jeff Ichnowski, editors, *Proceedings of The 6th Conference on Robot Learning*, volume 205 of *Proceedings of Machine Learning Research*, pages 2026–2040. PMLR, 14–18 Dec 2023. URL https://proceedings.mlr.press/v205/chen23d.html. 2.2

[31] Zixi Chen, Shixin Zhang, Shan Luo, Fuchun Sun, and Bin Fang. Tacchi: A pluggable and low computational cost elastomer deformation simulator for optical tactile sensors. *IEEE Robotics and Automation Letters*, 8(3):1239–1246, 2023. doi: 10.1109/LRA.2023.3237042. 2.10.1

[32] Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *Proceedings of Robotics: Science and Systems (RSS)*, 2023. 2.7.1, 2.8, 2.10.3, 5.1, 5.2.3

[33] Cheng Chi, Zhenjia Xu, Chuer Pan, Eric Cousineau, Benjamin Burchfiel, Siyuan Feng, Russ Tedrake, and Shuran Song. Universal manipulation interface: In-the-wild robot teaching without in-the-wild robots. *arXiv preprint arXiv:2402.10329*, 2024. 4.3, 5.1

[34] Sanjiban Choudhury, Mohak Bhardwaj, Sankalp Arora, Ashish Kapoor, Gireeja Ranade, Sebastian Scherer, and Debadeepta Dey. Data-driven planning via imitation learning. *The International Journal of Robotics Research*, 37(13-14): 1632–1672, 2018. 5.1, 5.2.2

[35] Alex Church, John Lloyd, Raia Hadsell, and Nathan F Lepora. Deep reinforcement learning for tactile robotics: Learning to type on a braille keyboard. *IEEE Robotics and Automation Letters*, 5(4):6145–6152, 2020. 2.1, 2.2

[36] Samuel Clarke, Travers Rhodes, Christopher G. Atkeson, and Oliver Kroemer. Learning audio feedback for estimating amount and flow of granular material. In Aude Billard, Anca Dragan, Jan Peters, and Jun Morimoto, editors, *Proceedings of The 2nd Conference on Robot Learning*, volume 87 of *Proceedings of Machine Learning Research*, pages 529–550. PMLR, 29–31 Oct 2018. URL https://proceedings.mlr.press/v87/clarke18a.html. 4.4.1

[37] Samuel Clarke, Negin Heravi, Mark Rau, Ruohan Gao, Jiajun Wu, Doug James, and Jeannette Bohg. Diffimpact: Differentiable rendering and identification of impact sounds. In *Conference on Robot Learning*, pages 662–673. PMLR, 2022. 4.2

[38] Timothée Darcet, Maxime Oquab, Julien Mairal, and Piotr Bojanowski. Vision transformers need registers. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=2dnO3LLiJ1. 2.10.2

[39] Vedant Dave, Fotios Lygerakis, and Elmar Rueckert. Multimodal visual-tactile representation learning through self-supervised contrastive pre-training, 2024. 2.2, 3.2.2

[40] Frank Dellaert and GTSAM Contributors. borglab/gtsam, May 2022. URL https://github.com/borglab/gtsam). 5.5.1

[41] Siyuan Dong, Wenzhen Yuan, and Edward H Adelson. Improved gelsight tactile sensor for measuring geometry and slip. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 137–144. IEEE, 2017. 2.4, 2.5.2

[42] Siyuan Dong, Devesh Jha, Diego Romeres, Sangwoon Kim, Daniel Nikovski, and Alberto Rodriguez. Tactile-rl for insertion: Generalization to objects of unknown geometry. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021. URL https://arxiv.org/pdf/2104.01167.pdf. 5.2.3

[43] Siyuan Dong, Devesh K. Jha, Diego Romeres, Sangwoon Kim, Daniel Nikovski, and Alberto Rodriguez. Tactile-rl for insertion: Generalization to objects of unknown geometry. *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6437–6443, 2021. URL https://api.semanticscholar.org/CorpusID:233004667. 2.1, 3.4.2, 4.2, 4.4.2

[44] Elliott Donlon, Siyuan Dong, Melody Liu, Jianhua Li, Edward Adelson, and Alberto Rodriguez. Gelslim: A high-resolution, compact, robust, and calibrated tactile-sensing finger. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1927–1934, 2018. doi: 10.1109/IROS.2018.8593661. 4.2

[45] Elliott Donlon, Siyuan Dong, Melody Liu, Jianhua Li, Edward Adelson, and Alberto Rodriguez. GelSlim: A high-resolution, compact, robust, and calibrated tactile-sensing finger. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1927–1934, 2018. doi: 10.1109/IROS.2018.8593661. 2.1, 3.1, 3.2.1

[46] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 2.10.1, 2.10.2, 3.7.2

[47] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 4.3

[48] Yiming Dou, Fengyu Yang, Yi Liu, Antonio Loquercio, and Andrew Owens. Tactile-augmented radiance fields. *arXiv preprint arXiv:2405.04534*, 2024. 2.2, 2.3

[49] Yipai Du, Guanlan Zhang, Yazhan Zhang, and Michael Yu Wang. High-resolution 3-dimensional contact deformation tracking for fingervision sensor

with dense random color pattern. *IEEE Robotics and Automation Letters*, 6(2): 2147–2154, 2021. doi: 10.1109/LRA.2021.3061306. 2.1

[50] Ronald Fearing. Implementing a force strategy for object re-orientation. In *Proceedings. 1986 IEEE International Conference on Robotics and Automation*, volume 3, pages 96–102. IEEE, 1986. 5.2.1

[51] Zhengcong Fei, Mingyuan Fan, and Junshi Huang. A-JEPA: Joint-embedding predictive architecture can listen. *ArXiv*, abs/2311.15830, 2023. URL `https://api.semanticscholar.org/CorpusID:265456289`. 2.10.1

[52] Ruoxuan Feng, Jiangyu Hu, Wenke Xia, Tianci Gao, Ao Shen, Yuhao Sun, Bin Fang, and Di Hu. Anytouch: Learning unified static-dynamic representation across multiple visuo-tactile sensors. *arXiv preprint arXiv:2502.12191*, 2025. 4.2

[53] Jeremy A. Fishel and Gerald E. Loeb. Sensing tactile microvibrations with the BioTac — Comparison with human sensitivity. In *2012 4th IEEE RAS & EMBS International Conference on Biomedical Robotics and Biomechatronics (BioRob)*, pages 1122–1127, 2012. doi: 10.1109/BioRob.2012.6290741. URL `https://ieeexplore.ieee.org/document/6290741`. 3.2.1

[54] Letian Fu, Gaurav Datta, Huang Huang, William Chung-Ho Panitch, Jaimyn Drake, Joseph Ortiz, Mustafa Mukadam, Mike Lambeta, Roberto Calandra, and Ken Goldberg. A touch, vision, and language dataset for multimodal alignment. In *Forty-first International Conference on Machine Learning*, 2024. URL `https://openreview.net/forum?id=tFEOOH9eH0`. 2.2, 2.6.3

[55] Dhiraj Gandhi, Abhinav Gupta, and Lerrel Pinto. Swoosh! rattle! thump!–actions that sound. *arXiv preprint arXiv:2007.01851*, 2020. 4.2

[56] Junli Gao, Zhaoji Huang, Zhaonian Tang, Haitao Song, and Wenyu Liang. Visuo-Tactile-Based Slip Detection Using A Multi-Scale Temporal Convolution Network, 2023. URL `https://arxiv.org/abs/2302.13564`. 3.2.1

[57] Peng Gao, Teli Ma, Hongsheng Li, Ziyi Lin, Jifeng Dai, and Yu Qiao. Convmae: Masked convolution meets masked autoencoders. *arXiv preprint arXiv:2205.03892*, 2022. 2.10.1

[58] Ruihan Gao, Kangle Deng, Gengshan Yang, Wenzhen Yuan, and Jun-Yan Zhu. Tactile dreamfusion: Exploiting tactile sensing for 3d generation. In *Conference on Neural Information Processing Systems (NeurIPS)*, 2024. 4.2

[59] Ruohan Gao, Yen-Yu Chang, Shivani Mall, Li Fei-Fei, and Jiajun Wu. Objectfolder: A dataset of objects with implicit visual, auditory, and tactile representations. In *CoRL*, 2021. 2.2, 2.10.2

[60] Ruohan Gao, Zilin Si, Yen-Yu Chang, Samuel Clarke, Jeannette Bohg, Li Fei-

Fei, Wenzhen Yuan, and Jiajun Wu. ObjectFolder 2.0: A multisensory object dataset for sim2real transfer. In *CVPR*, 2022. (document), 2.1, 2.2, 2.3

[61] Ruohan Gao, Zilin Si, Yen-Yu Chang, Samuel Clarke, Jeannette Bohg, Li Fei-Fei, Wenzhen Yuan, and Jiajun Wu. ObjectFolder 2.0: A multisensory object dataset for sim2real transfer. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10598–10608, 2022. 2.2

[62] Ruohan Gao, Yiming Dou, Hao Li, Tanmay Agarwal, Jeannette Bohg, Yunzhu Li, Li Fei-Fei, and Jiajun Wu. The objectfolder benchmark: Multisensory learning with neural and real objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17276–17286, 2023. 2.2

[63] Abraham George, Selam Gano, Pranav Katragadda, and Amir Barati Farimani. VITaL Pretraining: Visuo-Tactile Pretraining for Tactile and Non-Tactile Manipulation Policies, 2024. URL `https://arxiv.org/abs/2403.11898`. 3.2.2

[64] Clément Godard, Oisin Mac Aodha, Michael Firman, and Gabriel J. Brostow. Digging into self-supervised monocular depth prediction. October 2019. 2.5.2, 2.10.3

[65] Daniel Fernandes Gomes, Paolo Paoletti, and Shan Luo. Beyond flat gelsight sensors: Simulation of optical tactile sensors of complex morphologies for sim2real learning. 2023. URL `https://www.roboticsproceedings.org/rss19/p035.pdf`. 2.10.1

[66] Yuan Gong, Yu-An Chung, and James Glass. Ast: Audio spectrogram transformer. *arXiv preprint arXiv:2104.01778*, 2021. 4.2

[67] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, et al. Bootstrap your own latent-a new approach to self-supervised learning. *Advances in neural information processing systems*, 33:21271–21284, 2020. 2.2, 2.10.1, 3.2.2, 3.3.1, 4, 5.4.1

[68] Xiaofeng Guo, Hung-Jui Huang, and Wenzhen Yuan. Estimating properties of solid particles inside container using touch sensing. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 8985–8992, 2023. doi: 10.1109/IROS55552.2023.10341880. 4.2

[69] Harsh Gupta, Yuchen Mo, Shengmiao Jin, and Wenzhen Yuan. Sensor-invariant tactile representation. *arXiv preprint arXiv:2502.19638*, 2025. 4.1, 4.2

[70] Irmak Guzey, Ben Evans, Soumith Chintala, and Lerrel Pinto. Dexterity from touch: Self-supervised pre-training of tactile representations with robotic play. In *7th Annual Conference on Robot Learning*, 2023. URL `https://openreview.net/forum?id=EXQ0eXtX3OW`. 2.2, 3.1, 3.2.2, 3.3.2, 3.3.2, 4

[71] Irmak Guzey, Haozhi Qi, Julen Urain, Changhao Wang, Jessica Yin, Krishna Bodduluri, Mike Lambeta, Lerrel Pinto, Akshara Rai, Jitendra Malik, Tingfan Wu, Akash Sharma, and Homanga Bharadhwaj. Dexterity from smart lenses: Multi-fingered robot manipulation with in-the-wild human demonstrations, 2025. URL https://arxiv.org/abs/2511.16661. 6

[72] Li Han and Jeffrey C Trinkle. Dextrous manipulation by rolling and finger gaiting. In *Proceedings. 1998 IEEE International Conference on Robotics and Automation (Cat. No. 98CH36146)*, volume 1, pages 730–735. IEEE, 1998. 5.2.1

[73] Ankur Handa, Karl Van Wyk, Wei Yang, Jacky Liang, Yu-Wei Chao, Qian Wan, Stan Birchfield, Nathan Ratliff, and Dieter Fox. Dexpilot: Vision-based teleoperation of dexterous robotic hand-arm system. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9164–9170. IEEE, 2020. URL https://arxiv.org/abs/1910.03135. 3.3.1, 6

[74] Ankur Handa, Arthur Allshire, Viktor Makoviychuk, Aleksei Petrenko, Ritvik Singh, Jingzhou Liu, Denys Makoviichuk, Karl Van Wyk, Alexander Zhurkevich, Balakumar Sundaralingam, et al. Dextreme: Transfer of agile in-hand manipulation from simulation to reality. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5977–5984. IEEE, 2023. 5.1, 5.2.1

[75] Johanna Hansen, Francois Hogan, Dmitriy Rivkin, David Meger, Michael Jenkin, and Gregory Dudek. Visuotactile-rl: Learning multimodal manipulation policies with deep reinforcement learning. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 8298–8304. IEEE, 2022. 2.2

[76] Nicklas Hansen, Zhecheng Yuan, Yanjie Ze, Tongzhou Mu, Aravind Rajeswaran, Hao Su, Huazhe Xu, and Xiaolong Wang. On pre-training for visuo-motor control: Revisiting a learning-from-scratch baseline. *arXiv preprint arXiv:2212.05749*, 2022. 2.8

[77] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 4.6.3

[78] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16000–16009, 2022. (document), 2.2, 2.2, 2.3, 2.10.1, 3.1, 3.2.2, 3.3.2

[79] Tairan He, Wenli Xiao, Toru Lin, Zhengyi Luo, Zhenjia Xu, Zhenyu Jiang, Jan Kautz, Changliu Liu, Guanya Shi, Xiaolong Wang, et al. Hover: Versatile neural whole-body controller for humanoid robots. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9989–9996. IEEE, 2025. 5.1

[80] Dan Hendrycks, Mantas Mazeika, Saurav Kadavath, and Dawn Song. Using self-supervised learning can improve model robustness and uncertainty. *Advances in neural information processing systems*, 32, 2019. 4.1

[81] Carolina Higuera, Joseph Ortiz, Haozhi Qi, Luis Pineda, Byron Boots, and Mustafa Mukadam. Perceiving extrinsic contacts from touch improves learning insertion policies. *arXiv preprint arXiv:2309.16652*, 2023. 2.1, 2.2

[82] Carolina Higuera, Akash Sharma, Chaithanya Krishna Bodduluri, Taosha Fan, Patrick Lancaster, Mrinal Kalakrishnan, Michael Kaess, Byron Boots, Mike Lambeta, Tingfan Wu, and Mustafa Mukadam. Sparsh: Self-supervised touch representations for vision-based tactile sensing. 2024. URL `https://openreview.net/forum?id=xYJn2e1uu8`. 3.1, 3.2.2, 4.1, 4.2, 4.3, 4.4.1, 5.2.3

[83] Carolina Higuera, Akash Sharma, Taosha Fan, Chaithanya Krishna Bodduluri, Byron Boots, Michael Kaess, Mike Lambeta, Tingfan Wu, Zixi Liu, Francois Robert Hogan, and Mustafa Mukadam. Tactile beyond pixels: Multisensory touch representations for robot manipulation. In *9th Annual Conference on Robot Learning*, 2025. URL `https://openreview.net/forum?id=sMs4pJYhWi`. 5.1, 5.2.3, 5.4.1, 5.6.3

[84] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. 2.10.1

[85] Francois R Hogan, Jean-François Tremblay, Bobak H Baghi, Michael Jenkin, Kaleem Siddiqi, and Gregory Dudek. Finger-STS: Combined proximity and tactile sensing for robotic manipulation. *IEEE Robotics and Automation Letters*, 7(4):10865–10872, 2022. 2.7.1

[86] Binghao Huang, Yixuan Wang, Xinyi Yang, Yiyue Luo, and Yunzhu Li. 3d-vitac: Learning fine-grained manipulation with visuo-tactile sensing. In *8th Annual Conference on Robot Learning*, 2024. 5.2.3

[87] Hung-Jui Huang, Xiaofeng Guo, and Wenzhen Yuan. Understanding Dynamic Tactile Sensing for Liquid Property Estimation. In *Proceedings of Robotics: Science and Systems*, New York City, NY, USA, June 2022. doi: 10.15607/RSS.2022.XVIII.072. 4.4.1

[88] Hung-Jui Huang, Michael Kaess, and Wenzhen Yuan. NormalFlow: Fast, Robust, and Accurate Contact-based Object 6DoF Pose Tracking with Vision-based Tactile Sensors. *IEEE Robotics and Automation Letters*, 2024. URL `https://ieeexplore.ieee.org/document/10766628`. 1, 4.2

[89] Physical Intelligence, Kevin Black, Noah Brown, James Darpinian, Karan Dhabalia, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, et al. π0. 5: a vision-language-action model with open-world generalization, 2025. *URL https://arxiv. org/abs/2504.16054*, 1(2):3. 6

[90] Andrew Jaegle, Sebastian Borgeaud, Jean-Baptiste Alayrac, Carl Doersch, Catalin Ionescu, David Ding, Skanda Koppula, Daniel Zoran, Andrew Brock, Evan Shelhamer, et al. Perceiver io: A general architecture for structured inputs & outputs. *arXiv preprint arXiv:2107.14795*, 2021. 2.5.2, 2.10.3

[91] Rico Jonschkowski, Austin Stone, Jonathan T Barron, Ariel Gordon, Kurt Konolige, and Anelia Angelova. What matters in unsupervised optical flow. *arXiv preprint arXiv:2006.04902*, 2020. 2.5.2, 2.10.3

[92] Shubham Kanitkar, Helen Jiang, and Wenzhen Yuan. PoseIt: a visual-tactile dataset of holding poses for grasp stability analysis. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 71–78. IEEE, 2022. 2.6.2

[93] Siddharth Karamcheti, Suraj Nair, Annie S. Chen, Thomas Kollar, Chelsea Finn, Dorsa Sadigh, and Percy Liang. Language-driven representation learning for robotics. In *Robotics: Science and Systems (RSS)*, 2023. 2.10.1

[94] Haresh Karnan, Elvin Yang, Daniel Farkash, Garrett Warnell, Joydeep Biswas, and Peter Stone. Sterling: Self-supervised terrain representation learning from unconstrained robot experience. *arXiv preprint arXiv:2309.15302*, 2023. 4.1

[95] Justin Kerr, Huang Huang, Albert Wilcox, Ryan Hoque, Jeffrey Ichnowski, Roberto Calandra, and Ken Goldberg. Self-supervised visuo-tactile pretraining to locate and follow garment features. In *Robotics: Science and Systems*, 2023. 2.2, 2.10.1

[96] Sangwoon Kim, Antonia Bronars, Parag Patre, and Alberto Rodriguez. Texterity–tactile extrinsic dexterity: Simultaneous tactile estimation and control for extrinsic dexterity. *arXiv preprint arXiv:2403.00049*, 2024. 2.6.1

[97] Raj Kolamuri, Zilin Si, Yufan Zhang, Arpit Agarwal, and Wenzhen Yuan. Improving grasp stability with rotation measurement from tactile sensing. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6809–6816. IEEE, 2021. 2.6.2

[98] Ashish Kumar, Zipeng Fu, Deepak Pathak, and Jitendra Malik. Rma: Rapid motor adaptation for legged robots. *arXiv preprint arXiv:2107.04034*, 2021. 4.4.2, 5.1, 5.2.2, 5.3.3, 5.6

[99] Mike Lambeta, Po-Wei Chou, Stephen Tian, Brian Yang, Benjamin Maloon, Victoria Rose Most, Dave Stroud, Raymond Santos, Ahmad Byagowi, Gregg Kammerer, Dinesh Jayaraman, and Roberto Calandra. Digit: A novel design for a low-cost compact high-resolution tactile sensor with application to in-hand manipulation. *IEEE Robotics and Automation Letters*, 5(3):3838–3845, 2020. doi: 10.1109/LRA.2020.2977257. 4.2, 4.3

[100] Mike Lambeta, Po-Wei Chou, Stephen Tian, Brian Yang, Benjamin Maloon,

Victoria Rose Most, Dave Stroud, Raymond Santos, Ahmad Byagowi, Gregg Kammerer, Dinesh Jayaraman, and Roberto Calandra. Digit: A novel design for a low-cost compact high-resolution tactile sensor with application to in-hand manipulation. *IEEE Robotics and Automation Letters*, 5(3):3838–3845, 2020. doi: 10.1109/LRA.2020.2977257. 2.1, 2.3, 3.1, 3.2.1, 4.1, 5.2.3

[101] Mike Lambeta, Huazhe Xu, Jingwei Xu, Po-Wei Chou, Shaoxiong Wang, Trevor Darrell, and Roberto Calandra. PyTouch: A machine learning library for touch processing. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 13208–13214. IEEE, 2021. 2.2

[102] Mike Lambeta, Tingfan Wu, Ali Sengul, Victoria Rose Most, Nolan Black, Kevin Sawyer, Romeo Mercado, Haozhi Qi, Alexander Sohn, Byron Taylor, Norb Tydingco, Gregg Kammerer, Dave Stroud, Jake Khatha, Kurt Jenkins, Kyle Most, Neal Stein, Ricardo Chavira, Thomas Craven-Bartle, Eric Sanchez, Yitian Ding, Jitendra Malik, and Roberto Calandra. Digitizing touch with an artificial multimodal fingertip, 2024. URL `https://arxiv.org/abs/2411.02479`. 4.1, 4.2, 4.3

[103] Yann LeCun. A path towards autonomous machine intelligence version 0.9. 2, 2022-06-27. *Open Review*, 62(1), 2022. 2.3, 2.10.1

[104] Nathan F Lepora. Soft biomimetic optical tactile sensing with the tactip: A review. *IEEE Sensors Journal*, 21(19):21131–21143, 2021. 4.1, 4.2

[105] Nathan F Lepora, Yijiong Lin, Ben Money-Coomes, and John Lloyd. Digitac: A digit-tactip hybrid tactile sensor for comparing low-cost high-resolution robot touch. *IEEE Robotics and Automation Letters*, 7(4):9382–9388, 2022. 2.1

[106] Hao Li, Yizhi Zhang, Junzhe Zhu, Shaoxiong Wang, Michelle A Lee, Huazhe Xu, Edward Adelson, Li Fei-Fei, Ruohan Gao, and Jiajun Wu. See, hear, and feel: Smart sensory fusion for robotic manipulation. In Karen Liu, Dana Kulic, and Jeff Ichnowski, editors, *Proceedings of The 6th Conference on Robot Learning*, volume 205 of *Proceedings of Machine Learning Research*, pages 1368–1378. PMLR, 14–18 Dec 2023. URL `https://proceedings.mlr.press/v205/li23c.html`. 4.2, 4.3

[107] Hongyu Li, Snehal Dikhale, Jinda Cui, Soshi Iba, and Nawid Jamali. HyperTaxel: Hyper-Resolution for Taxel-Based Tactile Signals Through Contrastive Learning. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7499–7506, 2024. doi: 10.1109/IROS58592.2024.10802001. URL `https://ieeexplore.ieee.org/document/10802001`. 3.2.1, 3.2.2

[108] Yitang Li, Zhengyi Luo, Tonghe Zhang, Cunxi Dai, Anssi Kanervisto, Andrea Tirinzoni, Haoyang Weng, Kris Kitani, Mateusz Guzek, Ahmed Touati, et al. Bfm-zero: A promptable behavioral foundation model for humanoid control

using unsupervised reinforcement learning. *arXiv preprint arXiv:2511.04131*, 2025. 5.1

[109] Boyuan Liang, Wenyu Liang, and Yan Wu. Tactile-Guided Dynamic Object Planar Manipulation. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3203–3209, 2022. doi: 10.1109/IROS47612.2022.9981270. URL `https://ieeexplore.ieee.org/document/9981270`. 3.2.1

[110] Hongzhuo Liang, Shuang Li, Xiaojian Ma, Norman Hendrich, Timo Gerkmann, Fuchun Sun, and Jianwei Zhang. Making sense of audio vibration for liquid height estimation in robotic pouring. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5333–5339, 2019. doi: 10.1109/IROS40897.2019.8968303. 4.4.1

[111] Yijiong Lin, John Lloyd, Alex Church, and Nathan F Lepora. Tactile gym 2.0: Sim-to-real deep reinforcement learning for comparing low-cost high-resolution robot touch. *IEEE Robotics and Automation Letters*, 7(4):10754–10761, 2022. 2.1, 2.2, 4.1

[112] Liang Liu, Jiangning Zhang, Ruifei He, Yong Liu, Yabiao Wang, Ying Tai, Donghao Luo, Chengjie Wang, Jilin Li, and Feiyue Huang. Learning by analogy: Reliable supervision from transformations for unsupervised optical flow estimation. In *IEEE Conference on Computer Vision and Pattern Recognition(CVPR)*, 2020. 2.5.2, 2.10.3

[113] Zeyi Liu, Cheng Chi, Eric Cousineau, Naveen Kuppuswamy, Benjamin Burchfiel, and Shuran Song. Maniwav: Learning robot manipulation from in-the-wild audio-visual data. In *8th Annual Conference on Robot Learning*, 2024. 4.2

[114] John Lloyd and Nathan F. Lepora. Goal-driven robotic pushing using tactile and proprioceptive feedback. *IEEE Transactions on Robotics*, 38(2):1201–1212, 2022. doi: 10.1109/TRO.2021.3104471. 2.1, 2.6.1

[115] Zhenyu Lu, Zihan Liu, Xingyu Zhang, Yan Liang, Yuming Dong, and Tianyu Yang. 3d force identification and prediction using deep learning based on a gelsight-structured sensor. *Sensors and Actuators A: Physical*, 367:115036, 2024. 2.1, 2.4

[116] Shan Luo, Nathan F Lepora, Wenzhen Yuan, Kaspar Althoefer, Gordon Cheng, and Ravinder Dahiya. Tactile robotics: An outlook. *IEEE Transactions on Robotics*, 2025. 1

[117] Daolin Ma, Elliott Donlon, Siyuan Dong, and Alberto Rodriguez. Dense tactile force estimation using gelSlim and inverse FEM. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 5418–5424. IEEE, 2019. 2.2, 2.4

[118] Yecheng Jason Ma, Shagun Sodhani, Dinesh Jayaraman, Osbert Bastani, Vikash

Kumar, and Amy Zhang. Vip: Towards universal visual reward and representation via value-implicit pre-training. *arXiv preprint arXiv:2210.00030*, 2022. 2.10.1

[119] Viktor Makoviychuk, Lukasz Wawrzyniak, Yunrong Guo, Michelle Lu, Kier Storey, Miles Macklin, David Hoeller, Nikita Rudin, Arthur Allshire, Ankur Handa, et al. Isaac gym: High performance gpu-based physics simulation for robot learning. *arXiv preprint arXiv:2108.10470*, 2021. 5.6.1

[120] Zhao Mandi, Yifan Hou, Dieter Fox, Yashraj Narang, Ajay Mandlekar, and Shuran Song. Dexmachina: Functional retargeting for bimanual dexterous manipulation, 2025. URL `https://arxiv.org/abs/2505.24853`. 6

[121] Carolyn Matl, Yashraj Narang, Ruzena Bajcsy, Fabio Ramos, and Dieter Fox. Inferring the material properties of granular media for robotic tasks. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2770–2777, 2020. doi: 10.1109/ICRA40945.2020.9197063. 4.4.1

[122] Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. *arXiv preprint arXiv:1802.03426*, 2018. URL `https://arxiv.org/abs/1802.03426`. 3.3.2

[123] Jared Mejia, Victoria Dean, Tess Hellebrekers, and Abhinav Gupta. Hearing touch: Audio-visual pretraining for contact-rich manipulation. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6912–6919. IEEE, 2024. 4.1, 4.2

[124] Pedro Morgado, Nuno Vasconcelos, and Ishan Misra. Audio-visual instance discrimination with cross-modal agreement. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12475–12486, 2021. 4.2

[125] Andrew S Morgan, Kaiyu Hang, Bowen Wen, Kostas Bekris, and Aaron M Dollar. Complex in-hand manipulation via compliance-enabled finger gaiting and multi-modal planning. *IEEE Robotics and Automation Letters*, 7(2):4821–4828, 2022. 5.2.1

[126] Anusha Nagabandi, Kurt Konolige, Sergey Levine, and Vikash Kumar. Deep dynamics models for learning dexterous manipulation. In *Conference on robot learning*, pages 1101–1112. PMLR, 2020. 5.2.1

[127] Arsha Nagrani, Shan Yang, Anurag Arnab, Aren Jansen, Cordelia Schmid, and Chen Sun. Attention bottlenecks for multimodal fusion. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 14200–14213. Curran Associates, Inc., 2021. URL `https://proceedings.neurips.cc/paper_files/paper/2021/`

file/76ba9f564ebbc35b1014ac498fafadd0-Paper.pdf. 4.2, 4.3

[128] Suraj Nair, Aravind Rajeswaran, Vikash Kumar, Chelsea Finn, and Abhinav Gupta. R3m: A universal visual representation for robot manipulation. *arXiv preprint arXiv:2203.12601*, 2022. 2.10.1

[129] Daisuke Niizumi, Daiki Takeuchi, Yasunori Ohishi, Noboru Harada, and Kunio Kashino. Byol for audio: Exploring pre-trained general-purpose audio representations. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 31:137–151, 2023. doi: 10.1109/TASLP.2022.3221007. 4.2

[130] Allison M Okamura, Niels Smaby, and Mark R Cutkosky. An overview of dexterous manipulation. In *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*, volume 1, pages 255–262. IEEE, 2000. 1, 5.2.1

[131] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. DINOv2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023. (document), 2.2, 2.3, 2.10.1, 1, 3.3.1, 3.3.2, 4.1, 4.3

[132] Utku Ozbulak, Hyun Jung Lee, Beril Boga, Esla Timothy Anzaku, Ho min Park, Arnout Van Messem, Wesley De Neve, and Joris Vankerschaver. Know your self-supervised learning: A survey on image-based generative and discriminative training. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856. URL https://openreview.net/forum?id=Ma25S4ludQ. Survey Certification. 2.2

[133] Chaoyi Pan, Changhao Wang, Haozhi Qi, Zixi Liu, Homanga Bharadhwaj, Akash Sharma, Tingfan Wu, Guanya Shi, Jitendra Malik, and Francois Hogan. Spider: Scalable physics-informed dexterous retargeting. *arXiv preprint arXiv:2511.09484*, 2025. 6

[134] Simone Parisi, Aravind Rajeswaran, Senthil Purushwalkam, and Abhinav Gupta. The unsurprising effectiveness of pre-trained vision models for control. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 17359–17371. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/parisi22a.html. 2.10.1

[135] Venkatesh Pattabiraman, Yifeng Cao, Siddhant Haldar, Lerrel Pinto, and Raunaq Bhirangi. Learning Precise, Contact-Rich Manipulation through Uncalibrated Tactile Skins. *arXiv preprint arXiv:2410.17246*, 2024. URL https://arxiv.org/abs/2410.17246. (document), 3.2.1, 3.3.2, 3.1, 3.4.2

[136] PaXini. ITPU multi-dimensional tactile sensing unit, 2025. URL https://www.paxini.com/ax. 3.1, 3.2.1

[137] Xue Bin Peng, Pieter Abbeel, Sergey Levine, and Michiel Van de Panne. Deepmimic: Example-guided deep reinforcement learning of physics-based character skills. *ACM Transactions On Graphics (TOG)*, 37(4):1–14, 2018. 5.1

[138] Lerrel Pinto, Marcin Andrychowicz, Peter Welinder, Wojciech Zaremba, and Pieter Abbeel. Asymmetric actor critic for image-based robot learning. *arXiv preprint arXiv:1710.06542*, 2017. 5.1, 5.3.3

[139] Johannes Pitz, Lennart Röstel, Leon Sievers, and Berthold Bäuml. Dextrous tactile in-hand manipulation using a modular reinforcement learning architecture. *arXiv preprint arXiv:2303.04705*, 2023. 5.1

[140] Haozhi Qi, Ashish Kumar, Roberto Calandra, Yi Ma, and Jitendra Malik. In-Hand Object Rotation via Rapid Motor Adaptation. In *Conference on Robot Learning (CoRL)*, 2022. (document), 4.4.2, 4.6.4, 5.1, 5.2.1, 5.2.2, 5.5.2, 5.6, 5.6.2, 5.2, 5.6.3

[141] Haozhi Qi, Brent Yi, Yi Ma, Sudharshan Suresh, Mike Lambeta, Roberto Calandra, and Jitendra Malik. General In-Hand Object Rotation with Vision and Touch. In *Conference on Robot Learning (CoRL)*, 2023. 1, 2.4, 2.6.1, 3.1, 3.3, 5.5.2

[142] Ilija Radosavovic, Tete Xiao, Stephen James, Pieter Abbeel, Jitendra Malik, and Trevor Darrell. Real-world robot learning with masked visual pre-training. In *Conference on Robot Learning*, pages 416–426. PMLR, 2023. 2.10.1

[143] René Ranftl, Katrin Lasinger, David Hafner, Konrad Schindler, and Vladlen Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2020. (document), 2.10.2, 2.10.3, 2.7

[144] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 12179–12188, 2021. 2.5.2, 2.10.3

[145] Branden Romero, Hao-Shu Fang, Pulkit Agrawal, and Edward Adelson. Eye-Sight Hand: Design of a Fully-Actuated Dexterous Robot Hand with Integrated Vision-Based Tactile Sensors and Compliant Actuation. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1853–1860, 2024. doi: 10.1109/IROS58592.2024.10802778. URL https://arxiv.org/abs/2408.06265. 3.1, 3.2.1

[146] Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*,

pages 627–635. JMLR Workshop and Conference Proceedings, 2011. 5.3.2, 5.4.2

[147] Daniela Rus. In-hand dexterous manipulation of piecewise-smooth 3-d objects. *The International Journal of Robotics Research*, 18(4):355–381, 1999. 5.2.1

[148] Ayumu Saito, Prachi Kudeshia, and Jiju Poovvancheri. Point-jepa: A joint embedding predictive architecture for self-supervised learning on point cloud. In *2025 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 7348–7357. IEEE, 2025. 2.10.1

[149] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017. 5.3.2, 5.4.1, 5.6.1

[150] Nur Muhammad Mahi Shafiullah, Anant Rai, Haritheja Etukuru, Yiqian Liu, Ishan Misra, Soumith Chintala, and Lerrel Pinto. On bringing robots home. *arXiv preprint arXiv:2311.16098*, 2023. 4.3

[151] Shuwei Shao, Zhongcai Pei, Xingming Wu, Zhong Liu, Weihai Chen, and Zhengguo Li. Iebins: Iterative elastic bins for monocular depth estimation. *Advances in Neural Information Processing Systems*, 36, 2024. 2.10.3

[152] Akash Sharma, Carolina Higuera, Chaithanya Krishna Bodduluri, Zixi Liu, Taosha Fan, Tess Hellebrekers, Mike Lambeta, Byron Boots, Michael Kaess, Tingfan Wu, Francois Robert Hogan, and Mustafa Mukadam. Self-supervised perception for tactile skin covered dexterous hands. In *9th Annual Conference on Robot Learning*, 2025. URL `https://openreview.net/forum?id=eLeCrM5PEO`. 1, 4.4.2, 5.2.3, 5.4.1, 5.6.3

[153] Yu She, Shaoxiong Wang, Siyuan Dong, Neha Sunil, Alberto Rodriguez, and Edward Adelson. Cable manipulation with a tactile-reactive gripper. *The International Journal of Robotics Research*, 40(12-14):1385–1401, 2021. URL `https://journals.sagepub.com/doi/10.1177/02783649211027233`. 5.2.3

[154] Hidetoshi Shimodaira. Improving predictive inference under covariate shift by weighting the log-likelihood function. *Journal of statistical planning and inference*, 90(2):227–244, 2000. 2.7.1

[155] Zilin Si and Wenzhen Yuan. Taxim: An example-based simulation model for gelsight tactile sensors. *arXiv preprint arXiv:2109.04027*, 2021. 2.10.1

[156] Zilin Si, Zirui Zhu, Arpit Agarwal, Stuart Anderson, and Wenzhen Yuan. Grasp stability prediction with sim-to-real transfer from tactile sensing. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7809–7816, 2022. doi: 10.1109/IROS47612.2022.9981863. 2.6.2

[157] Zilin Si, Zirui Zhu, Arpit Agarwal, Stuart Anderson, and Wenzhen Yuan. Grasp stability prediction with sim-to-real transfer from tactile sensing. In *2022*

IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 7809–7816. IEEE, 2022. 2.10.1

[158] Zilin Si, Gu Zhang, Qingwei Ben, Branden Romero, Zhou Xian, Chao Liu, and Chuang Gan. DIFFTACTILE: A physics-based differentiable tactile simulator for contact-rich robotic manipulation. In *The Twelfth International Conference on Learning Representations*, 2024. URL `https://openreview.net/forum?id=eJHnSg783t`. 2.10.1

[159] Ritvik Singh, Arthur Allshire, Ankur Handa, Nathan Ratliff, and Karl Van Wyk. Dextrah-rgb: Visuomotor policies to grasp anything with dexterous hands. *arXiv preprint arXiv:2412.01791*, 2024. 1, 5.1

[160] Yunlong Song, Kexin Shi, Robert Penicka, and Davide Scaramuzza. Learning perception-aware agile flight in cluttered environments. *arXiv preprint arXiv:2210.01841*, 2022. 5.2.2

[161] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. PWC-Net: CNNs for optical flow using pyramid, warping, and cost volume. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8934–8943, 2018. 2.5.2, 2.10.3

[162] Sudharshan Suresh, Maria Bauza, Kuan-Ting Yu, Joshua G Mangelson, Alberto Rodriguez, and Michael Kaess. Tactile slam: Real-time inference of shape and pose from planar pushing. In *2021 IEEE international conference on robotics and automation (ICRA)*, pages 11322–11328. IEEE, 2021. 5.2.3

[163] Sudharshan Suresh, Zilin Si, Stuart Anderson, Michael Kaess, and Mustafa Mukadam. MidasTouch: Monte-Carlo inference over distributions across sliding touch. In *Proc. Conf. on Robot Learning, CoRL*, Auckland, NZ, December 2022. URL `https://proceedings.mlr.press/v205/suresh23a/suresh23a.pdf`. 1, 2.1, 2.2, 2.3, 2.10.2

[164] Sudharshan Suresh, Haozhi Qi, Tingfan Wu, Taosha Fan, Luis Pineda, Mike Lambeta, Jitendra Malik, Mrinal Kalakrishnan, Roberto Calandra, Michael Kaess, Joseph Ortiz, and Mustafa Mukadam. Neural feels with neural fields: Visuo-tactile perception for in-hand manipulation. In *arXiv preprint arXiv:2312.1346*, December 2023. 2.1

[165] Sudharshan Suresh, Zilin Si, Stuart Anderson, Michael Kaess, and Mustafa Mukadam. Midastouch: Monte-carlo inference over distributions across sliding touch. In *Conference on Robot Learning*, pages 319–331. PMLR, 2023. 4.2

[166] Sudharshan Suresh, Haozhi Qi, Tingfan Wu, Taosha Fan, Luis Pineda, Mike Lambeta, Jitendra Malik, Mrinal Kalakrishnan, Roberto Calandra, Michael Kaess, Joseph Ortiz, and Mustafa Mukadam. Neural feels with neural fields: Visuo-tactile perception for in-hand manipulation. *Science Robotics*, page

adl0628, 2024. URL https://arxiv.org/abs/2312.13469. 1, 4.2

[167] Jie Tan, Tingnan Zhang, Erwin Coumans, Atil Iscen, Yunfei Bai, Danijar Hafner, Steven Bohez, and Vincent Vanhoucke. Sim-to-real: Learning agile locomotion for quadruped robots. *arXiv preprint arXiv:1804.10332*, 2018. 5.1

[168] Bingjie Tang, Michael A Lin, Iretiayo Akinola, Ankur Handa, Gaurav S Sukhatme, Fabio Ramos, Dieter Fox, and Yashraj Narang. Industreal: Transferring contact-rich assembly tasks from simulation to reality. *arXiv preprint arXiv:2305.17110*, 2023. 4.4.2

[169] Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. *Advances in neural information processing systems*, 30, 2017. 2.10.1

[170] Ian H. Taylor, Siyuan Dong, and Alberto Rodriguez. GelSlim 3.0: High-resolution measurement of shape, force and slip in a compact tactile-sensing finger. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 10781–10787, 2022. doi: 10.1109/ICRA46639.2022.9811832. 2.2, 2.4

[171] Abitha Thankaraj and Lerrel Pinto. That sounds right: Auditory self-supervision for dynamic robot manipulation. In *Conference on Robot Learning*, pages 1036–1049. PMLR, 2023. 4.2

[172] Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 23–30. IEEE, 2017. 5.2.1

[173] Tito Pradhono Tomo, Wai Keat Wong, Alexander Schmitz, Harris Kristanto, Alexandre Sarazin, Lorenzo Jamone, Sophon Somlor, and Shigeki Sugano. A modular, distributed, soft, 3-axis sensor system for robot hands. In *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*, pages 454–460. IEEE, 2016. URL https://ieeexplore.ieee.org/document/7803315. 3.1, 3.2.1, 3.3

[174] Tito Pradhono Tomo, Massimo Regoli, Alexander Schmitz, Lorenzo Natale, Harris Kristanto, Sophon Somlor, Lorenzo Jamone, Giorgio Metta, and Shigeki Sugano. A new silicone structure for uskin—a soft, distributed, digital 3-axis skin sensor and its integration on the humanoid robot icub. *IEEE Robotics and Automation Letters*, 3(3):2584–2591, 2018. 2.2

[175] Tito Pradhono Tomo, Alexander Schmitz, Wai Keat Wong, Harris Kristanto, Sophon Somlor, Jinsun Hwang, Lorenzo Jamone, and Shigeki Sugano. Covering a Robot Fingertip With uSkin: A Soft Electronic Skin With Distributed 3-Axis Force Sensitive Elements for Robot Hands. *IEEE Robotics and Automation Letters*, 3(1):124–131, 2018. doi: 10.1109/LRA.2017.2734965. URL https:

//ieeexplore.ieee.org/document/8000399. 3.1, 3.2.1, 5.2.3, 5.5.1

[176] Zhan Tong, Yibing Song, Jue Wang, and Limin Wang. Videomae: Masked autoencoders are data-efficient learners for self-supervised video pre-training. *Advances in neural information processing systems*, 35:10078–10093, 2022. 2.10.1

[177] Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017. 2.2, 3.2.2

[178] A Vaswani. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf. 3.3.2

[179] Filipe Veiga, Herke van Hoof, Jan Peters, and Tucker Hermans. Stabilizing novel objects by learning to predict tactile slip. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5065–5072, 2015. doi: 10.1109/IROS.2015.7354090. 2.4

[180] Jun Wang, Ying Yuan, Haichuan Che, Haozhi Qi, Yi Ma, Jitendra Malik, and Xiaolong Wang. Lessons from learning to spin "pens". In *CoRL*, 2024. 5.2.3

[181] Shaoxiong Wang, Yu She, Branden Romero, and Edward H Adelson. GelSight Wedge: Measuring High-Resolution 3D Contact Geometry with a Compact Robot Finger. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021. URL https://dl.acm.org/doi/10.1109/ICRA48506.2021.9560783. 3.1, 3.2.1

[182] Shaoxiong Wang, Mike Lambeta, Po-Wei Chou, and Roberto Calandra. TACTO: A fast, flexible, and open-source simulator for high-resolution vision-based tactile sensors. *IEEE Robotics and Automation Letters (RA-L)*, 7(2):3930–3937, 2022. ISSN 2377-3766. doi: 10.1109/LRA.2022.3146945. URL https://arxiv.org/abs/2012.08456. 2.10.1

[183] Wenhui Wang, Hangbo Bao, Li Dong, Johan Bjorck, Zhiliang Peng, Qiang Liu, Kriti Aggarwal, Owais Khan Mohammed, Saksham Singhal, Subhojit Som, et al. Image as a foreign language: Beit pretraining for all vision and vision-language tasks. *arXiv preprint arXiv:2208.10442*, 2022. 2.10.1

[184] Junhang Wei, Shaowei Cui, Jingyi Hu, Peng Hao, Shuo Wang, and Zheng Lou. Multimodal Unknown Surface Material Classification and Its Application to Physical Reasoning. *IEEE Transactions on Industrial Informatics*, 18(7):4406–4416, 2022. doi: 10.1109/TII.2021.3126601. URL https://ieeexplore.ieee.org/document/9609678. 3.2.1

[185] Goran Westling and Roland S Johansson. Factors influencing the force control during precision grip. *Experimental brain research*, 53(2):277–284, 1984. 1

[186] Tianhao Wu, Jinzhou Li, Jiyao Zhang, Mingdong Wu, and Hao Dong. Canonical Representation and Force-Based Pretraining of 3D Tactile for Dexterous Visuo-Tactile Policy Learning, 2024. URL `https://arxiv.org/abs/2409.17549`. 3.1, 3.2.2

[187] Yansong Wu, Zongxie Chen, Fan Wu, Lingyun Chen, Liding Zhang, Zhenshan Bing, Abdalla Swikir, Sami Haddadin, and Alois Knoll. Tacdiffusion: Force-domain diffusion policy for precise tactile manipulation. *arXiv preprint arXiv:2409.11047*, 2024. 3.4.2, 4.4.2

[188] Tete Xiao, Ilija Radosavovic, Trevor Darrell, and Jitendra Malik. Masked visual pre-training for motor control. *arXiv preprint arXiv:2203.06173*, 2022. 2.10.1

[189] Qizhe Xie, Minh-Thang Luong, Eduard Hovy, and Quoc V Le. Self-training with noisy student improves imagenet classification. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10687–10698, 2020. 2.10.1

[190] Zhenda Xie, Zheng Zhang, Yue Cao, Yutong Lin, Jianmin Bao, Zhuliang Yao, Qi Dai, and Han Hu. Simmim: A simple framework for masked image modeling. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9653–9663, 2022. 2.10.1

[191] Huazhe Xu, Yuping Luo, Shaoxiong Wang, Trevor Darrell, and Roberto Calandra. Towards learning to play piano with dexterous hands and touch. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 10410–10416. IEEE, 2022. 2.1, 2.2

[192] Mengda Xu, Han Zhang, Yifan Hou, Zhenjia Xu, Linxi Fan, Manuela Veloso, and Shuran Song. Dexumi: Using human hand as the universal manipulation interface for dexterous manipulation. *arXiv preprint arXiv:2505.21864*, 2025. 5.1

[193] Zhengtong Xu, Raghava Uppuluri, Xinwei Zhang, Cael Fitch, Philip Glen Crandall, Wan Shou, Dongyi Wang, and Yu She. Unit: Unified tactile representation for robot learning. *arXiv preprint arXiv:2408.06481*, 2024. 2.2

[194] Zhengtong Xu, Raghava Uppuluri, Xinwei Zhang, Cael Fitch, Philip Glen Crandall, Wan Shou, Dongyi Wang, and Yu She. UniT: Data efficient tactile representation with generalization to unseen objects, 2025. URL `https://arxiv.org/abs/2408.06481`. 4.1, 4.2

[195] Han Xue, Jieji Ren, Wendi Chen, Gu Zhang, Yuan Fang, Guoying Gu, Huazhe Xu, and Cewu Lu. Reactive diffusion policy: Slow-fast visual-tactile policy learning for contact-rich manipulation. *arXiv preprint arXiv:2503.02881*, 2025. 4.2

[196] Fengyu Yang, Chenyang Ma, Jiacheng Zhang, Jing Zhu, Wenzhen Yuan, and

Andrew Owens. Touch and go: Learning from human-collected vision and touch. *arXiv preprint arXiv:2211.12498*, 2022. (document), 2.1, 2.2, 2.2, 2.3, 2.10.2

[197] Fengyu Yang, Chao Feng, Ziyang Chen, Hyoungseob Park, Daniel Wang, Yiming Dou, Ziyao Zeng, Xien Chen, Rit Gangopadhyay, Andrew Owens, et al. Binding touch to everything: Learning unified multimodal tactile representations. *arXiv preprint arXiv:2401.18084*, 2024. 2.2, 2.3, 2.6.2, 2.10.1, 3.2.2

[198] Max Yang, Chenghua Lu, Alex Church, Yijiong Lin, Chris Ford, Haoran Li, Efi Psomopoulou, David AW Barton, and Nathan F Lepora. Anyrotate: Gravity-invariant in-hand object rotation with sim-to-real touch. *arXiv preprint arXiv:2405.07391*, 2024. 2.4, 2.6.1, 3.1, 3.3, 5.1, 5.5.2

[199] Jessica Yin, Haozhi Qi, Jitendra Malik, James Pikul, Mark Yim, and Tess Hellebrekers. Learning in-hand translation using tactile skin with shear and normal force sensing. *arXiv preprint arXiv:2407.07885*, 2024. URL https://arxiv.org/abs/2407.07885. 1, 3.3

[200] Jessica Yin, Haozhi Qi, Youngsun Wi, Sayantan Kundu, Mike Lambeta, William Yang, Changhao Wang, Tingfan Wu, Jitendra Malik, and Tess Hellebrekers. Osmo: Open-source tactile glove for human-to-robot skill transfer. *arXiv preprint arXiv:2512.08920*, 2025. 6

[201] Zhao-Heng Yin, Binghao Huang, Yuzhe Qin, Qifeng Chen, and Xiaolong Wang. Rotating without seeing: Towards in-hand dexterity through touch. *arXiv preprint arXiv:2303.10880*, 2023. URL https://www.roboticsproceedings.org/rss19/p036.pdf. 1, 5.1, 5.2.1

[202] Zhao-Heng Yin, Changhao Wang, Luis Pineda, Francois Hogan, Krishna Bodduluri, Akash Sharma, Patrick Lancaster, Ishita Prasad, Mrinal Kalakrishnan, Jitendra Malik, et al. Dexteritygen: Foundation controller for unprecedented dexterity. *arXiv preprint arXiv:2502.04307*, 2025. 1, 5.1, 6

[203] Sarah Young, Dhiraj Gandhi, Shubham Tulsiani, Abhinav Gupta, Pieter Abbeel, and Lerrel Pinto. Visual imitation made easy. In Jens Kober, Fabio Ramos, and Claire Tomlin, editors, *Proceedings of the 2020 Conference on Robot Learning*, volume 155 of *Proceedings of Machine Learning Research*, pages 1992–2005. PMLR, 16–18 Nov 2021. URL https://proceedings.mlr.press/v155/young21a.html. 4.3

[204] Kelin Yu, Yunhai Han, Qixian Wang, Vaibhav Saxena, Danfei Xu, and Ye Zhao. MimicTouch: Leveraging Multi-modal Human Tactile Demonstrations for Contact-rich Manipulation. In *8th Annual Conference on Robot Learning*, 2024. URL https://openreview.net/forum?id=7yMZAUkXa4. 3.2.2, 4.1, 4.2, 4.3

[205] Samson Yu, Kelvin Lin, Anxing Xiao, Jiafei Duan, and Harold Soh. Octopi:

Object property reasoning with large tactile-language models. *arXiv preprint arXiv:2405.02794*, 2024. 2.2

[206] Wenzhen Yuan, Rui Li, Mandayam A. Srinivasan, and Edward H. Adelson. Measurement of shear and slip with a gelsight tactile sensor. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 304–311, 2015. doi: 10.1109/ICRA.2015.7139016. 1, 2.1, 2.2, 2.4

[207] Wenzhen Yuan, Siyuan Dong, and Edward Adelson. GelSight: High-resolution robot tactile sensors for estimating geometry and force. *Sensors: Special Issue on Tactile Sensors and Sensing*, 17(12):2762 – 2782, November 2017. 2.1, 2.3, 3.1, 3.2.1, 4.1, 5.2.3

[208] Wenzhen Yuan, Siyuan Dong, and Edward H. Adelson. GelSight: High-resolution robot tactile sensors for estimating geometry and force. *Sensors*, 17(12), 2017. ISSN 1424-8220. doi: 10.3390/s17122762. URL `https://www.mdpi.com/1424-8220/17/12/2762`. 4.2, 4.2, 4.3

[209] Wenzhen Yuan, Yuchen Mo, Shaoxiong Wang, and Edward H. Adelson. Active clothing material perception using tactile sensing and deep learning. *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1–8, 2017. URL `https://api.semanticscholar.org/CorpusID:3543735`. 2.4, 2.6.3, 2.10.3

[210] Wenzhen Yuan, Yuchen Mo, Shaoxiong Wang, and Edward H. Adelson. Active clothing material perception using tactile sensing and deep learning. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4842–4849, 2018. doi: 10.1109/ICRA.2018.8461164. 4.2

[211] Kevin Zakka, Baruch Tabanpour, Qiayuan Liao, Mustafa Haiderbhai, Samuel Holt, Jing Yuan Luo, Arthur Allshire, Erik Frey, Koushil Sreenath, Lueder A Kahrs, et al. Mujoco playground. *arXiv preprint arXiv:2502.08844*, 2025. 5.1

[212] Ben Zandonati, Ruohan Wang, Ruihan Gao, and Yan Wu. Investigating vision foundational models for tactile representation learning. *arXiv preprint arXiv:2305.00596*, 2023. 2.1, 2.2

[213] Andrea Zangrandi, Marco D'Alonzo, Christian Cipriani, and Giovanni Di Pino. Neurophysiology of slip sensation and grip reaction: insights for hand prosthesis control of slippage. *Journal of neurophysiology*, 126(2):477–492, 2021. 2.3, 2.10.3

[214] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 3836–3847, October 2023. (document), 4.6, 4.4.2

[215] Jialiang Zhao, Yuxiang Ma, Lirui Wang, and Edward Adelson. Transferable

Tactile Transformers for Representation Learning Across Diverse Sensors and Tasks. In *8th Annual Conference on Robot Learning*, 2024. URL https://openreview.net/forum?id=KXsropnmNI. 2.2, 3.2.2, 4.1, 4.2, 5.2.3

[216] Tony Z Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. *arXiv preprint arXiv:2304.13705*, 2023. 2.7.1, 2.8

[217] Tony Z. Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning Fine-Grained Bimanual Manipulation with Low-Cost Hardware. In *Proceedings of Robotics: Science and Systems*, Daegu, Republic of Korea, July 2023. doi: 10.15607/RSS.2023.XIX.016. 3.4.2

[218] Tony Z Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. *arXiv preprint arXiv:2304.13705*, 2023. 4.4.2, 5.1, 5.2.3

[219] Yongqiang Zhao, Xingshuo Jing, Kun Qian, Daniel Fernandes Gomes, and Shan Luo. Skill generalization of tubular object manipulation with tactile sensing and sim2real learning. *Robotics and Autonomous Systems*, 160:104321, 2023. ISSN 0921-8890. doi: https://doi.org/10.1016/j.robot.2022.104321. URL https://www.sciencedirect.com/science/article/pii/S092188902200210X. 2.10.1

[220] Jinghao Zhou, Chen Wei, Huiyu Wang, Wei Shen, Cihang Xie, Alan Yuille, and Tao Kong. ibot: Image bert pre-training with online tokenizer. *arXiv preprint arXiv:2111.07832*, 2021. 2.10.1

[221] Yi Zhou, Connelly Barnes, Jingwan Lu, Jimei Yang, and Hao Li. On the continuity of rotation representations in neural networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5745–5753, 2019. 5.4.3, 5.5.3

[222] Ziwen Zhuang, Zipeng Fu, Jianren Wang, Christopher Atkeson, Soeren Schwertfeger, Chelsea Finn, and Hang Zhao. Robot parkour learning. *arXiv preprint arXiv:2309.05665*, 2023. 5.2.2