Brain-Aligned Tactile Representations for Dexterous Robot Learning

Trinity J. Chung

CMU-RI-TR-26-65

June 2026

School of Computer Science
Department of Robotics Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

Thesis Committee:
Aran Nayebi, Chair
Katerina Fragkiadaki
Akash Sharma (Amazon Frontier AI & Robotics)

Thesis submitted in partial fulfillment of the requirements for the degree of Master of Science in Robotics

Acknowledgments

I am deeply grateful to my advisor, Aran Nayebi, whose generous support and guidance made this work possible. I thank Katerina Fragkiadaki for sharing her expertise in robot learning and for continually pushing my research forward, and Akash Sharma for his insights on tactile sensing that shaped how I think about the field.

I owe a great deal to my collaborators. Yuchen Shen was my first co-author, on the rodent-brain work, and a good friend throughout. Yiling Qiao brought me on at Genesis AI and entrusted me with the sensor simulation effort, and Alexis Duburcq was an exceptional mentor who taught me much of what I know about efficient simulation. Kashu Yamazaki built the learning framework my robot experiments relied on, and was a collaborator throughout. I am also thankful to all my friends and labmates in the Robotics Institute, who shared so many laughs and drinks along the way and made these years far from home a bit more homey.

Finally, I thank my parents for always believing in me; your love is the foundation of my confidence and my motivation to help build a better future a little sooner. I'm also very thankful to my brother who has always supported me and kept me grounded in reality.

This material is based upon work supported by the National Science Foundation Graduate Research Fellowship Program under Grant No(s) DGE2140739. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

Abstract

Touch is the essential sensory modality through which animals and robots physically negotiate the world. While much of robotic touch focuses on the capabilities of currently available tactile hardware, this thesis asks a more general question: what forms of tactile processing and representation could allow robots to approach the dexterity of animals? This thesis argues that simulated force and torque provide a privileged representation of touch: one that is biologically grounded, computationally scalable, and effective for dexterous behavior.

For biological touch, the rodent vibrissal pathway provides a tractable model system for studying tactile processing. We train task-optimized temporal networks on realistic whisker force/torque sequences to identify shapes and compare their internal representations against recordings from rats' barrel cortex. We find that convolutional recurrent models (ConvRNNs) align very closely with neural data, and contrastive self-supervision with tactile-specific augmentations matches supervised alignment as a label-free proxy.

In robotic touch, we test whether the same class of representation is useful for dexterous behavior. We develop a GPU-parallel tactile sensor simulator that exposes a family of tactile abstractions under one interface, from binary contact and per-taxel force/torque to marker displacement and temperature, fast enough to serve as the front-end for dexterous reinforcement learning. For in-hand manipulation tasks, we find that sensor placement matters more than sensor type or resolution, that whole-hand coverage closes most of the gap to a privileged teacher. Across 3 dexterous tasks, per-taxel force/torque emerges as the most useful and robust observation.

Together, these results argue that force and torque are not merely convenient signals to simulate, but a biologically grounded and behaviorally effective representation of touch. With recurrence and broad spatial coverage, simulated force and torque provide a common representational substrate for understanding tactile computation in brains and building tactile intelligence in robots.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

Touch is the sensory modality through which animals and robots physically negotiate the world, yet it remains comparatively underexplored—both in neuroscience, where the computations underlying somatosensory processing are still poorly characterized, and in robotics, where it is unclear which tactile signals a policy actually needs. This thesis approaches that gap from two complementary directions, unified by a single question: *which tactile representations matter for behavior, and why?*

1.1 Motivation

Tactile sensing lags vision and language in both biological understanding and artificial performance. Part of the reason is that touch is intrinsically active and embodied: a tactile signal exists only through physical contact and is shaped by the motion that produced it, so it cannot be harvested at internet scale the way images or text can. On the biological side, the mechanical inputs to the somatosensory system—the forces at a whisker base, the pressure folding the skin of a fingertip—are difficult to measure directly, and computational models of how the brain processes them remain scarce. On the engineering side, tactile hardware is fragmented across many transduction principles with sharply different coverage and fidelity, and it is still unclear which of the signals a sensor *could* provide a policy actually needs.

These two gaps are two faces of the same question, and there are two complementary ways to attack it. One is to *reverse-engineer* a biological system that already solves the problem: by searching the space of neural network models for those whose internal representations best match recorded brain activity, we can read off the computational ingredients the brain relies on. The other is to *forward-engineer* robots until they solve manipulation tasks, and ask which tactile abstraction is sufficient to do so. The first identifies what a working tactile system *is*; the second tests what a tactile representation must *do*. Pursued together, they offer convergent evidence about the same underlying question.

In each case, simulation of tactile input is the shared enabling tool, and for the same reason: it supplies realistic tactile signals at a scale and level of control that hardware cannot. The rodent-brain study uses a biomechanical whisker-array simulator to generate the high-variation force/torque sequences needed to constrain goal-driven models, while the robot study uses a GPU-parallel tactile-sensor simulator to expose many sensing abstractions under one interface and vary them in a controlled experiment. This methodological throughline—realistic, scalable tactile simulation—is what makes both directions possible.

1.2 Thesis Contributions

The central claim of this thesis is that effective tactile processing—in the brain and in a robot alike—rests on a small set of shared representational ingredients: low-level force and torque as the most informative and biologically aligned tactile signal, recurrent integration of those signals over time, and broad spatial coverage of the sensing surface. These ingredients can be identified from two independent directions—by matching the activity of a biological tactile system and by maximizing a robot's success on dexterous tasks—and biomechanically grounded, scalable simulation of tactile input is the common tool that makes either identification possible. The chapters that follow establish these two directions; read together (Chapter 5), their convergence on the same ingredients is what supports the claim.

Concretely, this thesis makes the following contributions:

- **Tactile processing in the rodent brain** (Chapter 3): searching over model architectures on realistic whisker inputs, we identify convolutional recurrent encoders as the architectures whose representations best align with rodent somatosensory cortex—the first quantitative model–brain comparison in this system.
- **Scalable tactile simulation for robot learning** (Chapter 4): a GPU-parallel tactile sensor simulator exposing a family of tactile abstractions under a common interface, used to ablate which tactile signals dexterous manipulation policies actually need, with transfer to real hardware.

1.3 Thesis Organization

Chapter 2 reviews biological tactile sensing, tactile sensor hardware, tactile simulation, and tactile models that the later chapters build on. Chapter 3 presents the rodent-brain modeling work, and Chapter 4 presents the robot tactile-simulation work. Chapter 5 draws the two threads together and outlines future directions.

Chapter 2

Background

This chapter reviews the biological, hardware, simulation, and modeling background that the rest of the thesis builds on. Reverse-engineering the tactile representations of the rodent brain and forward-engineering tactile sensing for dexterous robots approach touch from opposite directions, but rest on a common foundation: an understanding of how biological systems transduce and process touch, of the artificial sensors that attempt to imitate them, of the simulators that make either tractable to study at scale, and of the learning-based models that consume tactile signals in both domains.

2.1 Biological Tactile Sensing

Touch is the first sense to develop and arguably the most indispensable: an organism cannot survive long without it, since it underlies everything from withdrawing from tissue-damaging stimuli to the fine motor control of feeding, grasping, and locomotion [13, 14]. Unlike vision or audition, touch is intrinsically *active* and *bidirectional*: to perceive a surface is to deform it, and the motor act of exploration shapes the very signals that are sensed. Human haptic perception fuses three streams—the *cutaneous* signals of receptors in the skin, the *kinesthetic* signals that report the configuration and loading of the limbs, and the *motor* commands that drive exploration [13, 14]. Together these answer two complementary questions about an object in the hand: *what* it is (its material, shape, and identity) and *where* it is relative to the body. Because the answers are recovered jointly through interaction, models of touch are best framed as a closed sensorimotor loop rather than as passive, tactile-only readout—a theme that recurs throughout this thesis.

2.1.1 Cutaneous Mechanoreceptors

The glabrous (hairless) skin of the human hand is innervated by four populations of mechanoreceptive afferents, distinguished by how quickly they adapt to a sustained stimulus—slowly (SA) versus rapidly/fast (RA/FA)—and by the size of their receptive fields—type I

(small and dense) versus type II (large and diffuse); Table Table 2.1 summarizes their division of labor [13, 14]. SA-I afferents (Merkel disks) respond to static indentation and low-frequency deformation and carry fine spatial form such as edges, curvature, and Braille-like patterns; FA-I afferents (Meissner corpuscles) are insensitive to static force but report low-frequency skin motion (∼5–50 Hz) such as incipient slip, which makes them central to grip control; FA-II afferents (Pacinian corpuscles) are exquisitely sensitive to high-frequency vibration (∼40–400 Hz) and to transients propagating through a held tool, so that one effectively feels the world at the tool's tip; and SA-II afferents (Ruffini endings) sense skin stretch and thereby encode hand conformation and the direction of applied force [13].

Afferent (corpuscle)	Adaptation, field	Frequency band	Primary role
SA-I (Merkel)	slow, small/dense	static–∼5 Hz	form, edges, curvature, sustained pressure
FA-I (Meissner)	fast, small/dense	∼5–50 Hz	skin motion and slip detection, grip control
FA-II (Pacinian)	fast, large/diffuse	∼40–400 Hz	vibration and transients, including through held tools
SA-II (Ruffini)	slow, large/diffuse	static (skin stretch)	skin stretch, hand conformation, force direction

Table 2.1. **The four cutaneous mechanoreceptor afferents of the human k glabrous hand**, organized by adaptation rate (slow/fast) and receptive-field type (I, small and dense; II, large and diffuse). Together they encode the same contact simultaneously at several spatial and temporal scales.

Two properties of this population matter when designing an artificial analogue. *Spatially*, acuity is not uniform: receptive fields are smallest and most densely packed at the fingertips and grow progressively coarser toward the palm, giving the hand both broad coverage and high resolution exactly where fine manipulation demands it. The fidelity is remarkable—human afferents resolve features on the scale of a single fingerprint ridge [15], and the population signals edge orientation quickly and accurately enough to steer in-hand manipulation [16]. *Temporally*, the skin resolves successive taps about 5 ms apart—finer than vision but coarser than audition—and the SA/FA split means a single contact is encoded across multiple timescales at once. Crucially, the skin is compliant: it folds around whatever it touches, and the pressure distribution over the *deformed* surface is what the SA-I population transduces into a percept of local geometry, letting humans judge curvature over a vast range from flat surfaces to sharp edges. This deformation is the signal itself, not incidental

detail, which is why models that treat the skin as a rigid plane and read out responses from undeformed contact tend to inject noise into the predicted afferent activity [17]; capturing it faithfully favors deformable, finite-element treatments of the contact interface [18].

2.1.2 Thermal and Kinesthetic Channels

Beyond mechanoreception, thermoreceptors in the skin add a thermal channel that helps identify materials. The percept of an object's temperature is driven not by its absolute temperature but by how much the skin's own temperature *changes* on contact, which depends on the material's thermal conductivity and thus distinguishes, for example, metal from wood at the same ambient temperature [19]. The thermal system is also asymmetric and protective: warm receptors operate over a smaller dynamic range than cold ones, because temperatures past that range would damage tissue—an asymmetry worth mirroring when an artificial thermal sensor is designed.

The kinesthetic stream, transduced by receptors in muscles, tendons, and joints, reports the position, movement, and loading of the limbs. It is what lets the haptic system localize a contact *relative to the body* and, through interaction, estimate properties that no single contact reveals: object weight, for instance, is recovered largely from an object's resistance to rotation, formalized as the inertia tensor of the hand-plus-object system [20]. Supplying the analogous joint configuration and forces/torques to a model is therefore as important as the cutaneous signal itself.

2.1.3 Active Touch and Exploratory Procedures

Because touch is active, humans deploy stereotyped *exploratory procedures* matched to the property they wish to perceive: lateral motion across a surface for texture, static pressure against resistance for compliance and softness, enclosure for global shape, and so on [13, 14]. The chosen procedure is itself shaped by the material—explorations of soft and granular materials, from silk to sand, follow systematically different strategies than those of rigid objects [21, 22]—which is one reason a simulator able to model deformables, grains, and fluids is attractive for studying touch. At the level of neural representation, the afferent population projects to topographically organized maps in the spinal cord and somatosensory cortex that behave less like a bank of independent pixels than like a "haptic retina," in which populations of receptors jointly encode features such as edge orientation [16].

2.1.4 The Rodent Vibrissal System

While the human hand is the richest tactile organ, much of what is known about the *neural* processing of touch comes from a more experimentally tractable model: the rodent vibrissal ("whisker") system. Rodents sweep an array of roughly thirty facial whiskers per side across objects at 15–20 Hz, and although the whiskers bear no receptors along their length, bending and torsion at each follicle are transduced into afferent signals whose statistics are set by the array's morphology [1, 23, 24]. The pathway is strikingly vision-like—hierarchical and recurrent, from the trigeminal ganglion through thalamus to primary and secondary somatosensory ("barrel") cortex [25–28]—and the whisker, dense in Merkel afferents at its base, has been argued to be a close functional analogue of the human fingertip. This combination of fine experimental control with a clean, temporally structured input makes the vibrissal system an ideal setting for reverse-engineering tactile computation, which Chapter 3 pursues.

2.2 Tactile Sensor Hardware

Reproducing even a fraction of this biological capability in hardware has proven difficult: producing a compliant artificial skin that deforms and *stays* deformed around an object has been an open problem in haptics for roughly four decades [29, 30], and the resulting sensor landscape is fragmented across many transduction principles, each with its own resolution, bandwidth, footprint, durability, and cost. Recent surveys of flexible tactile sensing organize this space by the physical effect used to convert a mechanical stimulus into a measurable signal [31]. *Piezoresistive* sensors translate stimulus-induced deformation of a conductive network into a resistance change; they are simple to read out and sensitive to both static and dynamic loads, but face a trade-off between sensing range and sensitivity and suffer from hysteresis and drift [31, 32]. *Capacitive* sensors measure the change in capacitance as an applied force alters the gap or permittivity of a dielectric layer between electrodes, offering low power consumption and high sensitivity at the cost of susceptibility to crosstalk and electromagnetic interference [31, 33]. *Piezoelectric* and *triboelectric* sensors convert mechanical deformation and contact electrification, respectively, directly into charge or potential differences, which makes them fast and self-powered but intrinsically better suited to dynamic than to static stimuli [31]. *Optical* mechanisms transduce contact into a change in transmitted or reflected light, providing high spatial resolution and immunity to electromagnetic interference, while *magnetic* mechanisms infer deformation from changes in a magnetic field; the latter decouple the sensing electronics from the contact interface, giving a wide measurement range, low hysteresis, and good robustness [31, 34]. This taxonomy makes concrete a

recurring theme: no single transduction principle simultaneously optimizes resolution, bandwidth, range, robustness, and form factor, so each choice commits a sensor—and any robot built around it—to a particular regime.

On robot hands specifically, these principles instantiate as a handful of families with sharply different coverage characteristics. Capacitive arrays and strain-gauge fingertips provide direct, repeatable force readings but are typically confined to the distal pad of a finger [35, 36]; recent six-axis capacitive interfaces push toward simultaneous multidirectional force and contact-location sensing at fingertip scale [35], and hair-in-elastomer piezoresistive structures explicitly decouple normal from shear stress [32]. Vision-based elastomer sensors such as GelSight image the deformation of a gel pad with an internal camera, resolving fine surface geometry at high resolution but remaining bulky, restricted to flat pads, and yielding only a local height map that must be resampled over many contacts to recover an object's geometry—unlike the distributed, actively sampled mechanoreceptor population they stand in for [37]. Magnetic skins such as ReSkin estimate deformation indirectly from magnetometer displacements, trading spatial resolution for a thin, replaceable form factor that can cover larger, curved areas of a hand [38]. Multimodal fingertips combine several of these channels—force, vibration through contact microphones, and high-resolution deformation—into a single unit at the cost of substantial wiring and calibration [36, 39–42]. A complementary line of work attacks the wiring-and-resolution trade-off algorithmically, using deep models to synthesize high-resolution tactile fields from sparsely distributed taxels [43]. The practical consequence, central to Chapter 4, is that these sensors differ not only in fidelity but in *where* on the hand they can be placed, so that adopting a given sensor effectively defines a different hand and makes apples-to-apples comparison across sensing strategies infeasible in hardware alone.

2.3 Tactile Simulation

Because comparing tactile sensors and acquiring tactile neural stimuli fairly in hardware is rarely feasible, simulation has become the central tool for studying touch at scale in both neuroscience and robotics. On the neuroscience side, biomechanical simulation of the whisker array is what makes goal-driven modeling of the somatosensory system possible: WHISKiT Physics provides the first complete three-dimensional model of the rodent vibrissal array, representing each whisker as a chain of rigid conical segments coupled by torsional springs and predicting the time-varying forces and torques generated at each whisker base during active sampling [2]. Combined with quantitative array morphology [1], such simulators generate the high-variation, biomechanically grounded force/torque sequences that drive the

models in Chapter 3.

On the robotics side, tactile simulators broadly divide into two families. The first uses full deformable physics, typically the Finite Element Method (FEM), to model the sensing medium directly: DIFFTACTILE builds a differentiable FEM soft-body simulator supporting diverse contact modes and material types [18], and Taccel combines incremental potential contact with affine body dynamics to scale vision-based tactile simulation to thousands of parallel environments at an order-of-magnitude speedup over real time [44]. The second family extracts a tactile signal from the contact queries of a rigid-body engine and post-processes it into a sensor readout, trading some physical fidelity for speed: FOTS approximates the optical response and marker motion of vision-based sensors [4], Tacmap unifies simulation and reality through a geometry-consistent penetration-depth map [6], and HydroShear produces GPU-parallel marker displacement fields that capture shear and twist [5]. Sensor models that target a specific transduction principle—for example, ternary shear and binary normal force for magnetic skins [45]—and mixed real-plus-simulated datasets for benchmarking [46] round out the landscape. Across both families the dominant trend is toward GPU parallelism, since policy learning requires thousands of simultaneous environments; the platform of Chapter 4 pushes this trend further by exposing many sensor abstractions under one interface so that the abstraction itself, rather than the hardware, can be varied in a controlled study.

2.4 Tactile Models

A range of learning-based models consume tactile signals, from goal-driven networks used to explain neural data to self-supervised encoders that turn high-dimensional touch into representations for robot policies. The two subsections below trace these two lineages.

2.4.1 Goal-Driven / Task-Optimized Neural Models

Task-optimized neural networks are currently the most quantitatively accurate framework for modeling sensory brain areas. The approach first succeeded in the primate ventral visual stream, where networks optimized for object categorization predicted neural responses across hierarchical cortical stages without fitting to the neural data [47, 48], and was subsequently extended to auditory cortex [49], to language [50], and to rodent systems, where comparatively shallow networks trained with self-supervised objectives best capture mouse visual cortex [51, 52]. The same goal-driven logic has more recently been applied to higher-level cognitive computations such as mental simulation and future prediction [53], underscoring its generality across brain areas and tasks. Despite this breadth, the framework had been

applied to tactile processing only once: the foundational whisker-trigeminal model of Zhuang et al. [54] trained simple recurrent networks on simulated whisker sweeps but used only supervised objectives and was never compared against neural recordings.

Realizing this framework for touch requires choices about temporal architecture and training objective. Because tactile input is an intrinsically temporal force/torque stream, candidate architectures include convolutional recurrent networks shown to match primate cortical dynamics [55], modern state-space models such as S4 [56] and Mamba [57], and attention-based temporal models [58]. On the objective side, beyond supervised categorization, self-supervised losses—contrastive methods [59], non-contrastive self-distillation [60], and autoencoding [61]—offer label-free, ethologically more plausible training signals, and have been shown to match or exceed supervised models in predicting both primate and mouse visual cortex [51, 62]. Chapter 3 draws on these architectural and objective choices to build goal-driven models of the rodent somatosensory system.

2.4.2 Tactile Representation Learning for Robots

On the robotics side, the central question is not how to explain neural data but which tactile abstraction a policy actually needs, and prior work typically commits to a single point in the design space because the underlying hardware does. Empirically, even very coarse signals can be surprisingly sufficient: sparse binary contacts alone support several dexterous in-hand skills, particularly for interactions—such as decoupled robot–object motion—that proprioception cannot register [63], while richer ternary-shear and normal-force skins enable robust in-hand translation that outperforms proprioception- or shear-only baselines [45]. This sensitivity of policy performance to the choice of tactile abstraction is exactly what Chapter 4 isolates by holding task, hand, and policy fixed while varying the sensor.

A dominant strategy for turning high-dimensional, sensor-specific touch into usable features is self-supervised pre-training, mirroring the trajectory of representation learning in vision. Early work showed that self-supervised tactile encoders trained on a few hours of robotic play yield embeddings that make downstream dexterous policies learnable from only a handful of demonstrations [64]. This idea has since matured into general-purpose touch foundation models: a family of self-supervised encoders pre-trained on large corpora of unlabeled tactile images provides transferable representations across vision-based sensors of differing form factor [65], and has been extended in two complementary directions that together address the coverage and richness limitations of fingertip hardware. The first fuses multiple touch modalities—image, audio, motion, and pressure—into a unified representation, capturing physical properties that any single channel misses and substantially improving policy success and state-recovery robustness [41]. The second targets hand-wide coverage

rather than fingertip resolution, pre-training a self-supervised encoder for magnetic skin distributed across the fingertips, phalanges, and palm of a dexterous hand, on the premise that full-hand tactile perception—not just fingertip sensing—is what dexterity requires [66]. How much whole-hand coverage actually matters, relative to sensor type and resolution, is a question Chapter 4 examines directly.

A related thread uses tactile representations to bridge the gap between human demonstration and robot execution. Because humans manipulate fluently under rich natural touch feedback but on a different embodiment than any robot, recent methods learn shared tactile latent spaces that transfer human-collected signals to robots without paired data or identical sensors [67], retarget kinematic-only human demonstrations into dynamically feasible, force-aware robot trajectories at scale [68], or combine reinforcement-learned low-level motor primitives with coarse human commands into a foundation controller for dexterous manipulation [69]. Touch has likewise been bound into broader multimodal embedding spaces alongside vision, language, and audio [42, 70, 71], and tactile feedback increasingly underpins robust dexterous grasping and manipulation of challenging objects [72, 73]. What unifies these efforts is the recurring finding that the right tactile *representation*—and the coverage and modality it is built on—matters at least as much as raw sensor fidelity, a design question that Chapter 4 takes up directly in simulation.

Chapter 3

Tactile Processing in the Rodent Brain

The previous chapter reviewed how biological systems transduce and process touch and how artificial systems attempt to reproduce it. This chapter *reverse-engineers* the tactile representations of a biological system that already solves the problem, using the rodent vibrissal pathway as an experimentally tractable model of somatosensory cortex (Section 2.1).

The guiding question is which neural network architectures and training objectives give rise to internal representations that match the rodent brain. We use an Encoder-Attender-Decoder (EAD) framework to systematically search task-optimized temporal networks trained on biomechanically realistic whisker force/torque sequences from a customized array simulator, and we compare their representations directly against population recordings from barrel cortex. The search points to recurrent processing and ethologically relevant, label-free self-supervision as the ingredients that best explain somatosensory cortex—the first quantitative characterization of the inductive biases of biological touch, and one that nominates these same ingredients for robust artificial tactile perception, a thread that Chapter 4 takes up on the robot side.

3.1 Introduction

Tactile perception plays an essential role in the manipulation and interpretation of complex environments through active sensing [13]. Animals effectively leverage tactile sensors, such as whiskers, to precisely navigate, forage, and identify objects even in noisy and unstructured conditions [74]; specifically, contact whiskers deliver critical sensory input for navigation, foraging, and hunting in low-light environments [75], while water-flow whiskers enable seals to discriminate prey movements from general water currents in similar conditions [76], and specialized insect hairs respond to wind stimuli, providing information essential for flight stability and agility [77].

Rodents are a common model organism for studying tactile perception in the brain due to the fine experimental control they offer and their ability to discriminate object location, shape, and texture using only their whiskers [23, 78]. These whiskers, or *vibrissae*, transmit rich mechanical signals to mechanoreceptors at their base, enabling nuanced environmental understanding without direct sensory receptors along their lengths. In fact, rodent whisking behavior is known to parallel how humans touch objects with their fingertips [28], such as in the Lateral Motion Exploratory Procedure [13]. Despite significant interest in translating such biological capabilities into robots, artificial systems still struggle to match the tactile perceptual prowess of animals, limiting their functionality in realistic, unstructured environments [79].

There are two primary reasons for this gap: one from the robotics hardware side, and the other from the neuroscience side. On the hardware side, current bio-inspired whisker sensors for robots face several critical limitations, including substantial hardware complexity when scaling sensor arrays beyond approximately 18-20 whiskers, each requiring individual transducers, significantly increasing wiring and computational demands [80, 81]. These sensors also struggle to accurately discriminate multiple simultaneous stimuli such as airflow, direct contact, and inertia, unlike biological whiskers [82]. Additionally, mechanical discrepancies from biological whiskers–such as reduced sensitivity, limited flexibility, and constrained bending angles–impede precise tactile sensing in dynamic environments [83, 84].

These limitations also apply to anthropomorphic robot hands, which remain in a relatively early stage of development. For example, besides not being able to sense temperature, good mechanical skin has remained an open challenge in haptics for approximately four decades, primarily due to difficulties in creating artificial skins that maintain realistic deformation during object contact, with current pneumatic approaches proving inadequate to retain shape [29, 30]. While visuotactile camera-based sensors such as GelSight [37] offer a limited haptic solution by providing high-resolution localized surface deformations, they fundamentally differ from human tactile sensing, which actively integrates diverse mechanoreceptor inputs over larger surfaces and multiple contacts [85]. Other issues persist with magnetic skin sensors [38, 86], which have low spatial resolution and are prone to error from electromagnetic interference [87]. These hardware limitations thus make it difficult at the moment to identify robust algorithms for tactile processing that operate on realistic sensory inputs.

On the neuroscience side, despite extensive experimental characterization of rodent somatosensory pathways (e.g. [88, 89]), the neural computations underlying precise tactile perception remain poorly understood, primarily due to a scarcity of computational models of this system. Rodent whisker sensing involves hierarchical processing, analogous to vision,

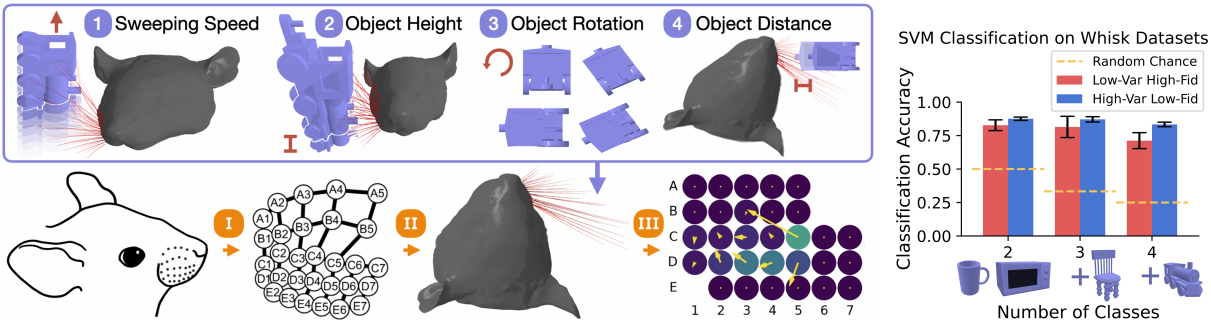
where sensory signals propagate from primary neurons in the trigeminal ganglion through parallel thalamic pathways, eventually reaching the primary and secondary somatosensory cortices (S1 and S2) [26, 27]. However, relatively few computational approaches have explicitly modeled these neural transformations [54], and none have assessed if any of these models accurately match neural population responses in somatosensory cortex.

To bridge these gaps, we build and systematically evaluate temporal neural networks explicitly trained on biomechanically-realistic force and torque tactile sequences that mice receive, customized from the first complete 3D simulation of the rat whisker array by Zweifel et al. [2]. We identify convolutional recurrent neural networks (ConvRNNs), particularly IntersectionRNNs, as superior in tactile categorization and neural alignment compared to feedforward (ResNet) and attention-based state-space models (S4 and Mamba). Leveraging supervised and contrastive self-supervised learning adapted specifically for tactile data, we demonstrate a direct linear relationship between tactile categorization performance and neural fit, saturating the currently explainable neural variability and surpassing inter-animal consistency benchmarks, with contrastive self-supervised learning serving as an equally neurally predictive, label-free proxy. Thus, we provide the *first* quantitative characterization of inductive biases required for tactile algorithms to match brain processing.

3.2 Related Work

Task-optimized ("goal-driven") neural networks are the most quantitatively accurate framework for modeling sensory cortex; we review their development across visual, auditory, language, and rodent systems in Section 2.4. Despite this breadth, the framework had been applied to tactile processing only once: Zhuang et al. [54] provided a foundational goal-driven model of the rodent whisker-trigeminal system, but explored only simple recurrent architectures and a supervised categorization loss, and never compared the resulting models against neural recordings, even though somatosensory cortex shares the hierarchical and recurrent organization that makes the framework successful in vision [25].

We extend this work in three ways. First, we greatly broaden the architectural search through our Encoder-Attender-Decoder parameterization, incorporating convolutional recurrent networks ("ConvRNNs") shown to match primate cortical dynamics [55] alongside state-space models (S4 [56], Mamba [57]) and Transformers [58]. Second, beyond supervised categorization we employ self-supervised objectives validated in mouse [51] and primate [62] visual cortex, adapted to force and torque inputs. Third, we provide the first direct comparison of all 64 models against population recordings from rodent somatosensory cortex.



(a) (b)

Figure 3.1. **ShapeNet Whisking Dataset. (a)** (I) With average mouse whisker array measurements from Bresee et al. [1], (II) objects are whisked in simulation using WHISKiT [2] resulting in (III) force and torque data for sweeping 9981 ShapeNet objects of 117 categories with various sweep augmentations. The augmentations vary the (1) speed, (2) height, (3), rotation, and (4) distance of the objects relative to the whisker array. We constructed two datasets: a large, low-fidelity set with more sweep augmentations, and a small, high-fidelity set with fewer augmentations (see appendix). **(b)** An SVM classification on up to 4 different classes of objects (cups, microwaves, chairs, and trains) for the 2 datasets show that the classes are distinguishable (above chance).

3.3 Methods

All our code for whisker simulation, model training, and neural data analysis is available on GitHub: https://github.com/neuroagents-lab/2025-tactile-whisking

3.3.1 High-Variation Whisker Dataset

Given the ongoing development of tactile sensor hardware to match the flexibility of biological tactile sensors such as fingertips and whiskers, we trained our models on synthetic data, focusing on object categorization and self-supervised learning using a high-variation dataset like ShapeNet [90]. This involves the simulated objects interacting with a biomechanically realistic rodent whisker model, WHISKiT Physics [2], the first 3D simulation of the rodent's complete whisker system, to provide high-dimensional force and torque inputs. Each of the 30–33 whiskers of an average whisker array [91, 92] is modeled as a chain of rigid conical segments interconnected by torsional springs and actuated according to established equations of motion [93].

Since our objective is to compare models with currently available mouse somatosensory data [3], we adapted this array to the 30 whiskers of the mouse [1], arranged as a 5×7 array with zero padding. Furthermore, adult mice can exhibit maximal bite forces of approximately 8-10 Newtons [94, Table 1], and since facial tolerance to applied forces would realistically be a fraction of this maximal bite force, our chosen simulation clipping range of ± 1000 milli-

Newtons (±1 N) remains comfortably within biologically plausible limits.

We created a whisking dataset for the shape categorization task, generated by passively brushing objects against the whisker array. Our primary consideration here is to generate a high-variation dataset by which to strongly constrain the representations that are learned, allowing us to more effectively simulate evolutionary pressures, in line with the "Contravariance Principle" [95] of goal-driven modeling. On each of the 9981 different objects from ShapeNet, we apply several combinations of sweep augmentations, including adjusting the object sweeping speed, height, rotation, and distance relative to the whisker array (Fig. Figure 3.1). These augmentations on our passive whisker sweeps can be considered to produce data that is isomorphic to performing active whisking in a systematic way. The 117 object categories are selected based on the work of Zhuang et al. [54] to ensure reasonably distinguishable classes. We constructed two datasets: one large dataset replicating the Zhuang et al. [54] data with 288 randomized sweep augmentations and a physics simulation step rate of 110 Hz (High-Variation/Low-Fidelity), and another with 16 sweep augmentations and a higher simulation step rate of 1000 Hz (Low-Variation/High-Fidelity), with the full augmentation procedure detailed next. Although higher temporal resolution is available from the simulation, we extract 22 timesteps for both datasets, corresponding to the average whisking frequency of 20 Hz in rodents [24].

Sweep augmentations and simulator modifications. Our whisking dataset uses the same 9981 ShapeNet objects and 117 category labels as in Zhuang et al. [54], but using an improved whisker model and various sweep augmentations, which are listed in Table Table 3.1.

	Sim. Freq.	Speed	Height	Rotation	Distance	Size	Total Sweeps
(1)	1000 Hz	30 mm/s	-5, 0 mm	0, 30, 90, 120°	5, 8 mm	40 mm	316,192
(2)	110 Hz	[30∼60]	-3, 0, 3	[0∼359]	5	[20∼60]	2,076,048

Table 3.1. **Sweep Augmentations** used for the two whisking datasets, which we refer to as (1) Low-Variation High-Fidelity and (2) High-Variation Low-Fidelity. Simulation Frequency refers to the frequency corresponding to the physics timestep used in Bullet physics engine [12], the backend for WHISKiT simulator [2]. For dataset (2), the speed, rotation, and size is each sampled from the range 26 times. The total number of sweeps equals the number of sweep augmentation combinations × 9981 objects.

WHISKiT Simulator Modifications. We make a few enhancements to the original WHISKiT simulator [2]:

- Number of whisker links (where whiskers are modeled as a chain of springs [54]) are dynamically adjusted by the length of the whisker instead of a fixed number.
- Allow for loading objects with or without convex hull. In the whisk datasets, objects are loaded with convex hull (using V-HACD [96]) to keep collisions more stable. In generating the 6 simulated stimuli for model input neural evaluation, convex hull was not used in order to preserve the concavity of the "concave" object. The object was geometrically simple enough to not affect collision stability.
- Add camera settings for viewing in orthographic or perspective projection.
- Load multiple mice/rats at a time.

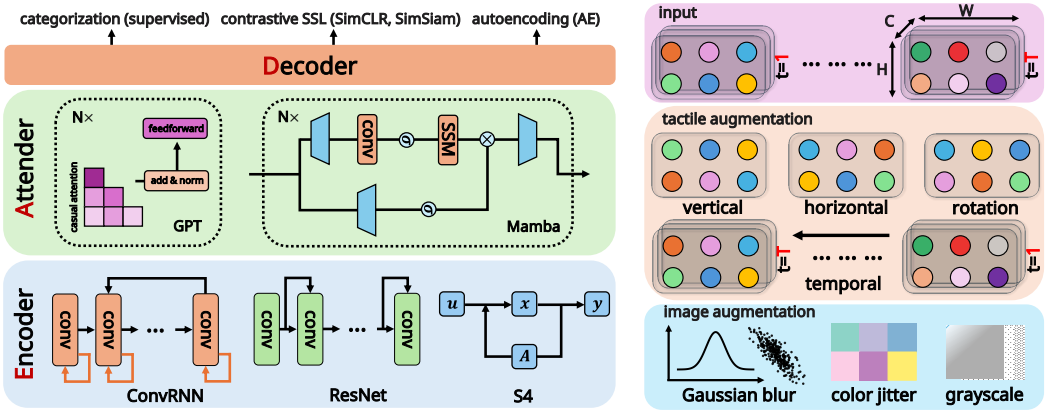
3.3.2 Encoder-Attender-Decoder Model Search

Tactile recognition is performed by the somatosensory cortex, which, though less understood than vision, exhibits hierarchical processing and long-range feedback connections [13, 79]. These basic anatomical insights motivate our exploration of hierarchical, temporal neural network (TNN) models such as ConvRNNs shown previously to match primate visual cortex dynamics [55, 97], SSMs [56], Transformers [58], and ResNets [98] as a feedforward control. We provide our library `PyTorchTNN` which enables large-scale exploration of integrating multiple TNN modules that are also combined by long-range feedback with feedforward connections.

These considerations produce a rather large search space of model architectures. Therefore, to be able to effectively search the space of TNN models systematically, we came up with an Encoder-Attender-Decoder (EAD) parameterization, schematized in Fig. Figure 3.2(a). Convolutional recurrent and state-space layers (such as ConvRNNs and S4) are particularly effective for encoding smooth and compressible temporal signals, given their recurrent smoothing properties and linear-time complexity. In contrast, transformers and other attention-like architectures (e.g., Mamba) excel at processing temporally sparse or irregularly informative inputs, as they independently weight each timestep. Given that tactile inputs from force and torque sensors provide temporally smooth signals, encoder **(E)** layers closer to these inputs naturally benefit from convolutional and recurrent mechanisms that integrate information locally over time, effectively reducing redundancy and noise. In contrast, higher-level temporal aggregation **(A)**, which must selectively integrate meaningful signals across longer and potentially irregular timescales, is better served by attention-based architectures like Transformers or Mamba, which can dynamically weigh distinct timesteps based on their informational content. Finally, the decoder **(D)** is either the classification layer for supervised learning, or similarly producing self-supervised features for contrastive

learning or autoencoding. The EAD paradigm flexibly combines these complementary modules, enabling an effective and structured exploration of the temporal model space for tactile processing.

Specifically, we explore the following EAD combinations (visualized in Fig. Figure 3.2(a)): 1) Encoder: Zhuang's recurrent model [54], ResNet [98], S4 [56]; 2) Attender: Transformer [58] (e.g., GPT), Mamba [57], None (i.e., attender-free); 3) Decoder: MLP. We further investigate different variants of Zhuang's model by replacing the recurrent layers in the ConvRNNs with UGRNN [99], IntersectionRNN [99], LSTM [100], and GRU [101], whose update rules we give below. In the `PyTorchTNN` library, each time step in ConvRNNs corresponds to a single feedforward layer processing its input and passing it to the next layer, in contrast to treating each entire feedforward pass from input to output as a single integral time step, as is normally done with RNNs [102]. This implementation of temporal unrolling parallels biological systems, where stimuli are sequentially processed from one cortical layer to the next.



(a) (b)

Figure 3.2. **Architecture and data augmentations. (a)** **E**ncoder-**A**ttender-**D**ecoder (**EAD**) architecture, with task objectives being supervised categorization, self-supervised learning (SimCLR, SimSiam, autoencoding). The ConvRNN encoder includes **self-recurrence** at each layer where we vary different RNNs.
(b) Types of data augmentations applied to SSL models. Given a temporal **tactile input** over time T, our **tactile augmentation** vertically, horizontally, temporally flips, and rotates the features, while traditional **image augmentation** introduces Gaussian noise, color jitter, and grayscale.

Recurrent cell variants. The recurrent layers of the ConvRNN encoder can be realized with several gated cells. We detail the two that anchor our results—the UGRNN and the IntersectionRNN that builds upon it—and refer to Nayebi et al. [97] for the LSTM and GRU variants. Throughout, x_t^ℓ and s_t^ℓ denote the input and recurrent state at time t and layer ℓ;

W^ℓ, U^ℓ, and b^ℓ are the learnable weight matrices and biases; $\circ$ is element-wise multiplication; $*$ is a linear transformation or convolution, depending on context; σ, tanh, and ReLU are the usual nonlinearities; and h_t^ℓ is the layer output.

The Update-Gate RNN (UGRNN) [99] is a minimal gated cell that, at each step, interpolates between carrying its previous state forward and overwriting it with a fresh candidate. It first computes a candidate state c_t^ℓ and a single update gate g_t^ℓ,

$$c_t^\ell = \tanh\big(W_c^\ell * x_t^\ell + U_c^\ell * s_{t-1}^\ell + b_c^\ell\big), \qquad g_t^\ell = \sigma\big(W_g^\ell * x_t^\ell + U_g^\ell * s_{t-1}^\ell + b_g^\ell + 1\big), \tag{3.1}$$

where the $+1$ offset inside the gate biases the cell toward retaining its state early in training. The gate then forms a convex combination of the old state and the candidate, $s_t^\ell = g_t^\ell \circ s_{t-1}^\ell + (1 - g_t^\ell) \circ c_t^\ell$, which is emitted directly as the layer output, $h_t^\ell = s_t^\ell$.

The IntersectionRNN [99]—our best-performing recurrent cell—extends this design with a second, symmetric gating pathway that modulates the layer's *output* as well as its recurrent state, easing the flow of information up the stack of ConvRNN layers. Alongside a tanh candidate state m_t^ℓ it forms a ReLU candidate output n_t^ℓ,

$$m_t^\ell = \tanh\big(W_m^\ell * x_t^\ell + U_m^\ell * s_{t-1}^\ell + b_m^\ell\big), \qquad n_t^\ell = \mathrm{ReLU}\big(W_n^\ell * x_t^\ell + U_n^\ell * s_{t-1}^\ell + b_n^\ell\big), \tag{3.2}$$

together with a state gate p_t^ℓ and an output gate y_t^ℓ (each again using the $+1$ retention bias),

$$p_t^\ell = \sigma\big(W_p^\ell * x_t^\ell + U_p^\ell * s_{t-1}^\ell + b_p^\ell + 1\big), \qquad y_t^\ell = \sigma\big(W_y^\ell * x_t^\ell + U_y^\ell * s_{t-1}^\ell + b_y^\ell + 1\big). \tag{3.3}$$

The state gate updates the recurrent state exactly as in the UGRNN, $s_t^\ell = p_t^\ell \circ s_{t-1}^\ell + (1 - p_t^\ell) \circ m_t^\ell$, while the output gate interpolates between passing the input straight through and emitting the new candidate, $h_t^\ell = y_t^\ell \circ x_t^\ell + (1 - y_t^\ell) \circ n_t^\ell$.

3.3.3 Model Training and Tactile Augmentations

Besides the supervised categorization objective, we also consider self-supervised learning (SSL) losses: SimCLR [59], which learns robust representations via distinguishing the embeddings of augmentations of one image from other images, SimSiam [60], which maximizes the similarity between the embeddings of two augmentations of the same image, and autoencoding (AE) [61], where we use a 3-layer deconvolution network to reconstruct an image from its sparse latent representation.

We use a train/validation/test split of 80/10/10%, and report the top-5 test accuracy. For supervised learning, we train models for 100 epochs and test on the checkpoint saved

with the highest validation accuracy. For SSL objectives, we train for 100 epochs and save the model with the lowest validation loss, then continue training for another 100 epochs with the checkpoint frozen and a learnable linear layer on top of it. To stabilize training, we also consider adding layer norm [103] to the ConvRNNs as an alternative when training was unstable. All experiments are conducted on NVIDIA A6000 GPUs, and one set of model search experiment takes 8∼16 hours; the full optimizer, scheduler, and architecture configurations are given below.

In addition to the various *sweep augmentations* used to generate the initial dataset, we apply tactile augmentations at training time as a cheap way to generate more training data. We illustrate the tactile input, our tactile augmentations, and traditional image augmentations for SSL in Fig. Figure 3.2(b). Given a temporal tactile input over time T, the proposed augmentation either vertically flips, horizontally flips, rotates, or reverses the input over time. Although the variable whisker lengths mean that these operations are not completely physically accurate, our tactile augmentations still meaningfully mimic flipping/rotating the whisker array or reversing the direction of motion. We considered temporal masking as an additional augmentation, but found that this did not significantly improve task performance (Section 3.4.1, Table Table 3.7). Applying traditional image augmentations like color jitter or gray scale cause the models to fail to train, which provides evidence that the choice of augmentations should represent operations relevant to the given modality.

Training configurations. For supervised learning, we use a batch size of 256 for all the models and train for 100 epochs. For SSL, during the pre-training stage, we use a batch size of 256 for SimCLR and autoencoding (AE), and a batch size of 1024 for SimSiam, following [51], and train for 100 epochs. During the linear probing stage, we freeze the checkpoint saved with the lowest validation loss and add a trainable linear layer. We further train such a model with labels for 100 epochs, with a batch size of 256, an initial learning rate of 0.1, with the StepLR scheduler, and the SGD optimizer with momentum [104].

We detail the optimizers [104–107] and schedulers, and their configurations we used in supervised learning and SSL in Table Table 3.2.

For supervised learning, we present the model training configurations in Table Table 3.3, where we detail the choices of optimizer, scheduler, learning rate, and the encoder and attender of our EAD architecture; we omit the decoder, as it is always a linear layer/MLP. As Zhuang+GPT is the best performing supervised model in terms of classification accuracy, we explore different variants of Zhuang's encoder with attender being GPT.

For SSL, we follow [51] and use specific configurations of optimizer and scheduler for different losses, which are shown in Table Table 3.4. For SimSiam, we try the CosineAnnealing scheduler both with and without 10 epochs of warmup to search for better model

Optimizer	Configuration
SGD	`momentum` $= 0.9$, `weight-decay` $= 10^{-4}$
Adam	`weight-decay` $= 10^{-4}$
AdamW	`weight-decay` $= 10^{-4}$
LARS	`momentum` $= 0.9$, `weight-decay` $= 10^{-4}$
Scheduler	**Configuration**
StepLR	`step_size` $= 30$
ConstantLR	fixed learning rate
CosineLR	`warmup_epoch` $= 10$
CosineAnnealing	`min_lr` $= 0.0$, `warmup_epoch` $= 10$, `warmup_ratio` $= 10^{-4}$

Table 3.2. **Optimizers and Schedulers** with default configurations of supervised learning and SSL when training different model architectures.

Encoder	Attender	Optimizer	Scheduler	Learning Rate
ResNet	None	SGD	StepLR	$10^{\{-1,-2,-3\}}$
Zhuang	None	SGD	StepLR	$10^{\{-1,-2,-3,-4\}}$
		SGD	ConstantLR	$10^{-2}, 5 \times 10^{-3}$
Zhuang-{UGRNN, IntersectionRNN, GRU, LSTM}	None	{SGD, Adam, AdamW}	StepLR	$10^{\{-1,-2,-3,-4\}}$
		(LayerNorm) - AdamW	StepLR	$10^{\{-1,-2,-3,-4\}}$
ResNet, Zhuang, Zhuang-{UGRNN IntersectionRNN-LN, GRU, LSTM}, S4	GPT	AdamW	CosineLR	10^{-4}
		{SGD, AdamW}	StepLR	$10^{\{-1,-2,-3\}}$
ResNet, Zhuang, S4	Mamba	AdamW	CosineLR	10^{-4}
		{SGD, AdamW}	StepLR	$10^{\{-1,-2,-3\}}$

Table 3.3. **Model Training Configurations for Supervised Learning**. We use {} to indicate different choices of a specific component (i.e., encoder, optimizer, learning rate) in the search space. We consider adding layer norm as a variant when searching the best configuration for Zhuang's variants (i.e., the second row), which is denoted as "-LN".

performance. For AE, we use a 3-layer deconvolution network to decode the sparse latent representation from our EAD framework into the original tactile input, regardless of the choice of model architectures.

We present the model architectures explored for SSL in Table Table 3.5, where we present the encoder and attender of our EAD architecture, as during the pre-training stage, only the encoder and attender are used, and during the linear probing stage, the decoder is always a one-layer linear classification head.

Loss	Optimizer	Scheduler	Learning Rate
SimCLR	LARS	CosineAnnealing	$10^{\{-1,-2,-3,-4\}}$
SimSiam	SGD	CosineAnnealing (with & w/o warmup)	$10^{\{-1,-2,-3,-4\}}$
AE	SGD	StepLR	$10^{\{-1,-2,-3,-4\}}$

Table 3.4. **Optimizer, Scheduler, and Learning Rate Configurations for SSL** when training different model architectures.

Encoder	Attender
Zhuang, Zhuang-{UGRNN, IntersectionRNN-LN, GRU, LSTM}	None
ResNet, Zhuang-IntersectionRNN-LN, Zhuang, S4	GPT
ResNet, Zhuang, S4	Mamba

Table 3.5. **Model Architecture Configurations for SSL**. We use {} to indicate the different variants of encoders. We consider adding layer norm (LN) as a variant when searching the best configuration for Zhuang's variants.

3.3.4 Neural Data and Alignment

We use the neural dataset from Rodgers [3], which recorded neural population activity across superficial (L2/3) to deep (L6) layers of the barrel cortex as the mice used their whiskers to perform a simple 2-object (convex/concave) classification task. Note that this low number of distinct stimuli is a critical limitation which we address in the Discussion section. After filtering for valid trials, the neural data we used contains a total of 999 neural units across 11 mice. The time window of each trial is clipped to the time of first whisker contact with the object until the time the mouse makes its decision (about 1-2 seconds). The neural response is binned into intervals of approximately 45–50 ms, corresponding neatly with the 15–20 Hz whisking frequency typical of rodents (50 ms per cycle at 20 Hz [24]), and aligning specifically with our model's integration scheme of five sub-timesteps (~9 ms each) per timestep. This choice provides a temporal resolution optimal for capturing detailed neural activity during whisker-object contact periods.

We evaluate neural alignment using Representational Similarity Analysis (RSA), a parameter-free approach that compares the pairwise dissimilarity structure of model activations and neural population responses across a common set of stimuli, allowing direct comparison of representational geometry without fitting any additional parameters [108].

To ensure only reliable neurons are included, we compute split-half internal consistency of the neural responses across trials and retain only neurons with a Spearman-Brown–corrected reliability greater than 0.5. All RSA evaluations are thus conducted on this filtered subset of self-consistent neurons, and corrected by this internal consistency. The full noise-correction and inter-animal consistency procedure is detailed below.

The noise-corrected RSA Pearson's r score is also computed between one mouse and every other mouse, and then averaged across all mice to obtain the mean animal score which will serve as the baseline for model-brain evaluation, to account for inherent animal-to-animal variability (which we denote as "a2a" in the barplots). Thus, we want our models to match the brain, at least as well as animals do to each other.

Next, we replicated the experimental setup of the barrel cortex dataset Rodgers [3] *in silico*. Fig. Figure 3.7(a) shows the 6 different stimuli used in the experiment, which are concave/convex objects at three different distances (near, medium, far) from the whisker array. With 3D model reproduction of the concave/convex objects in the experiment, active whisking is simulated under the 6 stimuli using a real recorded whisking trajectory [1] to generate the model input.

We then obtain the neural alignment score for models by computing the maximum, across all layers, of the median RSA score between each layer's representation and the neural responses, averaged across mice. The standard error is calculated across the RSA Pearson's r value from the model to each mouse.

Noise correction and inter-animal consistency. We use the NeuroAI Turing Test [109] to quantify the neural similarity of mice whisking to models performing tactile categorization. RSA is computed over stimuli and neurons, where the average is over source animals/subsampled source neurons, bootstrapped trials, and train/test splits.

For the neurons of animal A to animal B in the set of animals $\mathcal{A}$ we estimate the RSA correlation:

$$\left\langle \mathsf{RSA}\left(\mathrm{t}^{\mathsf{A}}, \mathrm{t}^{\mathsf{B}}\right)\right\rangle_{\mathsf{A}\in\mathcal{A}:(\mathsf{A},\mathsf{B})\in\mathcal{A}\times\mathcal{A}} \sim \left\langle \frac{\mathsf{RSA}\left(\mathrm{s}_1^{\mathsf{A}}, \mathrm{s}_2^{\mathsf{B}}\right)}{\sqrt{\widetilde{\mathsf{RSA}}\left(\mathrm{s}_1^{\mathsf{A}}, \mathrm{s}_2^{\mathsf{A}}\right) \times \widetilde{\mathsf{RSA}}\left(\mathrm{s}_1^{\mathsf{B}}, \mathrm{s}_2^{\mathsf{B}}\right)}} \right\rangle_{\mathsf{A}\in\mathcal{A}:(\mathsf{A},\mathsf{B})\in\mathcal{A}\times\mathcal{A}} . \tag{3.4}$$

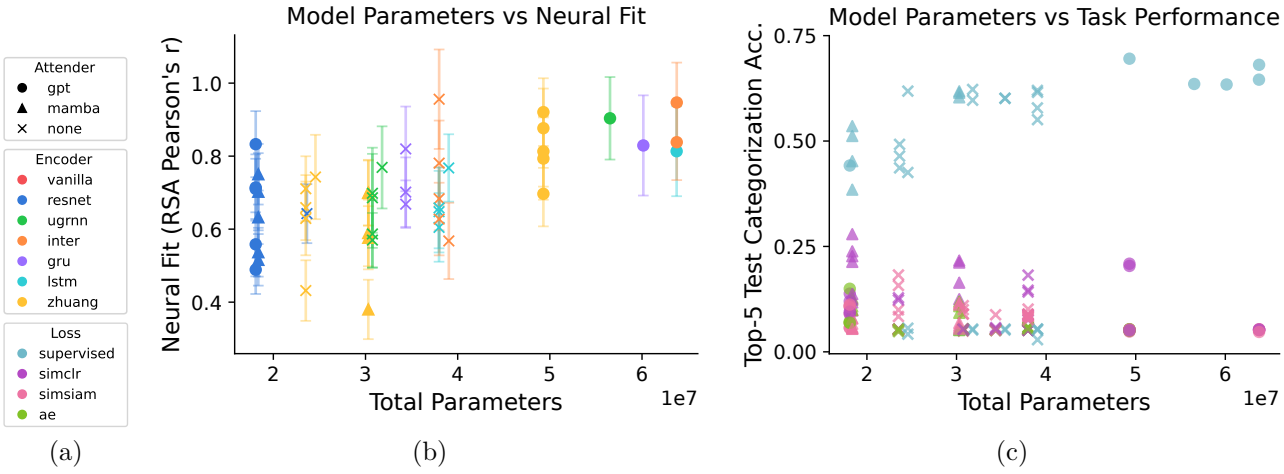
Each neuron in our analysis is associated with a value for when it was a target animal (B), averaged over subsampled source neurons and 1000 bootstrapped trials. This yields a vector of these average values, which we can take median and standard error of the mean (s.e.m.) over, as we do with standard explained variance metrics.

Spearman-Brown Correction. The Spearman-Brown correction can be applied to each of the terms in the denominator individually, as they are each correlations of observa-

tions from half the trials of the *same* underlying process to itself (unlike the numerator).

$$\widetilde{\mathsf{RSA}}\,(X, Y) \coloneqq \widetilde{\mathsf{Corr}}(\mathsf{RDM}(x), \mathsf{RDM}(y)) = \frac{2\,\mathsf{RSA}\,(X, Y)}{1 + \mathsf{RSA}\,(X, Y)}.$$

Inter-Animal Consistency. To estimate the inter-animal consistency, we evaluate the pooled animal consistency for each animal. One animal is held out at a time, then compared to the pseudo-population aggregated across units from the remaining animals. We found the mean pooled animal score was 0.175 with a s.e.m. of 0.161 and maximum score of 1.34.



(a) (b) (c)

Figure 3.3. **Model Parameters** compared with categorization performance and neural fit. **(a)** Legends for (b) (Encoder colors) and (c) (Loss colors). **(b)** Models with GPT as an attender have a higher correlation score slightly higher than those without, but high neural fit is still achievable without more parameters as demonstrated by the Inter+SimCLR model (high green "×"). **(c)** Models with GPT as the attender has more parameters and, when trained with supervised learning, tends to have higher top-5 categorization accuracy.

3.4 Results

Validation of Sensor and Dataset via SVM Decoding. We first verified the basic discriminability of the whisker array dataset using support vector machine (SVM) classifiers on the High-Variation/Low-Fidelity and Low-Variation/High-Fidelity datasets (Fig. Figure 3.1(b)). Both showed strong decoding performance, exceeding chance and remaining robust even with increased task difficulty from more object categories. The comparable results across sampling rates confirm that the smaller Low-Var./High-Fid. dataset retains

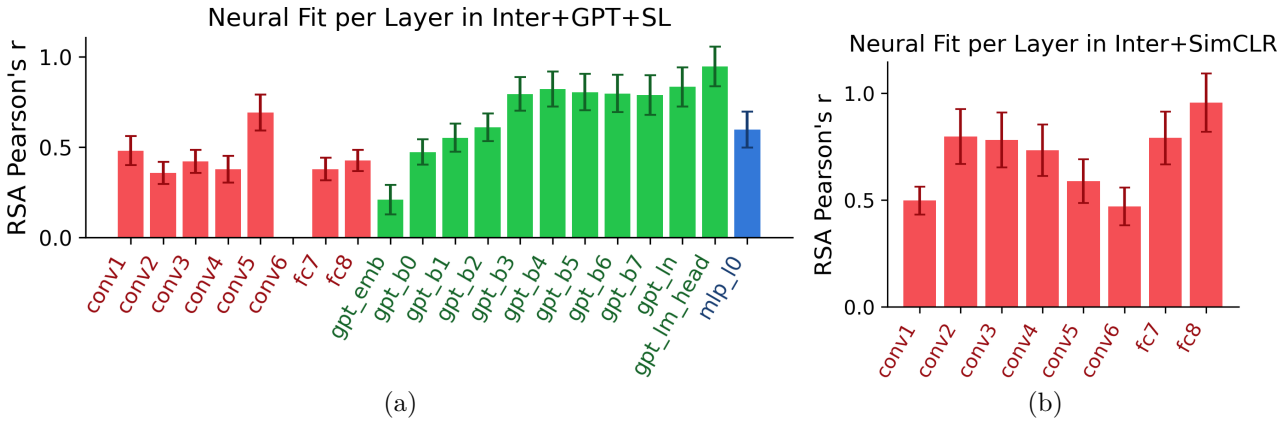


Figure 3.4. **Neural Fit Score per Model Layers** colored by encoder, attender, and decoder. No bar means the score is NaN for that layer. **(a)** Inter+GPT+SupervisedLearning is the model that scored the highest neural fit out of the supervised models. We observe that later GPT layers perform increasingly better. **(b)** The last fully-connected layer of Inter+SimCLR achieved the highest r value.

sufficient discriminative tactile information, allowing for efficient model training without compromising performance.

ConvRNN Encoders Outperform Feedforward and Attention-based Architectures for Tactile Categorization. We next systematically evaluated the task performance of the EAD architectures trained on tactile force/torque sequences. Overall, we found that across EAD architectures, the choice of encoder (E) was quite important for tactile recognition. Specifically, ConvRNN encoders, especially the IntersectionRNN [99], surpassed the purely feedforward ResNet18 and SSM (S4) encoders, in supervised tactile categorization tasks (Fig. Figure 3.6). Additionally, models trained with our custom force-and-torque-specific contrastive self-supervised learning (SSL) augmentations (Fig. Figure 3.2b) outperformed untrained networks of the same architecture, and those trained with standard image-based augmentations that involve Gaussian blur and color jittering [59] did not train with the best architecture despite hyperparameter tuning, demonstrating the importance of tailored tactile augmentations for enhancing task performance (Fig. Figure 3.6).

Importance of Architectural Inductive Bias. We note that the raw model input (the tactile force/torque sequences obtained from actively whisking on the 6 stimuli objects) achieves a correlation of 0.46 (Fig. Figure 3.7(b)), and untrained randomly initialized models also achieve moderately positive correlation to the mouse neural data. Randomly initialized networks are highly non-random functions as they are architectures selected to perform the task well when trained. Therefore, architecture matters a lot, and this is a well-noted phenomenon in NeuroAI across brain regions [47, 51, 110]. By doing this comparison, it

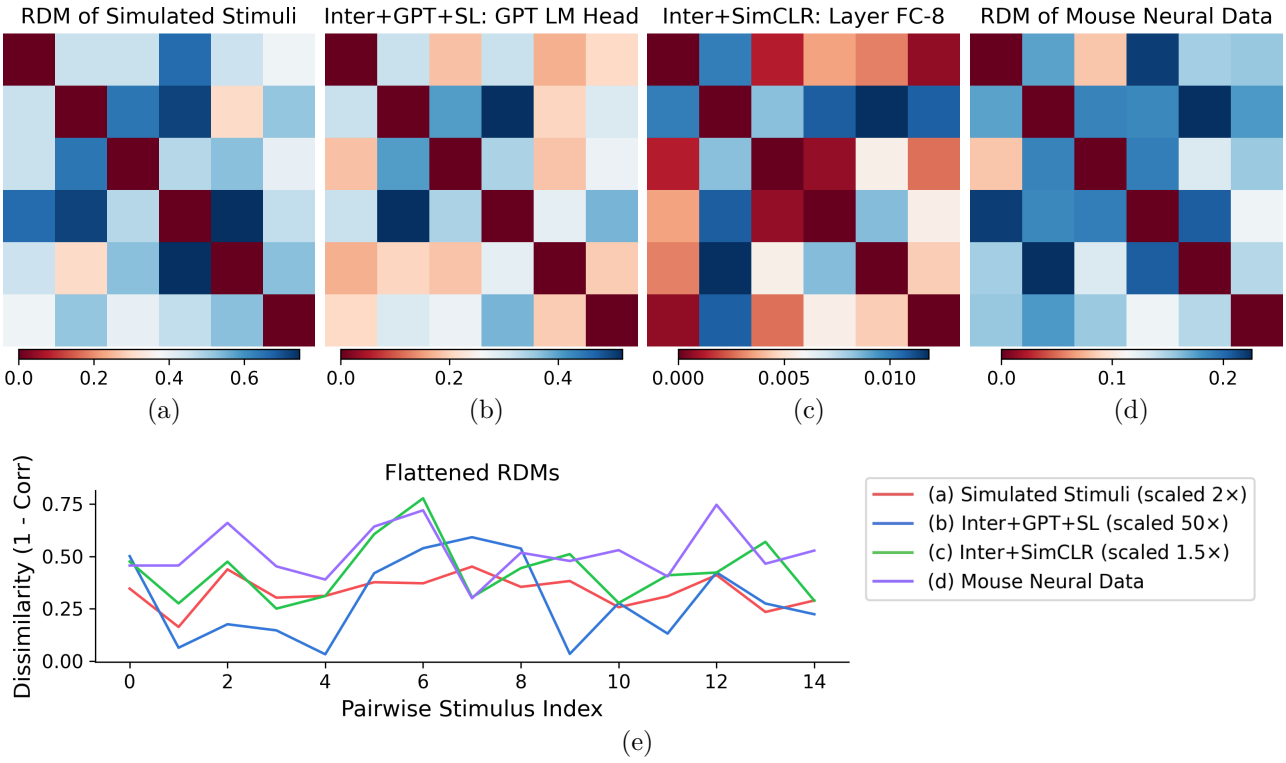


Figure 3.5. **Representational Dissimilarity Matrices** (RDM) for tactile data and neural evaluation. **(a)** RDM of the 6 simulated stimuli which is used as the model input in neural evaluation. **(b)** The GPT LM Head layer of Inter+GPT+SupervisedLearning is the supervised model with the highest neural fit score. **(c)** The last Fully-Connected layer in Inter+SimCLR achieves the highest neural fit score out of SSL models. **(d)** RDM performed on the 6 stimuli in the mice neural data. **(e)** A visualization of the flattened RDMs, scaled approximately to fit (RSA correlation does not take scale into account).

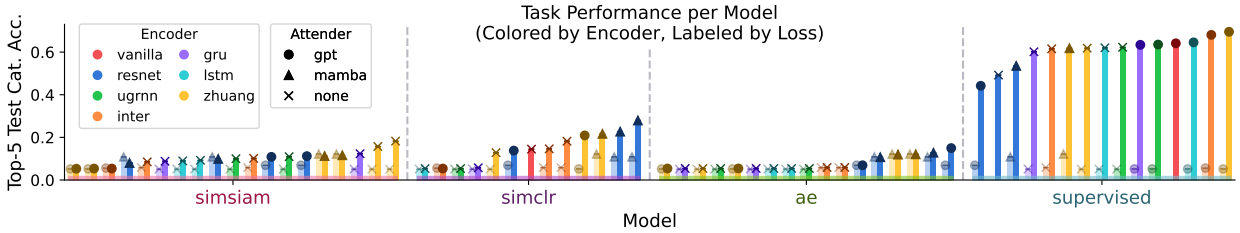
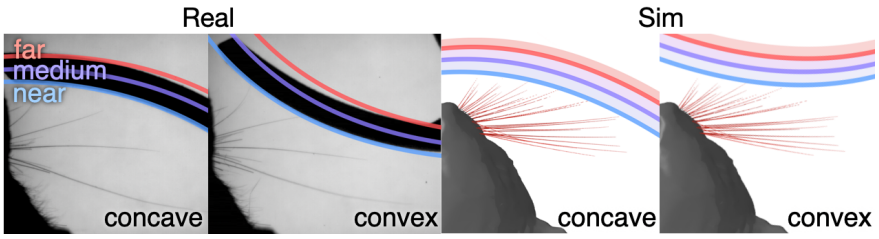
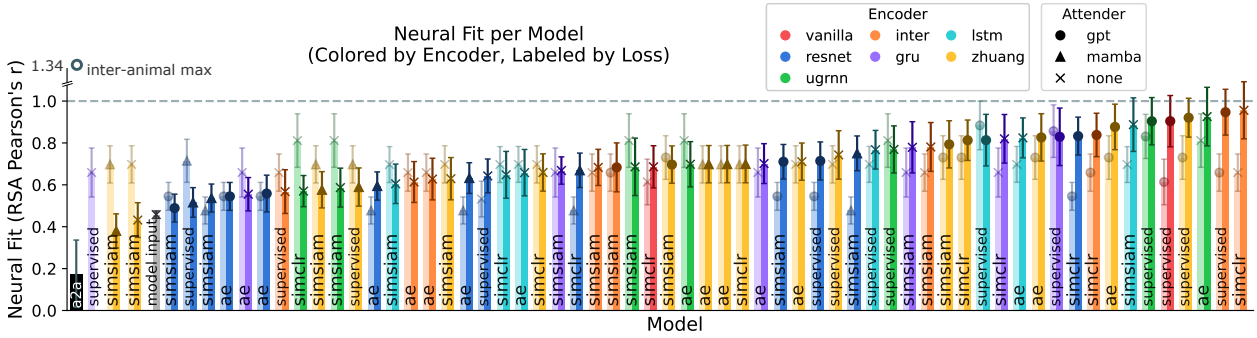


Figure 3.6. **Tactile Categorization Accuracy.** The lighter-colored left bar represents the randomly initialized version for every model. The best-performing model is Zhuang+GPT+Supervised (rightmost yellow bar). Models with the encoder being S4 are excluded as the training losses explode before the first epoch is finished.



(a)



(b)

Figure 3.7. **Model Neural Evaluation. (a)** We use six different stimuli (concave/convex × near/medium/far) replicating the conditions in the mouse neural dataset in simulation. (Real images were taken from video recordings in neural dataset [3]). **(b)** Comparison of neural fit (noise-corrected RSA Pearson's r) across models. The mean animal-to-animal score is 0.18 and the maximum between all pairs of animals is 1.34. The leftmost "a2a" bar represents the *mean* animal-to-animal neural consistency score. The lighter-colored left bar represents the randomly initialized version for every model.

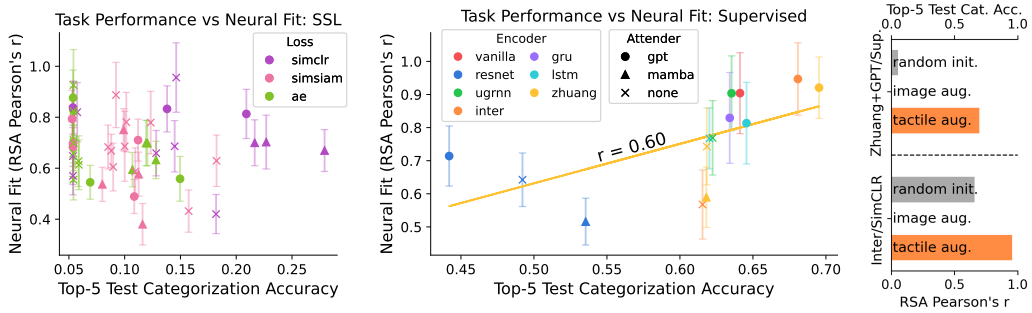
allows us to isolate the contributions of the architecture vs. loss function for every pair of (architecture, loss function) tuples. The models that saturate our noise ceiling (bars on far right in Fig. Figure 3.7(b)) are noticeably improved when trained.

ConvRNN Encoders Saturate Explainable Neural Variance in Rodent Somatosensory Cortex. To assess biological realism, we compared internal model representations to neural recordings from rodent whisker somatosensory cortex. All trained EAD models outperformed raw sensor inputs in neural alignment, underscoring the importance of nonlinear temporal processing in modeling brain-like tactile representations (Fig. Figure 3.7(b)). In fact, the best models saturated *all* of the held-out explainable neural variance–*without* fitting any additional parameters–even when tested on entirely novel objects under substantially different experimental conditions (active whisker sensing rather than passive contact, as used in training). Remarkably, the neural predictivity exceeded the average inter-animal neural consistency (leftmost black "a2a" bar in Fig. Figure 3.7(b)), thereby robustly passing the NeuroAI Turing Test on this dataset [109]. Among these and consistent with the categorization results, EAD models with ConvRNN encoders consistently provided

better neural fits compared to feedforward (ResNet) and SSM-based (S4) encoder architectures, underscoring the biological plausibility of recurrence in modeling tactile processing. In fact, we saw a strong linear trend between tactile supervised categorization performance and neural fit ($r = 0.59$, Fig. Figure 3.8(b)), with the model layers that best predict the tactile neural responses being closest to the decoder layer (Fig. Figure 3.4). Although the number of task-optimized parameters matters for both categorization test set performance and neural fit generalization (Fig. Figure 3.3), they are not the whole story, as the SimCLR-trained EAD with the IntersectionRNN encoder best matches the neural data with far fewer parameters ($\sim 3.80 \times 10^7$ parameters) than its supervised counterpart with a GPT-based attender ($\sim 6.38 \times 10^7$ parameters).

GPT-based Attenders Provide Modest Improvements in Task Performance and Neural Alignment. In addition to varying the encoder (E) layer of the EADs, we investigated different temporal aggregation ("Attender") schemas. We observed that GPT-based Attenders modestly outperformed Mamba-based and no-attention controls in both supervised task categorization and neural alignment (Fig. Figure 3.8(b)). Although these improvements were subtle, the consistency of the result suggests that incorporating some form of attention downstream of the ConvRNN encoder is beneficial, particularly for our highest-performing neural models. This result suggests the prediction that attention-like mechanisms may be implemented in somatosensory cortex through selective modulation of hierarchical processing pathways, such as those from primary sensory neurons in the trigeminal ganglion through the thalamus and subsequently into primary and secondary somatosensory cortical areas (S1 and S2), which could be validated by experiments involving targeted perturbations or optogenetic manipulation of these specific pathways during tactile discrimination tasks.

Self-Supervised SimCLR Training Matches Supervised Models and Serves as an Ethologically-Relevant Label-Free Proxy. We compared neural alignment achieved by supervised training against SSL methods, adapted with tactile-specific force-and-torque augmentations. This comparison is necessary because discriminating among 117 human-recognizable shape classes is not directly ethologically relevant for rodents. Nevertheless, ShapeNet's extensive diversity of 3D objects provides strong structural constraints for modeling whisker-trigeminal processing at biological scales, constraints which smaller, mouse-relevant object sets might fail to impose. Paralleling findings in vision [47, 48], where training on large-scale object categorization leads to generalizable representations beyond specific categories, we similarly suggest that ShapeNet's diversity provides meaningful constraints on network structure irrespective of exact object identity. Additionally, by demonstrating that self-supervised learning (SSL) methods yield neural representations comparable (if not



(a) (b) (c)

Figure 3.8. **Comparing Task Performance and Neural Fit. (a)** The task performance of SSL models are about one order of magnitude below the performance of supervised models, yet are able to achieve comparable neural fit. **(b)** For supervised models, we observe a trend of better task performance leading to increased neural correspondence. Plotting a best fit line, we find the correlation $r = 0.59$. **(c)** The tactile augmentations were effective in improving both the neural fit and task performance. The models were unable to be trained with image augmentations.

marginally better) to supervised approaches, we directly address an open direction highlighted by Zhuang et al. [54], who emphasized the need for developing more ethologically relevant yet practically scalable tasks for studying tactile processing.

In fact, we found specifically that *contrastive* SSL loss functions consistently reached neural alignment scores similar to the best supervised models, confirming its utility as an ethologically-relevant but label-free approach to tactile representation learning (Fig. Figure 3.8(a)), and outperforming other *non-contrastive* self-supervised approaches such as autoencoding. The SimCLR-optimized IntersectionRNN EAD achieved the best absolute neural alignment score across all models overall, and was comparable in its neural alignment to its supervised (and more parameter-dense) variant (rightmost green bars in Fig. Figure 3.7(b)). This result mirrors findings in primate visual cortex [62], where contrastive SSL methods equally predicted neural response patterns compared to supervised alternatives, suggesting supervised learning serves as a proxy for this more ethologically-relevant loss function. Interestingly, the parity between contrastive SSL and supervised models observed here differs from mouse visual cortex, where contrastive SSL methods substantially exceeded supervised methods in neural predictivity [51]. Notably, the SSL methods, which are best aligned with tactile neural representations overall, achieve only moderate linear probe categorization performance (Fig. Figure 3.8(a)), implying that somatosensory cortex representations might prioritize broader, task-agnostic sensory encoding rather than purely specialized, categorization-driven features. Furthermore, just as we found for downstream

task performance in Fig. Figure 3.6, we also observed it was critical to have tactile-specific SSL augmentations for developing biologically accurate tactile representations, compared to both standard image-based SimCLR augmentations and randomly initialized architecture-fixed controls (Fig. Figure 3.8(c)). We further provide evidence for the design of our tactile augmentation in Section 3.4.1.

3.4.1 Additional Experiments

As a basic way to probe the distinguishability of the models' learned representations of stimuli, we performed a decoding test with results shown in Table Table 3.6. Note that Inter+SimCLR is the model with the best neural score; Zhuang+GPT+Supervised has the best task score; Inter+GPT+Supervised has best neural score out of supervised models; Resnet+Mamba+SimCLR has best task score out of SSL models.

Rodgers' task variables (convex/concave) are behavioral probes, and are not necessarily the natural computational goals of somatosensory cortex. Our RSA analysis shows that cortical representational geometry is best matched by tactile-optimized models, especially with self-supervised losses (Fig. Figure 3.5), indicating that the cortex builds a general substrate for tactile recognition and discrimination across diverse objects. When reduced to a six-condition probe, this broad representational space may not project well onto the labels, yielding poor decoding despite underlying ethologically aligned representations.

Model	**Shape** (chance=0.5)	**Distance** (chance=0.33)
Inter+SimCLR	0	0
Zhuang+GPT+Supervised	0.33	0.5
Inter+GPT+Supervised	0	0.5
Resnet+Mamba+SimCLR	0	0.67
Animal Neural Data	0.44	0

Table 3.6. **Stimuli decoding**. For each of the 6 stimuli, we train a logistic regression model on the 5 other stimuli and test on the 1 held-out stimulus to decode either the shape (convex/concave) or the distance (far/medium/near). The stimuli decoding score for Animal Neural Data was obtained by decoding per mouse, then averaging across mice.

We have also tried adding temporal masking, where we randomly mask 75% of the input timesteps as a tactile augmentation, but it did not significantly improve the task performance or neural fit score (Table Table 3.7).

Model	Top-5 Cat. Accuracy	Neural Fit
Inter+SimCLR	0.15	0.96
Inter+SimCLR **with Temporal Masking**	0.18	0.42
Resnet+Mamba+SimCLR	0.23	0.70
Resnet+Mamba+SimCLR **with T. Masking**	0.28	0.67

Table 3.7. **Temporal Masking**. Results from retraining two models with temporal masking as an additional tactile augmentation. Top-5 categorization task performance improved slightly (~5%), but neural fit scores decreased.

3.5 Discussion

Implications for Somatosensory Cortical Processing. We developed a novel Encoder-Attender-Decoder (EAD) parameterization of the space of temporal neural network models (TNN) trained to perform tactile recognition on biomechanically realistic force and torque sequences, greatly extending and answering the open question posed by Zhuang et al. [54] of characterizing the ethologically relevant constraints of rodent whisker-based tactile computations. Our results establish ConvRNN encoders, particularly the IntersectionRNN, as currently superior in tactile categorization performance and neural representational alignment compared to feedforward (ResNet) and state-space models (SSM), suggesting recurrent processing is more relevant overall to the rodent somatosensory system. Furthermore, we demonstrate that contrastive self-supervised learning (SimCLR), particularly when trained with tactile-specific augmentations, yields neural alignments comparable to supervised methods, highlighting supervised training as a proxy for a more ethologically relevant, label-free representation. The modest yet consistent benefits observed from GPT-based Attenders indicate potential attention-like mechanisms operating within hierarchical tactile processing pathways, suggesting a fruitful avenue for experimental validation.

Taken together, our findings indicate that nonlinear recurrent processing plays an essential role in the rodent somatosensory cortex, reflecting neural encoding strategies that prioritize broad, general-purpose tactile representations. This work provides the first quantitative characterization of the inductive biases required for tactile algorithms to match brain processing, opening new opportunities for deeper insights into sensory representation learning and somatosensory cortical dynamics.

Implications for Embodied AI and Robotics. Current artificial tactile sensors and models fall short of animal-like capabilities, limiting the functional use of robots in real-world, unstructured scenarios. Our results underscore the importance of biomechanically realistic inputs and temporally recurrent architectures for developing tactile perception models that

perceive touch similarly as animals do. Our demonstration that tactile-specific (sequential force/torque array) SSL augmentations significantly enhance performance underscores this point, emphasizing the necessity of tailored training methods for robotic tactile systems. Future work in embodied robotic systems that leverages these insights could overcome existing sensor limitations—scaling complexity, stimulus discrimination, and restricted sensing surfaces—to achieve more robust and nuanced interactions, similar to biological organisms; the next chapter pursues exactly this forward-engineering direction.

Limitations and Outlook. The principal limitation of this study is the neural data itself: currently available tactile recordings remain severely limited in stimulus diversity, with few object conditions tested, which restricts the captured neural variability and likely depresses the inter-animal consistency ceiling against which our models are judged (Fig. Figure 3.7(b)). Expanding neural datasets to broader stimulus sets and more animals is therefore the most important next step—and, fittingly, the scalable tactile simulator developed in Chapter 4 offers machinery that could help generate the richer stimuli such experiments would require. A fuller discussion of limitations and future directions, including multimodal sensory integration and the relationship to robot tactile learning, is deferred to the synthesis in Chapter 5.

Chapter 4

Scalable Tactile Simulation for Learning Dexterous Robot Tasks

Where Chapter 3 reverse-engineered tactile representations from the rodent brain and found recurrent processing of force/torque signals to be central, this chapter turns to *forward-engineering* tactile representations for robots. It asks the mirror-image question—of all the tactile signals a hand *could* provide, which does a dexterous manipulation policy actually need, and when do richer tactile fields justify their hardware cost?

This is hard to study empirically because, as Section 2.2 noted, each real sensor effectively defines a different hand, so no lab can run the same learning experiment across all of them. We therefore develop a GPU-parallel tactile sensor simulator that exposes binary contact, contact depth, per-taxel force/torque, elastomer marker displacement, geometry-aware proximity, and a voxelized temperature field (the first of its kind in a robot-learning simulator) under one configurable interface, together with a realistic noise model of drift, hysteresis, dead taxels, and crosstalk. The platform scales past 16,384 parallel environments and 10,000 taxels per hand on a single GPU—fast enough to serve as the tactile front-end for dexterous reinforcement learning (Fig. Figure 4.1). Holding the task, hand, and policy fixed and varying only the tactile abstraction, we find that proprioception alone is never enough, that sensor *placement* matters more than sensor type or resolution—whole-hand coverage closes most of the gap to a privileged teacher, while a few hundred taxels suffice—and that per-taxel force/torque is a strong default, with the findings transferring to the real XHand1.

4.1 Introduction

Dexterous manipulation is fundamentally contact-rich. Vision can localize objects before contact, and proprioception can track the robot's own motion, but many manipulation failures happen through local phenomena that neither modality observes directly: slip, incipient loss of force closure, decoupled object-hand motion, small contact timing errors, and

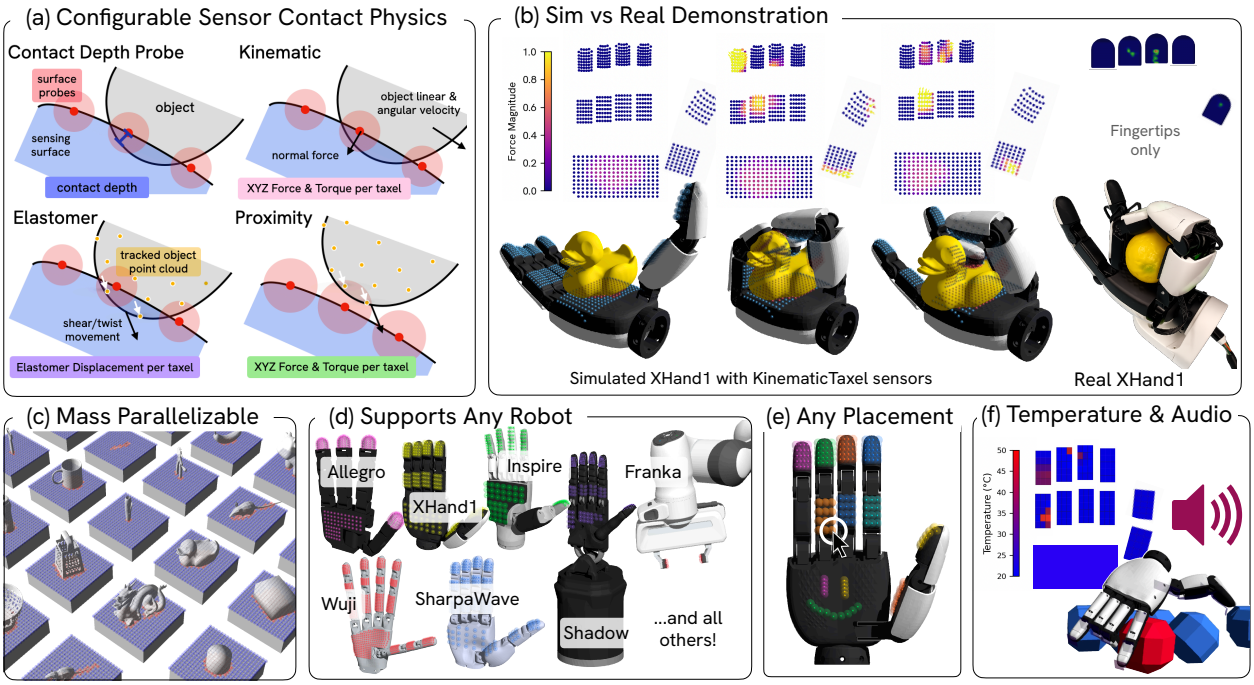


Figure 4.1. **Overview of our tactile simulator features.** (a) The sensor physics can be configured to match their real sensor analogues, including 6-axis force/torque measurements, elastomer displacement, and proximity signal. (b) A visual comparison of the simulated tactile force reading per taxel on an XHand1 compared to the real XHand1's sensor fidelity. (c) Our sensor implementations are highly parallelized and supports heterogeneous objects and randomization. (d) The simulated tactile sensors can be applied on any robot hardware surface and (e) the placement can be of arbitrary shape and resolution. (f) A temperature sensor which simulates contact heat transfer, heat diffusion, heat generation, and radiation, and a contact audio sensor which outputs high frequency signals based on material properties which the rigid body physics engine alone cannot capture.

hidden contacts inside clutter. Biological manipulation makes the same point from the other direction: humans and animals can manipulate objects with little or no visual feedback once contact is established. The question for robot learning is therefore not whether touch can help, but what kind of tactile information a policy needs.

Current tactile hardware spans capacitive arrays [35], magnetic skins [111], vision-based elastomer sensors [112], strain gauge [36], contact microphones [36, 42], multisensory fingertips [39–41]. These sensors differ in spatial resolution, bandwidth, cost, durability, wiring complexity, calibration burden, and suitability to cover the entire hand versus only the fingertip. A lab that buys or builds one sensorized hand usually cannot repeat the same dexterous learning experiment across all of these alternatives.

Simulation gives us the missing controlled comparison, but only if the simulated tactile sensing is realistic enough to transfer and fast enough for large-scale policy training. Previous work has already shown that carefully calibrated tactile simulators can support sim-to-real manipulation for visuotactile sensors [4–6]. **We ask a complementary design question:** if we can simulate many tactile abstractions at scale, which representation of touch is sufficient to learn general-purpose dexterous manipulation tasks?

We present a scalable and hardware-agnostic platform for tactile sensor simulation. The simulator covers a representative set of contact abstractions under a unified interface (Fig. Figure 4.1): binary contact, raw contact depth, per-taxel kinematic force/torque, elastomer marker displacement, and geometry-aware proximity. Each sensor exposes configurable placement, resolution, and noise parameters (drift, hysteresis, dead taxels, crosstalk). The implementation is GPU-parallelized across thousands of environments and taxels, making it usable as the tactile front-end for dexterous reinforcement learning (Fig. Figure 4.3). Using this platform, we train teacher-student policies on three dexterous tasks and ablate sensor type, placement, resolution, and noise. Our findings give concrete guidance for which tactile abstractions are worth the hardware cost on a given manipulation task. In summary, our contributions are:

1. We introduce a GPU-parallel tactile simulation platform that unifies diverse tactile sensing abstractions under a common configurable interface, scaling to over 16,000 parallel environments and 10,000+ taxels per hand on a single GPU.

2. We implement a temperature sensor, the first in any robot learning simulation, and use it to learn a policy that locates a hot object among geometrically identical distractors from proprioception and temperature alone. We ablate the thermal properties of the sensing surface and show that the low sensitivity of current real hardware is insufficient for learning this task.

3. We perform a controlled study of tactile representations across dexterous tasks, robot hands, sensor placements, resolutions, and noise settings.

4. We show that tactile placement matters more than sensor type, with whole-hand coverage substantially outperforming fingertip-only sensing, and that per-taxel force/torque is a strong default representation.

All code is available at `https://github.com/neuroagents-lab/tactile-genesis`.

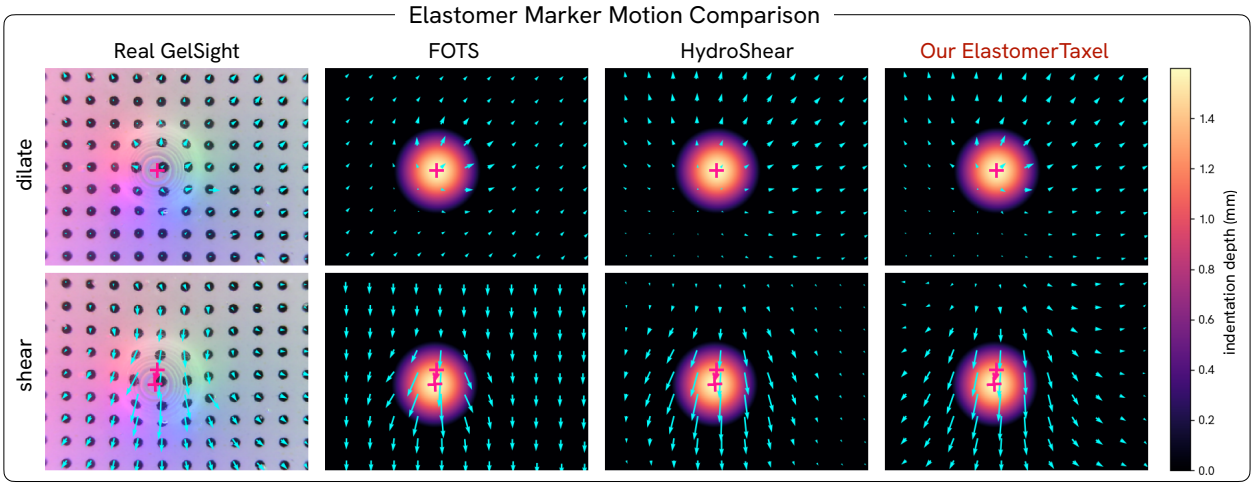


Figure 4.2. **Elastomer marker motion comparison.** Marker displacement fields on a real GelSight under dilation (normal indentation) and shear (tangential drag), compared with FOTS [4], HydroShear [5], and our `ElastomerTaxel`. The table on the right reports the relative marker-displacement error for each simulator after optimizing parameters to match the real image. The real GelSight image was obtained from the FOTS paper codebase, and we replicate the setup in sim, using our `ContactDepthProbe` sensor to measure depth.

	Relative RMSE (↓)		
	FOTS	HydroShear	Ours
dilate	0.514	0.403	**0.329**
shear	0.210	0.217	**0.174**

4.2 Related Work

We review the broader landscape of tactile hardware, simulation, and representation learning in Sections 2.2–2.4; here we position our platform against the most directly related work. Because each sensor effectively defines a different hand, comparing sensing strategies apples-to-apples on a fixed hand and task is rarely feasible in hardware—the gap that simulation fills.

4.2.0.1 Tactile simulation.

Tactile simulators fall into two families: full deformable physics, typically the Finite Element Method (FEM), and rigid-body simulation with soft-contact post-processing. FEM is accurate but historically slow for robot learning, though Taccel [44] narrows the gap with GPU-parallel incremental potential contact on the sensor surface. The rigid-body family extracts tactile signal from existing contact queries. Tacmap [6] casts rays from the sensor surface to produce per-pixel contact depth, which is a good match for rigid-pad finger-

tips such as SharpaWave and XHand1. We support the same query via either raycasting or a Signed Distance Function (SDF), and benchmark against Tacmap in Fig. Figure 4.4. FOTS [4] simulates the visual output of GelSight-style sensors directly, bypassing the deformation physics by modeling how the elastomer indentation maps to the camera image. HydroShear [5] extends this idea to GPU-parallel marker displacement, combining SDF-based depth with anchored point-cloud tracking to capture shear and twist. Our analogous Elastomer sensor builds on HydroShear with substantially better throughput and memory usage (Fig. Figure 4.4). TacSL [8] is likewise GPU-parallel, integrated into Isaac Lab, and renders a tactile depth image, RGB image, and a penalty-based tactile force field; we benchmark our throughput against it in Fig. Figure 4.4. We do not render a tactile RGB image directly, since its appearance is specific to the sensor vendor (e.g. GelSight), but it can be derived from our `ContactDepthProbe` depth with an example-based renderer such as Taxim [113]. Yin et al. [45] uses sampled point-cloud tracking, not for deformation but to estimate ReSkin [111] magnetometer readings; we include a corresponding Proximity sensor variant as well. To our knowledge, no prior simulator offers all of binary contact, depth, kinematic force/torque, elastomer displacement, and proximity under one interface, let alone a temperature field—which is what makes the controlled representation ablation in this chapter possible.

4.2.0.2 Which tactile representation a policy needs.

What tactile information a policy actually requires is an open empirical question, and prior work typically commits to a single point in the design space because the underlying hardware does—from sparse binary contacts, which are already sufficient for several in-hand skills [63], to rich multimodal fingertips that fuse image, audio, motion, and pressure [41], with force/torque arrays and elastomer displacement fields in between. Rather than commit, we hold the policy architecture, task, and hand fixed and vary the tactile abstraction directly in simulation, asking which abstraction is sufficient on which task.

4.3 Methods

4.3.1 Tactile Sensor Simulation

We integrate our tactile sensors into the open-source Genesis World physics simulator [114], exposing 7 sensor abstractions summarized in Table Table 4.1, which together span the design space of current tactile hardware. All sensors share a common pose-and-radius geometry, can be attached to arbitrary surfaces of any robot, and expose both a clean and a noisy readout under a configurable noise model (Section 4.3.2). Our Elastomer sensor

Sensor Type	Data Description	Shape
Surface Distance Probe	Shortest distance to the surface of tracked object.	(N)
Contact Depth Probe	Raw contact depth scalar from simulated physics, either from SDF or raycasting and sphere-triangle intersection.	(N)
Contact Probe	Binarized contact with depth threshold and hysteresis	(N)
Kinematic Taxel	Per-taxel force/torque estimate from SDF depth, contact normal, and linear/angular velocity of object in contact.	$(N, 6)$
Proximity Taxel	Measures object surface mass within sensing distance using point cloud sampled on tracked objects for signal strength. Computes force/torque using object velocities.	$(N, 6)$
Elastomer Taxel	Marker displacement modeling the sensing surface as an elastomer.	$(N, 3)$
Temperature Grid	Temperature in $°C$ over voxelized link.	(N)
Contact Audio	Block of K synthesized vibration samples per step.	(N, K)

Table 4.1. **Sensor implementations.** N represents the number of probes/taxels. The full implementation details and configurable parameters per sensor are available below.

approach is based on HydroShear [5] and the Proximity sensor extends Yin et al. [45]; full equations are in Section 4.3.2. In these prior works, these sensor simulation approaches have been validated to transfer to a real GelSight Mini on a robot gripper and ReSkin on the Allegro hand. Our `ElastomerTaxel` extends HydroShear with an elastomer compressibility term and a clamped boundary condition, which lowers the marker-displacement error against real GelSight under both dilation and shear motion (Fig. Figure 4.2).

Compared to our predecessors, our platform improves throughput by up to 20× at matched sensor configurations and reduces GPU memory per environment by roughly 5× (Fig. Figure 4.4), with the gap widening as the number of parallel environments grows. Three implementation choices drive this. First, the per-probe contact, depth, and force kernels are vectorized over probes *and* environments, so a single launch covers the entire batch rather than iterating per sensor. Second, mesh and point-cloud queries (SDF lookups, sphere–triangle intersection, proximity neighbor search) are accelerated by Bounding Volume Hierarchies (BVHs) over the tracked geometry, which keeps cost sublinear in the number of points. Third, for the elastomer sensor's dilation kernel and the spatial crosstalk model, we exploit the regular planar taxel grid to replace the dense convolution by a 2D Fast Fourier Transform with separable kernels. Together these allow us to scale the platform past 16,384 parallel environments (Fig. Figure 4.3) and beyond 10,000 taxels per hand (Fig. Figure 4.4c) on a single GPU, which is the regime needed for student-policy training.

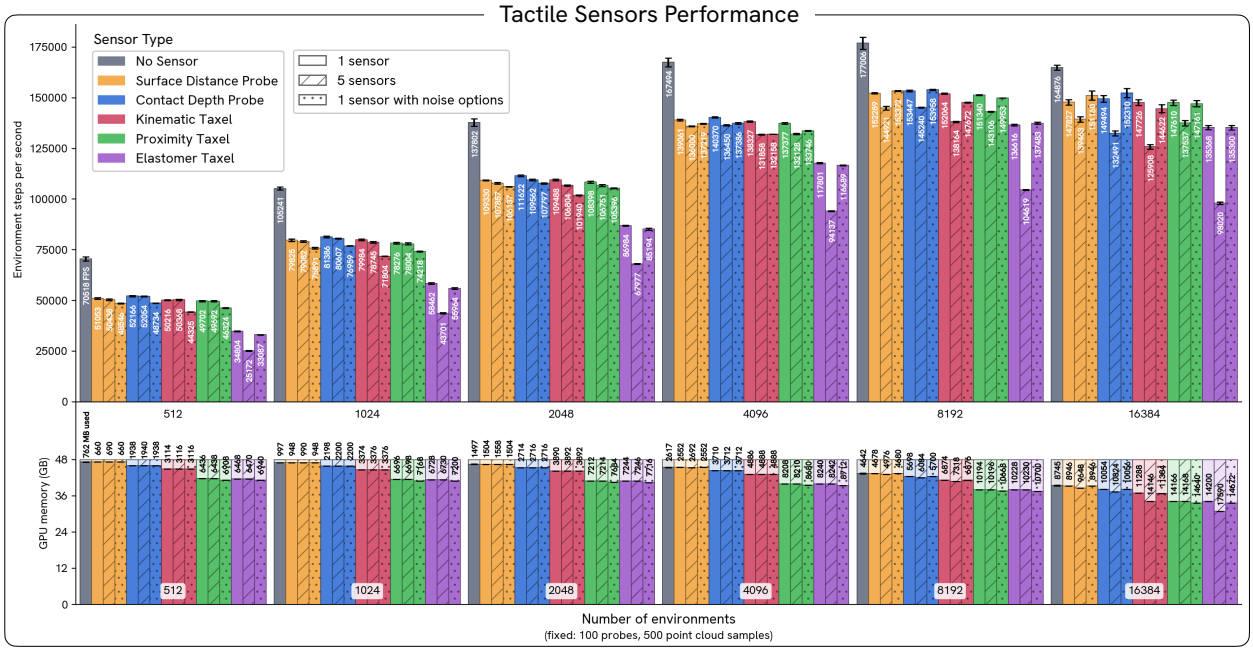


Figure 4.3. **Performance benchmark per each simulated sensor type.** We demonstrate that our sensors are able to be parallelized on a single NVIDIA RTX A6000 beyond 16,384 environments, with a total throughput of 150,000 environment steps per second (FPS). To isolate the effect of the sensors, we perform the benchmark in a simple scene of a pyramid of 10 cubes. We compare the performance of different sensor types, fixing the sensor resolution at 10×10 (100 taxels) and 500 tracked point cloud samples (applicable to Elastomer and Proximity sensors). Adding more sensors or noise parameters adds small overhead (-10% FPS) for most sensors. The Elastomer sensor slows as more sensors are added (-22% FPS) since local displacement effects must be computed per sensor.

4.3.2 Tactile Sensor Implementation

This section gives the mathematical definitions of our tactile sensor implementations, which have been mostly integrated into the open-source Genesis World physics simulation platform since its v1.0 release.

Probe geometry and contact-depth query.

Each probe is rigidly attached to a sensor link and described by a link-frame position, unit normal, and radius. Let sensor link L have pose (p_L, R_L) and let probe j have link-frame position x_j, unit normal n_j, and radius R_j. Its world-frame position and normal are

$$q_j = p_L + R_L x_j, \qquad a_j = R_L n_j. \tag{4.1}$$

Depth and force probes only consider collision geometries that are currently in a rigid-body contact pair with the sensor link. A probe with $R_j = 0$ is treated as an inactive filler and

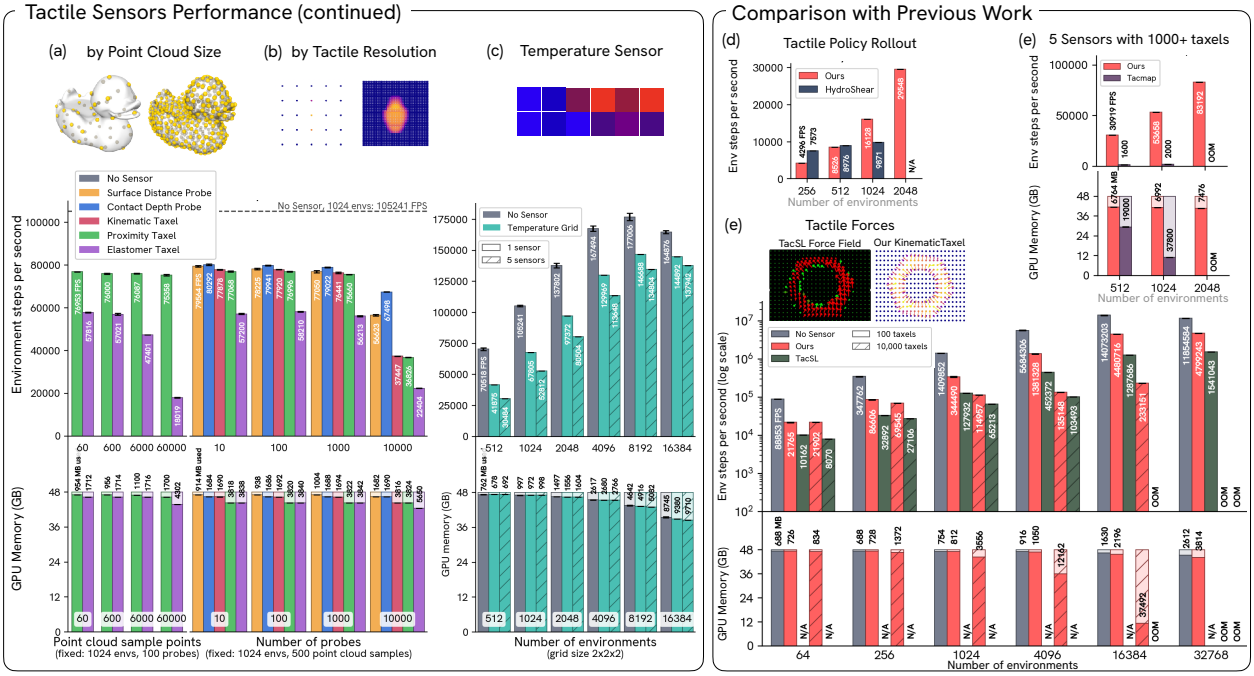


Figure 4.4. **Performance benchmark (continued).** (a, b) With the number of environments fixed at 1024 and varying point-cloud size and taxel count, our sensors retain low GPU memory and high throughput up to over 10,000 taxels per hand. The elastomer sensor has to track the motion of each object point in contact and therefore scales less well with point-cloud size, but point clouds larger than ∼6000 are not typically needed for dexterous tasks. (c) Our temperature sensor also scales well with number of environments, achieving 80% of the no sensor baseline throughput FPS with 5 active sensors with 8 voxels each.

Comparison with previous work. Comparison to performance numbers reported by Tacmap [6] and HydroShear [5]; neither paper reports the GPU used or metrics beyond 1024 environments. All of our benchmarks were run on one NVIDIA RTX A6000. (d) Tacmap is based on SharpaWave, which has $> 1,000$ tactile pixels per fingertip [7]; our FPS for 10,000 taxels across 5 ContactDepthProbe sensors is 20× Tacmap's and uses 7× less GPU memory per environment. (e) HydroShear reports per-step time for a single 7×9 (35 taxels) elastomer sensor on a robot arm; we rollout a trained robot-hand policy with five 5×4 (100 taxels) elastomer sensors and achieve higher FPS as the number of parallel environments grows (1.6× HydroShear's at 1024 envs). (f) Comparison against TacSL [8] reported FPS for their penalty-based force field vs. our KinematicTaxel sensor FPS and GPU memory usage for 10x10 and 100x100 taxels. We consistently achieve around 3x higher throughput (note the log scale on FPS) and can run at 16k envs without running out of memory.

returns zero, which keeps batched tensors regular over heterogeneous link layouts.

Two contact-depth backends. The penetration depth at probe j can be queried in two ways, selected by the `contact_depth_query` option. The `sdf` backend queries the per-geometry analytic signed-distance field maintained by the rigid solver. For an opposing collision geometry g with SDF ϕ_g, the penetration depth is

$$d_j^{\text{sdf}} = \max_g [R_j - \phi_g(q_j)]_+, \tag{4.2}$$

and the contact normal is the SDF gradient $m_j = \nabla \phi_{g^\star}(q_j)$ at the winning geometry $g^\star$ when $d_j^{\text{sdf}} > 0$. This is fast and exact on primitives, but requires SDF activation on the collider. The `raycast` backend instead walks a per-frame Bounding Volume Hierarchy over the rigid-body collision meshes, shared with Genesis's raycaster sensor. At each candidate BVH leaf, the probe runs a sphere–triangle closest-point test (penetration $= R_j -$ dist) and a ray–triangle test along $-a_j$ (penetration $= R_j -$ hit_distance); the deepest of the two wins, and the contact normal is the face normal of the deepest-penetrating triangle. The two backends produce equivalent depths to leading order for smooth meshes and differ at sharp features. `sdf` is the default; `raycast` handles arbitrary triangle meshes uniformly and is preferable when the scene mixes many small or thin objects.

`SurfaceDistanceProbe`.

The `SurfaceDistanceProbe` reports, per probe, the shortest distance from the probe center to the surface of each tracked object. Unlike the contact sensors below, it does not require a rigid-body contact pair and is defined before contact, which makes it a pre-contact proximity cue.

`ContactProbe`.

The `ContactProbe` binarizes the measured penetration depth d_{bj}^{m} at probe j into a contact bit, with optional Schmitt hysteresis to suppress chatter near the contact boundary:

$$y_{bj}^{\text{contact}}(t) = \mathbb{I}\left[d_{bj}^{\text{m}}(t) > \eta_{\text{on}} \ \vee \ \left(y_{bj}^{\text{contact}}(t-1) = 1 \wedge d_{bj}^{\text{m}}(t) > \eta_{\text{off}}\right)\right]. \tag{4.3}$$

Here η_{on} is the contact threshold and $\eta_{\text{off}} \leq \eta_{\text{on}}$ the release threshold; setting $\eta_{\text{off}} = \eta_{\text{on}}$ disables the hysteresis. We use $\eta_{\text{on}} = 5 \times 10^{-4}$ m, adding a release threshold $\eta_{\text{off}} = 2 \times 10^{-4}$ m and dead-taxel noise in the noisy condition (Table Table 4.12).

`ContactDepthProbe`.

The `ContactDepthProbe` reports the raw measured penetration depth d_{bj}^{m} at each probe, taken from the `sdf` or `raycast` backend of the previous section.

`KinematicTaxel`.

The `KinematicTaxel` converts depth, contact normal, and relative velocity into a per-taxel six-channel force/torque estimate without modeling a deformable substrate. Let $\bar{m}_j =$

$R_L^\top m_j$ be the SDF normal in the sensor-link frame. For a contact with opposing link C, the point velocities are

$$v_C(q_j) = v_C + \omega_C \times (q_j - r_C), \tag{4.4}$$
$$v_L(q_j) = v_L + \omega_L \times (q_j - r_L), \tag{4.5}$$

where r_C, r_L are the corresponding centers of mass. The relative velocity in the sensor-link frame is

$$v_{\mathrm{rel},j} = R_L^\top \left(v_C(q_j) - v_L(q_j) \right). \tag{4.6}$$

With $s_j = (d_j)^\alpha$, the normal and tangential components are

$$v_{n,j} = (v_{\mathrm{rel},j}^\top \bar{m}_j)\bar{m}_j, \qquad v_{t,j} = v_{\mathrm{rel},j} - v_{n,j}. \tag{4.7}$$

The local force and torque are

$$f_j = k_n s_j \bar{m}_j + c_n s_j v_{n,j} - k_t v_{t,j}, \tag{4.8}$$
$$\tau_j = x_j \times f_j - k_\omega \left((\omega_C - \omega_L)^\top m_j \right) \bar{m}_j. \tag{4.9}$$

If no opposing contact link is available we drop the velocity terms, returning only the spring force and $\tau_j = x_j \times f_j$. The sensor reports the six channels (f_j, τ_j) per taxel. The parameters used in our experiments are

$$k_n = 500, \qquad c_n = 1, \qquad \alpha = 1.2, \qquad k_t = 2, \qquad k_\omega = 2. \tag{4.10}$$

For regular planar taxel grids, the kinematic taxel additionally mixes neighboring channels to model spatial crosstalk between adjacent sensing sites, described together with the other sensor noise models in Section 4.3.2.

`ProximityTaxel`.

The `ProximityTaxel` models a geometry-aware taxel that responds to nearby tracked mass, mirroring how capacitive and magnetic skins behave. Tracked links are sampled into a surface point cloud at scene reset and organized in a static BVH for fast neighbor queries. For taxel j, let $\mathcal{P}_j = \{i : \|p_i - q_j\| < R_j\}$ be the set of tracked points inside the taxel sensing sphere. Each point contributes

$$P_{ij} = R_j - \|p_i - q_j\|. \tag{4.11}$$

Let

$$v_{\mathrm{tax},j} = v_L + \omega_L \times (q_j - r_L), \qquad v_{p_i} = v_i + \omega_i \times (p_i - r_i), \tag{4.12}$$

and remove the taxel-normal component of relative velocity:

$$v_{t,ij} = (v_{p_i} - v_{\text{tax},j}) - a_j \left((v_{p_i} - v_{\text{tax},j})^\top a_j \right). \tag{4.13}$$

The density scale is

$$\rho_b = \frac{\texttt{density\_scalar}}{\max(N_{\text{active},b}, 1)}. \tag{4.14}$$

The world-frame force and torque are

$$f_{bj}^w = k\rho_b \sum_{i \in \mathcal{P}_j} P_{ij} a_j + k_s \rho_b \sum_{i \in \mathcal{P}_j} P_{ij} v_{t,ij}, \tag{4.15}$$

$$\tau_{bj}^w = k\rho_b \sum_{i \in \mathcal{P}_j} P_{ij} \left((p_i - q_j) \times a_j \right). \tag{4.16}$$

The reported channels are $R_L^\top f_{bj}^w$ and $R_L^\top \tau_{bj}^w$. Our experiments use $R_j = 0.01$ m, $N_{\text{pc}} = 5000$, $k = 300$, $k_s = 10$, and a density scalar of 100. Radius noise enters by perturbing R_j before the BVH query; per-probe gain scales the accumulated penetration, slip, and torque terms before the force/torque conversion.

`ElastomerTaxel`.

The `ElastomerTaxel` models marker displacements on a deformable substrate by combining SDF-driven dilation with anchored shear history [5]. Let

$$\Pi_n(z) = z - (z^\top n) n \tag{4.17}$$

be projection onto the tangent plane. For source probe i, the per-probe penetration into the elastomer is

$$h_i = \max\left(0, -\min_g \phi_g(q_i)\right). \tag{4.18}$$

For target marker j, the direct dilation contribution from source i is

$$u_{ij}^d = s_d \left(\Pi_{n_j}(x_j - x_i) h_i + n_j h_i^\alpha \right) \exp\left(-\lambda_d \left\| \Pi_{n_j}(x_j - x_i) \right\|^2 \right). \tag{4.19}$$

Tracked links are also sampled into surface points to capture shear history. A sampled point p initializes an anchor e_p when its elastomer SDF drops below $-\eta_{\text{enter}}$ and clears the anchor when the SDF rises above $+\eta_{\text{exit}}$. Let z_p be the current sensor-frame point position and h_p the elastomer penetration depth. The shear contribution to target marker j is

$$u_{pj}^s = s_s \Pi_{n_j}(z_p - e_p) h_p \exp\left(-\lambda_s \left\| \Pi_{n_j}(x_j - z_p) \right\|^2 \right). \tag{4.20}$$

The displacement output is

$$y_j^{\text{elastomer}} = \sum_i u_{ij}^d + \sum_p u_{pj}^s. \tag{4.21}$$

Our experiments use $N_{\text{pc}} = 1000$, $s_d = 100$, $s_s = 100$, $\lambda_d = 8000$, $\lambda_s = 2000$, $\alpha = 1.2$, and shear enter/exit thresholds $\eta_{\text{enter}} = 10^{-5}$ m and $\eta_{\text{exit}} = 10^{-4}$ m. For regular planar grids we accelerate the dilation term with FFT: tangent channels convolve h_i and the normal channel convolves h_i^α; shear is accumulated directly.

Sensor Noise Model.

Every sensor in our simulator exposes both a clean ground-truth readout and a noisy readout; only the noisy readout is fed to the policy in our noisy condition. The available noise knobs depend on the sensor type. Table Table 4.2 lists which knobs each sensor type supports; the numerical values we use in the experiments are in Section 4.3.4 (Table Table 4.12).

Base noise (every sensor). At the simulator level, every sensor honours a read delay Δ_{read} (with optional uniform jitter), additive Gaussian white noise σ, constant bias b, random-walk drift σ_{rw}, and uniform quantization with step q:

$$\tilde{y}_t = \text{quantize}\left(y_{t-\Delta_{\text{read}}} + b + n_t + d_t,\, q\right), \quad n_t \sim \mathcal{N}(0, \sigma^2), \quad d_t = d_{t-1} + w_t,\; w_t \sim \mathcal{N}(0, \sigma_{\text{rw}}^2). \tag{4.22}$$

Probe-level imperfections (probe-based sensors). Probe sensors additionally perturb the per-probe sensing radius and apply a per-probe multiplicative gain:

$$\tilde{R}_j = \max(0,\, R_j + \epsilon_j), \qquad \epsilon_j \sim \mathcal{U}[-\rho_j, \rho_j], \tag{4.23}$$

$$\gamma_{bj} \sim \mathcal{U}[\gamma_{\min,j}, \gamma_{\max,j}] \text{ at reset}, \tag{4.24}$$

where the gain γ_{bj} models persistent per-unit calibration error.

Taxel-level imperfections (tactile probe sensors). Tactile probe sensors further sample dead taxels at episode reset:

$$M_{bj} \sim \text{Bernoulli}(p_j), \qquad z_{bj} \sim \mathcal{U}[\ell_j, u_j], \tag{4.25}$$

and when $M_{bj} = 1$ the measured value at taxel j is overwritten by the stuck readout z_{bj} for the entire episode.

Viscoelastic hysteresis (contact, depth, kinematic, elastomer, proximity). Substrate-like sensors model a single-Maxwell viscoelastic loop on the noisy readout:

$$\xi_t = \exp(-\Delta t/\tau)\, \xi_{t-1} + (x_t - x_{t-1}), \tag{4.26}$$

$$\tilde{x}_t = x_t + \beta\, \xi_t, \tag{4.27}$$

with strength β and time constant τ. After a rising step the noisy signal overshoots and decays back to equilibrium; after a falling step it undershoots analogously.

Spatial crosstalk (kinematic taxel only). On regular planar grids, the kinematic taxel mixes neighboring channels to model spatial crosstalk between adjacent sensing sites. Let z_c be one force or torque channel on the grid, χ the crosstalk strength, and G_σ an L1-normalized Gaussian kernel of standard deviation σ:

$$\tilde{z}_c = (1 - \chi) z_c + \chi (G_\sigma * z_c), \tag{4.28}$$

applied independently to each of the six channels.

Noise knob	SurfaceDist.	ContactProbe	ContactDepth	Kinematic	Elastomer	Proximity
Read delay / jitter	✓	✓	✓	✓	✓	✓
White noise σ	✓	✓	✓	✓	✓	✓
Constant bias	✓	✓	✓	✓	✓	✓
Random-walk drift	✓	✓	✓	✓	✓	✓
Quantization	✓	✓	✓	✓	✓	✓
Probe-radius noise	✓	✓	✓	✓	✓	✓
Per-probe gain	–	✓	✓	✓	✓	✓
Dead taxels	–	✓	✓	✓	✓	✓
Viscoelastic hysteresis	–	✓	✓	✓	✓	✓
Spatial crosstalk	–	–	–	✓	–	–

Table 4.2. Imperfection knobs available per sensor type. A check means the knob is available for that sensor type and active in the noisy condition when set.

4.3.3 Dexterous Task Training

Our goal is to compare *tactile observation types*, not specific hardware, on a fixed set of dexterous tasks. We pair each simulator-level sensor class with a downstream postprocessing step to produce eight tactile observation types plus a proprioception-only baseline (`none`), listed in Table Table 4.3. The `agg_*` variants aggregate per-taxel signals to a single per-link value, matching the convention used by real fingertip force sensors in XHand1. The per-taxel variants instead expose the full tactile field to the policy. Holding placement, resolution, and policy architecture fixed across these types lets us isolate the effect of the abstraction itself.

For each task–hand tuple, we first train a privileged teacher with PPO using full object state, then distill a tactile student that replaces the privileged state group with one of the tactile observation types from Table Table 4.3 (Fig. Figure 4.5). The student is trained with behavioral cloning against the teacher's actions, plus auxiliary heads that decode privileged

Type	Sensor Type	Output (after optional postprocessing)
`none`	–	Proprioception-only baseline with tactile group removed.
`bool`	ContactProbe	Binarized contact *per taxel*.
`agg_bool`	ContactProbe	Binary contact *per link* after thresholding by the number of taxels in contact.
`depth`	ContactDepth-Probe	Contact depth *per taxel*
`agg_force`	ContactDepth-Probe	Estimated force *per link* based on summing contact depth along probe normals.
`force`	KinematicTaxel	Force *per taxel*.
`force_torque`	KinematicTaxel	Force and torque *per taxel*.
`elastomer`	ElastomerTaxel	XYZ marker displacement *per taxel* (marker) on an elastomer surface.
`proximity`	ProximityTaxel	Force and torque signals *per taxel* based on object surface proximity for signal strength.

Table 4.3. **Tactile observation types used by student policies.** We compare 8 different tactile representations, using the tactile signals from the simulated sensors and optionally postprocessing before passing into the observations. Tactile data *per link* means we aggregate the data and output one tactile signal per every sensing area (e.g. 5 fingertips).

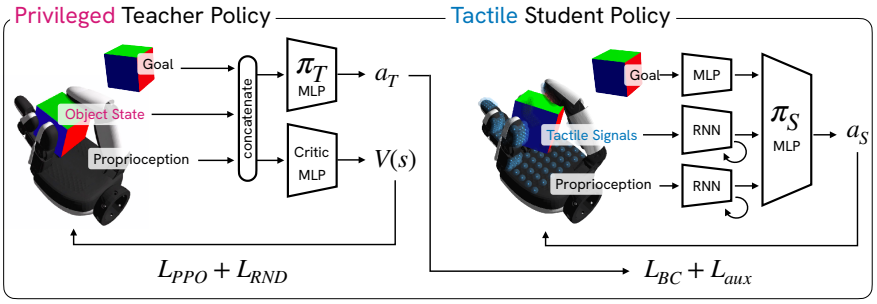


Figure 4.5. **Teacher-student training setup.** A privileged teacher is trained with PPO and an MLP actor-critic. We additionally incorporate a Random Network Distillation (RND) [9, 10] loss to explore states more quickly. Tactile student policies encode each observation group before passing to the MLP head. In addition to the DAgger [11] behavioral cloning (BC) loss, we incorporate auxiliary losses to decode object state. See Section 4.3.4 for the full training parameters.

object state from the policy's hidden representation. The decoders are not used at deployment; they act as a regularizer that pushes the tactile encoder to recover task-relevant object state from touch. Each tactile observation group is processed by its own small MLP encoder before being concatenated with the proprioception features and fed to the policy head. Full training hyperparameters and the per-task reward terms are listed below.

We evaluate 3 tasks spanning complementary contact regimes: `in_palm_rotate` requires reorienting an object on the palm with the thumb sweeping in to capture it, so locating the object before contact is informative. `in_hand_repose` requires reposing an object to a target pose while it is in near-continuous contact with multiple fingers, so slip and grip-strength signals dominate. `screwdriver` requires a fast finger gait that keeps a screwdriver spinning, so contacts are brief and rapidly changing. We sweep three tactile placements (`tips`, `fingers`, `hand`) at three resolution levels, with both clean and noisy sensor settings; the full matrix is in Tables Table 4.10–Table 4.12.

4.3.4 Training Setup

The training pipeline, task definitions, and tactile sensor wiring used in this chapter live in the companion `dexterous_hands` repository. This section documents the optimization and network hyperparameters, the observation–type mapping, the dexterous tasks, the sweep matrix, the per-resolution probe counts, and the noise values used in the noisy condition.

Optimization and Network Hyperparameters.

Table Table 4.4 lists the teacher PPO, RND exploration, and student DAgger optimization settings; values that differ across tasks are shown per task. Teacher and student share the same actor–critic backbone: a three-layer MLP head with hidden sizes $[512, 256, 128]$ and ELU activations, with input normalization and an initial policy standard deviation of 1.0. Each non-tactile observation group is embedded by a per-group encoder (an MLP $[512, 256, 128] \to 64$, or an LSTM with hidden size $128 \to 64$) before concatenation. Each tactile group is embedded by its own encoder projecting to a 32-dimensional embedding: gridless types use an MLP ($[64, 64]$) or an LSTM (hidden 64), while grid-structured types use a convolutional encoder (`tactile_cnn`, channels $[16, 32]$, kernel 3) or a convolutional–recurrent encoder (`tactile_convrnn`, 16 channels, kernel 3, layer norm). The RND predictor uses learning rate 10^{-3}, an 8-dimensional output, and normalized state and reward. The student additionally trains auxiliary heads that decode privileged object state from its latent (Table Table 4.5); these decoders are used only during training and discarded at deployment. The simulation steps at 200 Hz with a control decimation of 5 (40 Hz control).

Tactile Observation Types.

Table Table 4.6 shows how each downstream observation type maps to an underlying Genesis sensor abstraction together with the postprocessing applied to the raw read.

Each observation type derives from one native sensor (Section 4.3.2) followed by light postprocessing. Writing d_{bj}^{m} for the measured `ContactDepthProbe` depth and n_j for the link-frame probe normal:

	in_palm_rotate	in_hand_repose	screwdriver
Teacher (PPO)			
parallel environments	8192	8192	8192
steps per env	24	24	12
teacher iterations	6000	20000	5000
learning rate (adaptive)	10^{-3}	10^{-3}	10^{-3}
learning epochs / minibatches	5 / 4	5 / 4	5 / 4
discount γ	0.998	0.998	0.99
GAE λ	0.95	0.95	0.95
clip param / desired KL	0.2 / 0.01	0.2 / 0.01	0.2 / 0.01
entropy coef.	2×10^{-3}	2×10^{-3}	0
value loss coef. / max grad norm	1.0 / 1.0	1.0 / 1.0	1.0 / 1.0
episode length (s)	10	20	20
Student (DAgger)			
student iterations	6000	6000	2000
learning rate	10^{-4}	10^{-4}	10^{-5}
learning epochs	5	5	5
gradient length	2	2	4
clip param	–	–	0.1
max grad norm	2.0	–	0.5
BC loss	MSE	MSE	inv-var MSE

Table 4.4. Teacher PPO and student DAgger optimization hyperparameters. A dash means the knob is left at the library default (unbounded gradient norm, no extra clipping).

Task	Decoded target	Weight	Target scale	Head hidden dims
`in_palm_rotate`	object size	1.0	20.0	$[128, 64]$
`in_palm_rotate`	goal distance	1.0	1.0	$[128, 64]$
`screwdriver`	object tilt	0.5	1.0	$[512, 128]$
`screwdriver`	rotation progress	0.5	1.0	$[512, 128]$
`in_hand_repose`	none			

Table 4.5. Auxiliary decoder losses used during student distillation. Each head predicts a privileged scalar from the student latent under an MSE loss (target scaled by the listed factor); the total loss adds the weighted auxiliary terms to the DAgger behavioral-cloning loss. The decoders regularize the tactile encoder toward task-relevant object state and are discarded at deployment.

Type	Sensor abstraction	Key parameters	Policy observation
`bool`	`ContactProbe`	$\eta_{\text{on}} = 5 \times 10^{-4}$ m	thresholded contact bits
`agg_bool`	`ContactProbe`	$\eta_{\text{on}} = 5 \times 10^{-4}$ m; count threshold > 2	per-link aggregate contact bit
`depth`	`ContactDepthProbe`	`contact_depth_query` = `sdf`	flattened contact depths
`agg_force`	`ContactDepthProbe`	`sdf` query; 10^4 scale, ZYX axis order	per-link patch force
`force`	`KinematicTaxel`	$k_n = 500$, $c_n = 1$, $\alpha = 1.2$, $k_t = 2$, $k_\omega = 2$	force channels only
`force_torque`	`KinematicTaxel`	same as `force`	force and torque channels
`elastomer`	`ElastomerTaxel`	$N_{\text{pc}} = 1000$, $\lambda_d = 8000$, $\lambda_s = 2000$, $s_d = 100$, $s_s = 100$, $\alpha = 1.2$	marker displacements
`proximity`	`ProximityTaxel`	$R = 0.01$ m, $N_{\text{pc}} = 5000$, $k = 300$, $k_s = 10$	force and torque channels

Table 4.6. Mapping from downstream observation type to underlying Genesis sensor abstraction and key parameters.

Contact observations. `depth` flattens the per-taxel `ContactDepthProbe` depths. `bool` is the per-taxel `ContactProbe` bit, and `agg_bool` sums those bits over a link and reports a single bit when more than two taxels are in contact. `agg_force` aggregates the contact-depth probes into one patch force per link,

$$y_b^{\text{agg\_force}} = 10^4 \, P_{\text{ZYX}} \sum_j d_{bj}^{\text{m}} \, n_j, \qquad P_{\text{ZYX}}(f_x, f_y, f_z) = (f_z, f_y, f_x), \tag{4.29}$$

permuting axes and applying an empirical scale to match the per-fingertip force that the XHand1 `calc_pressure` channel returns on the real robot.

Force, displacement, and proximity observations. `force` and `force_torque` read the `KinematicTaxel`: `force_torque` keeps all six channels (f_j, τ_j), while `force` drops the torque, $h_{\text{force}}(F, T) = \text{vec}(F)$. `proximity` flattens the `ProximityTaxel` force and torque channels, and `elastomer` flattens the `ElastomerTaxel` marker displacements.

Except for `agg_bool`, `agg_force`, and `force`, the policy input flattens each raw sensor tensor and concatenates the results:

$$h_{\text{flat}}(Y_1, \ldots, Y_K) = \text{concat}\big(\text{vec}(Y_1), \ldots, \text{vec}(Y_K)\big). \tag{4.30}$$

Only the per-link aggregate types `agg_bool` and `agg_force` have a counterpart on the real XHand1; their deployment is described in Section 4.3.5.

Tasks and Hands.

`in_palm_rotate`. A partial-hand in-palm rotation task in which the policy controls only the thumb and middle finger; the index, ring, and pinky fingers are frozen. The reward rewards rotation around the commanded axis and penalizes off-axis tilt; the object is reset to a sampled bottom-aligned pose at the center of the palm.

`in_hand_repose`. A whole-hand reposing task in which the policy must drive the object pose to a target orientation while keeping it in continuous contact with multiple fingers. The reward combines orientation tracking with grip-strength and slip penalties.

`screwdriver`. A finger-gait task in which the hand spins a screwdriver about its long axis while keeping the tip engaged. The reward rewards angular progress and penalizes losing contact or dropping the tool.

Observations. All three tasks share the same observation grouping (Table Table 4.7). The student (deployable) policy sees only proprioception (joint positions and velocities and the last action), the tactile group (one observation type from Table Table 4.3), and, for the reorientation tasks, the goal. The privileged groups available to the teacher and critic add per-link contact forces, full object state, and object properties. `screwdriver` has no goal group and folds the surface-distance probes into its object-state group.

Group	Contents	Available to
`proprio`	joint positions, joint velocities, last action	student + teacher
`tactile_sensors`	one tactile observation type (Table Table 4.3)	student
`goal`	goal orientation and angular error to target	student ‡
`priv_proprio`	proprioception plus per-link contact forces	teacher, critic
`priv_obj_state`	object position, 6D orientation, linear and angular velocity	teacher, critic
`priv_obj_props`	object friction, mass, and surface-distance probes	critic

Table 4.7. Observation groups. The student policy is restricted to `proprio`, `tactile_sensors`, and (where present) `goal`; the privileged groups are used only to train the teacher and critic. ‡`in_palm_rotate` and `in_hand_repose` only; `screwdriver` has no goal group and places the surface-distance probes in `priv_obj_state`.

Reward terms. Table Table 4.8 lists the reward-term weights for each task; blank entries mark terms not used by that task.

Domain randomization. Every task applies the same actuator randomization, re-sampled per reset: proportional and derivative gains and motor strength each scaled by $\mathcal{U}[0.95, 1.05]$, a position bias in ± 0.01 m, a deadband in $[0, 0.005]$, gear backlash in $[0, 0.01]$

Reward term	`in_palm_rotate`	`in_hand_repose`	`screwdriver`
success bonus	150	500	–
orientation tracking (Gaussian)	–	2.0	–
orientation tracking (inverse-L_2)	–	2.0	–
rotation progress / spin rate	2.0	0.2	10.0
surface-distance reward	0.1	0.5	1.0
off-axis orientation penalty	-1.0	–	–
vertical-alignment penalty	–	–	-400
object-distance penalty	-100	–	-300
hand pose-deviation penalty	–	–	-1.0
contact-force penalty	-5.0	-5.0	–
drop / termination penalty	-100	-100	-100
goal-timeout penalty	–	-200	–
action-rate penalty	-5×10^{-4}	-1×10^{-5}	-1×10^{-2}
joint-limit penalty	-40	-20	-40
work penalty	-5×10^{-3}	-1×10^{-3}	-1×10^{-3}

Table 4.8. Reward-term weights per task. Positive weights are shaping rewards; negative weights are penalties. A dash means the term is not used by that task.

rad, a torque-kick ratio in $[0.9, 1.1]$, and per-step torque noise at RFI scale 0.1. The task-specific object and disturbance randomization is listed in Table Table 4.9.

Quantity	`in_palm_rotate`	`in_hand_repose`	`screwdriver`
object friction ($\times$)	$[0.5, 1.5]$	$[0.3, 2.0]$	$[0.2, 1.2]$
object mass offset (kg)	$[0, 0.1]$	$[0, 0.1]$	$[0, 0.2]$
initial placement xy (m)	± 0.02	± 0.02	sampled grasp
initial yaw (rad)	$\pm \pi/12$	$\pm \pi$	sampled grasp
external force x, y (N)	± 1	± 1	± 5
external force z (N)	± 1	± 1	$[-5, 0]$
force interval (s)	2–4	2–4	2–4

Table 4.9. Task-specific domain randomization, on top of the shared actuator randomization described in the text. External forces are reapplied at random intervals within the listed range. `screwdriver` initializes from a set of pre-sampled grasps rather than a randomized free placement.

Distillation Sweep.

Each task–hand entry in Table Table 4.10 is expanded into the proprioception-only baseline (`none`) plus the cross product of taxel resolutions (`low`, `med`, `high`), the available placement subsets for that hand, the seven tactile observation types, and both clean and noisy

sensor settings. Type-ablation runs pin resolution to `med` and use the clean setting; resolution and noise sweeps vary one axis at a time.

Task	Robot	Placement subsets swept	Max student iters
`in_palm_rotate`	`xhand1`	`tips`, `hand`	6000
`screwdriver`	`xhand1`	`fingers`	2000
`in_hand_repose`	`xhand1`	`hand`	6000
`in_hand_repose`	`sharpa`	`hand`	6000

Table 4.10. Task–hand entries in the distillation sweep.

Tactile Placement and Probe Counts.

Sensor placement is encoded as a probe asset per (robot, resolution) pair; placement subsets such as `tips`, `fingers`, and `palm` are link filters applied at runtime to the full `hand` probe set. Table Table 4.11 shows the resulting probe counts.

Robot	Resolution	Hand	Tips	Fingers	Palm
`xhand1`	`low`	90	45	78	12
`xhand1`	`med`	199	100	164	35
`xhand1`	`high`	667	224	448	219
`sharpa`	`low`	98	45	89	9
`sharpa`	`med`	206	70	190	16
`sharpa`	`high`	781	245	660	121

Table 4.11. Number of active probes per (robot, resolution) and placement subset used in our experiments. `tips`, `fingers`, and `palm` are link-filtered subsets of the full `hand` probe set; `fingers` includes the fingertips.

Sensor Imperfection Parameters in the "Noisy" Condition.

Table Table 4.12 lists the numerical values used in the noisy condition for each observation type, layered on top of the clean parameters from Table Table 4.6. The available knobs and their semantics are defined in Section 4.3.2 (Table Table 4.2); knobs not listed here are zero or disabled.

4.3.5 Sim-to-Real Deployment

The tactile observation term is the same object in simulation and on hardware: at each step it first looks for a real-hand reading and falls back to the simulated sensors when none is present. This lets a policy trained in simulation run on the real XHand1 without changing its observation interface.

Type	σ	`rw`	`quant`	ρ (m)	rel. thr. (m)	p_{dead}	γ	(β, τ)
`bool`, `agg_bool`	–	–	–	–	2×10^{-4}	0.05	–	–
`depth`, `agg_force`	2×10^{-4} m	–	10^{-4} m	3×10^{-4}	–	0.05	$[0.85, 1.15]$	$(0.5, 0.05)$
`force`, `force_torque`	5×10^{-3} N	10^{-4} N	10^{-2}	3×10^{-4}	–	0.05	$[0.85, 1.15]$	$(0.5, 0.05)$
`proximity`	10^{-2}	10^{-3}	10^{-2}	6×10^{-4}	–	0.05	$[0.85, 1.15]$	$(0.5, 0.05)$
`elastomer`	10^{-4} m	10^{-5} m	10^{-4} m	3×10^{-4}	–	0.05	$[0.85, 1.15]$	$(0.5, 0.05)$

Table 4.12. Sensor imperfection parameters layered on the clean configuration (Table Table 4.6) in the noisy condition. σ is additive white noise, `rw` random-walk drift, `quant` the quantization step, ρ probe-radius noise, "rel. thr." the Schmitt release threshold, p_{dead} the dead-taxel probability, γ the per-reset gain range, and (β, τ) the viscoelastic hysteresis strength and time constant (in seconds). Dashes mark knobs that are disabled for that sensor type.

The real XHand1 SDK reports both a per-taxel raw pressure field and an aggregate contact pressure $f_{\text{calc\_pressure}} = (f_x, f_y, f_z)$ per fingertip. The hand does therefore expose individual taxel values, but the exact position and response characteristics of each taxel are not documented, so we cannot register the raw field to our simulated probe layout; the most faithful usable signal is the SDK's aggregate contact pressure. Consequently only the two per-link aggregate observation types transfer to hardware; the per-taxel types (`bool`, `depth`, `force`, `force_torque`, `proximity`, `elastomer`) have no faithful counterpart on the real fingertip and are simulation-only. The simulated `agg_force` is defined precisely so that its per-link output matches $f_{\text{calc\_pressure}}$ in axis order and scale, so on hardware we read the contact pressure directly. For `agg_bool` we threshold the contact-pressure magnitude per finger,

$$y^{\text{agg\_bool,real}} = \mathbb{I}\left[\|f_{\text{calc\_pressure}}\|_2 > \eta_{\text{real}}\right]. \tag{4.31}$$

The real hand resolves a single value per finger, whereas the simulated observation may carry several per-link slots (one per kept substep of the within-step history). We therefore repeat each per-finger real reading across those slots, so the deployed observation has exactly the layout and dimensionality the policy was trained on.

4.4 Results

4.4.1 Tactile Student Ablations

4.4.1.1 Proprioception is not enough.

In Fig Figure 4.6, we can see that the `none` baseline trails every tactile student on all three tasks, including the cheapest binary contact variant. The auxiliary state decoders alone are

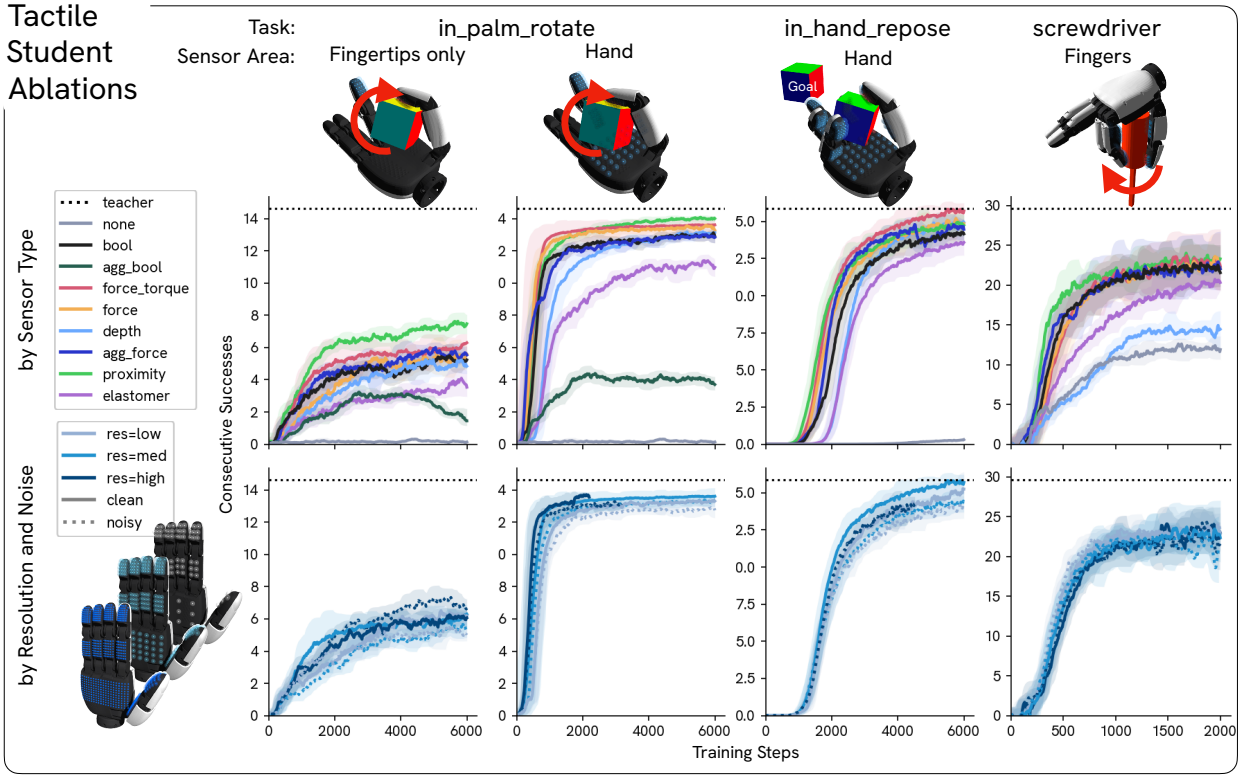


Figure 4.6. **Tactile student ablations.** For 3 tasks `in_palm_rotate`, `in_hand_repose`, and `screwdriver` using the XHand1, we compare tactile data types against the privileged teacher and distilled tactile student. For `in_palm_rotate` we try having fingertips only (the real XHand1 only has fingertips) vs including sensors on the whole hand. We also vary the tactile resolution and add noise parameters (white noise, random walk drift, hysteresis, sensing radius noise) and compare. A complete table of parameters is given in the Methods above (Tables Table 4.10–Table 4.12).

not enough to recover task-relevant object state from proprioception; any meaningful student performance requires the tactile group.

4.4.1.2 Placement dominates sensor type.

Restricting sensing to fingertips, the placement that most current commercial hardware supports, trails whole-hand coverage by a large margin on `in_palm_rotate`. Adding the palm and mid-finger surfaces closes most of the remaining gap to the privileged teacher, even for sensor types that individually carry less information. Adding taxels on the palm and proximal phalanges of the fingers is, at the margin, more useful than upgrading the fingertip sensor.

4.4.1.3 The best sensor type is task-dependent, with force/torque as a robust default.

On `in_hand_repose`, where the object is in near-continuous contact and the dominant failure mode is incipient slip, the `force_torque` student is best and clearly separates from the binary and depth variants. On `in_palm_rotate`, `proximity` edges out the contact-only types, since its sensing radius registers the approaching object before direct contact and lets the thumb pre-shape rather than search. On `screwdriver`, where contacts are brief and the fingers gait quickly, all tactile signals perform similarly and none saturates the teacher; we conjecture that integration over time, or visual feedback, is the missing channel here rather than a different contact abstraction. Aggregated across tasks, per-taxel `force_torque` matches or outperforms every other type and is our recommended default when hardware allows.

4.4.1.4 Elastomer displacement underperforms when per-taxel locality matters.

The `elastomer` student trails `force_torque` on both `in_palm_rotate` and `in_hand_repose`. This is consistent with how the substrate model behaves: marker displacement at one taxel is a function of indentation *and* shear at neighboring taxels, so a displacement in x can be caused by adjacent dilation as easily as by local shear. The resulting signal is well suited for inferring object shape patches, the use case GelSight-like sensors were originally designed for, but less suited for reading off the local force vector that the in-hand tasks actually need.

4.4.1.5 Sim-to-real validation.

We validate transfer by deploying the `in_palm_rotate` policy on the real XHand1, whose only tactile sensing is fingertip aggregate force. The deployed policy achieves one to two successes, which matches the success rate of the fingertip `agg_bool` student in simulation, the observation type that most closely mirrors the real fingertip readout. This confirms that policies trained in simulation transfer to hardware, and that the simulated fingertip abstraction is a faithful enough proxy for the real sensor to predict its performance.

4.5 Discussion and Limitations

Across all tasks, sensor types, and placements, the takeaways for practitioners outfitting a dexterous hand are threefold. First, cover the palm and proximal phalanges before paying for higher-end fingertip sensors. Second, prefer per-taxel force/torque as a default abstraction. Third, consider a proximity channel when the task involves capturing an object that approaches the hand rather than one already grasped. The first two findings are at odds

with the de facto convention in current commercial tactile fingertips, which concentrate spatial resolution on the distal pad and stop there. From a simulation-design perspective, our experiments also suggest that the dominant source of useful tactile information for these tasks is the coarse spatial distribution of contact rather than the fine-grained mechanics of the substrate. This is encouraging for rigid-body tactile simulation, since substrate physics is exactly what is most expensive to model faithfully.

We deliberately scope this chapter to sensor implementation and observation-type comparison. Because our students distill from a privileged teacher, they inherit its strategy and are therefore bounded by it; a larger sweep over hands and tasks with continued, curiosity-driven reinforcement learning [9]—using tactile observations as the only sensory channel, or paired with vision—would help separate "what the teacher can be matched on" from "what touch is actually capable of."

4.6 Additional Sensing Modalities: Audio and Temperature

The study above deliberately holds the sensing modality to mechanical contact. The platform, however, also exposes two further channels that fall outside that comparison but broaden what a tactile policy can perceive: the structure- and air-borne *sound* of contact and actuation, and a voxelized *temperature* field. We present them here as proof-of-concept extensions rather than as part of the controlled ablation, both because they exercise the breadth of the unified sensor interface and because they motivate the multisensory directions discussed in the conclusion.

4.6.1 Contact and Actuation Audio

Alongside the contact sensors above, we develop a proof-of-concept audio pipeline in Genesis that synthesizes both the structure-borne sound of contacts and the airborne noise of joint actuation. Procedurally generating sound from simulated physics is well established for interactive games and film, where the goal is perceptual plausibility rather than measurement. We instead propose it as a training signal: a policy that hears its own contacts can infer object properties that the rigid-body solver never resolves. Audio has already been shown to help contact-rich manipulation [42], and this pipeline makes such a signal available directly inside the simulator.

The motivating observation is one of timescale. The acoustic signature of a material lives in its vibration modes, which for stiff objects sit in the kilohertz range, whereas the rigid-body solver integrates at a few hundred hertz. Resolving those vibrations directly

would require shrinking the simulation timestep by orders of magnitude, which is infeasible for RL rollouts. Rather than simulate the vibration, we faithfully approximate it: each physics step emits a short block of audio samples synthesized from the contact state the solver already computes, so the audio carrier sits well above the physics Nyquist frequency while the dynamics step stays cheap. A physics step of duration Δt produces K samples at the sub-step rate $\Delta t_{\text{sub}} = \Delta t / K$, for an audio sample rate $f_s = 1/\Delta t_{\text{sub}}$.

Contact audio. We use source–filter modal synthesis: the material is a bank of damped resonators, and the contact is the source that excites it. Each material is a set of modes $\{(f_i, d_i, g_i)\}$ of center frequency, amplitude decay rate, and output gain. Mode i is a two-pole resonator advanced over the block $k = 1, \ldots, K$,

$$\begin{aligned} y_i[k] &= a_{1,i}\, y_i[k-1] - a_{2,i}\, y_i[k-2] + u_i[k], \\ a_{1,i} &= 2 r_i \cos(2\pi f_i\, \Delta t_{\text{sub}}), \quad a_{2,i} = r_i^2, \quad r_i = e^{-d_i\, \Delta t_{\text{sub}}}. \end{aligned} \tag{4.32}$$

and the emitted sample is the gain-weighted sum $s[k] = \sum_i g_i\, y_i[k]$. The excitation u_i is read from the solver contacts on the sensor link: a sharp rise in the normal contact force injects an impulse that pings the modes into an impact ring-down, while sliding injects a velocity- and force-scaled noise source so a scrape colors the same resonances.

Material modes from modal analysis. The per-material modes are what make different objects sound different, and they follow from the object's geometry and its isotropic elastic material. We assemble the linear-elasticity stiffness K and a lumped mass M over a tetrahedral mesh and solve the generalized eigenproblem

$$K\, \phi_i = \omega_i^2\, M\, \phi_i, \tag{4.33}$$

keeping the non-rigid modes and reading their frequencies $f_i = \omega_i / 2\pi$. Rayleigh damping $C = \alpha M + \beta K$ sets each decay rate $d_i = (\alpha + \beta\, \omega_i^2)/2$, and the gains g_i are the surface normal component of each mode shape ϕ_i. Because K scales with Young's modulus and M with density, the modal spectrum is a fingerprint of the material: this is the cue a policy could use to tell a steel box from a wooden one through contact alone, even when the two are dynamically identical to the rigid solver. The eigensolve is a one-time precompute per object, and hand-tuned material presets are also supported.

Actuation audio. The actuation source models motor and joint noise with one emitter per actuated DOF, driven by the controller effort τ and joint speed ω the solver already exposes. Its loudness tracks the mechanical load and power,

$$\ell = \beta_{\text{load}}\, |\tau| + \beta_{\text{pow}}\, |\tau\, \omega|, \tag{4.34}$$

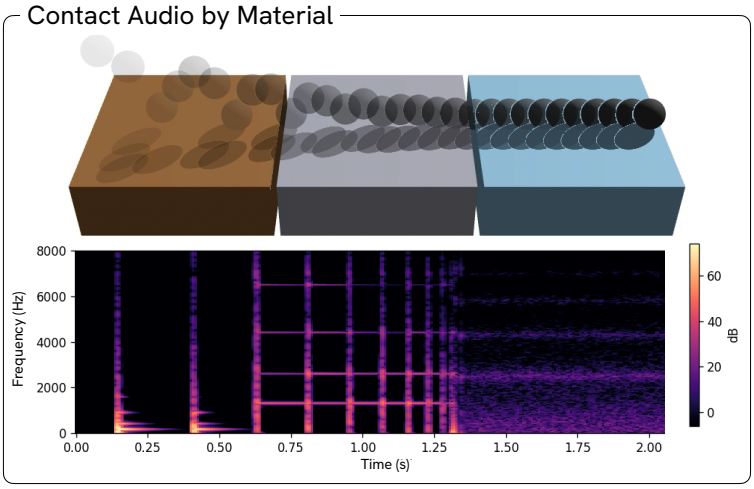


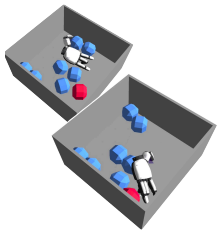
Figure 4.7. **Example contact audio by materials.** A ball bounces and rolls across a wooden, metallic, and glass box. Although physically identical to the rigid-body solver, the three boxes can be distinguished by their modal spectra.

and a velocity-pitched partial bank at fundamental $f_0 = \kappa\,|\omega|$ gives the characteristic whine, layered with velocity-scaled friction noise and an idle hum, synthesized with the same two-pole primitives. A microphone sensor renders what a listener point hears by summing every source over the scene, each attenuated by distance as $1/r^p$ and delayed by r/c for propagation, so contact and actuation sound mix into a single airborne signal.

Toward greater realism. This pipeline is a simplified, real-time model, and several improvements can be made to achieve acoustic realism. Modal synthesis captures the dominant resonances of a struck body, but it omits the frequency- and geometry-dependent radiation efficiency that turns surface vibration into pressure, the room acoustics a real microphone records, and the fine transients of stick–slip friction that our single noise source only coarsely approximates. The actuation source is likewise a parametric approximation of gear-mesh, bearing, and winding noise rather than a measured motor signature. Several extensions could narrow this gap. The dry synthesized signal could be convolved with measured impulse responses of the object and its environment, so it inherits real radiation and reverberation instead of being heard in free field. Per-material and per-motor recordings could replace or augment the procedural banks, either as sample libraries indexed by contact state and joint speed, or as data to fit the modal frequencies, decays, and gains against rather than tuning them by hand. A model trained on paired physics-state and audio recordings could learn this mapping directly. We leave a study of which level of audio fidelity actually benefits policy learning to future work.

4.6.2 Temperature Sensing for Learning Object Discrimination

Temperature can be a useful cue for telling objects apart, but temperature sensors on current robots are mostly used for hardware health monitoring rather than sensing, since the actuators heat up under sustained use. Using our simulated tactile sensors, we explore what temperature sensitivity is needed to distinguish an object only by temperature. We train a policy with proprioception and tactile sensors to find the hot ball out of 8 balls in a bin. Our results in Fig. Figure 4.8 show that high sensitivity is indeed critical, and we fail to succeed on the task with material properties matching real temperature sensors on current robot hands.



conductivity	emissivity 0.5 (steel)		emissivity 0.85 (rubber)	
	heat generation (W/m^2)			
(W/m·K)	500	5000	500	5000
1 (glass)	–	–	–	–
10 (steel)	–	–	–	–
100 (aluminum alloy)	–	–	–	–
150 (aluminum)	✓	✓	–	–

Figure 4.8. **Temperature properties ablation for finding a target hot object.** Checkmark indicates that the hand successfully maintains touch with the hot ball. Emissivity values approximate stainless steel (0.5) and rubber (0.85); conductivities approximate glass (1), stainless steel (10), and aluminum (100). For scale, human skin dissipates on the order of 60 W/m^2 at rest and 100 to 600 W/m^2 during exercise, while small actuators under heavy load can reach 1,000–5,000 W/m^2. The temperature sensor implementation, task, and success metric are detailed in Section 4.6.3.

4.6.3 Temperature Object-Discrimination Experiment

This section details the `TemperatureGrid` sensor model together with the task and success metric for the temperature experiment of Section 4.6.2 (Fig. Figure 4.8).

`TemperatureGrid` Sensor Model.

The `TemperatureGrid` models heat transfer over a voxelized sensor link rather than contact mechanics, returning a temperature field in $°C$. Each sensor link is voxelized into a grid of $n_x \times n_y \times n_z$ cells from its axis-aligned bounding box, with per-axis voxel size $\Delta = \text{extent}/(n_x, n_y, n_z)$. The cell temperatures T are advanced each step by diffusion and an internal source, contact conduction, then radiation and convection, with material properties (conductivity k, density ρ, specific heat c_p, emissivity ε, base temperature) assigned per link. We write ρc_p for the volumetric heat capacity and $\alpha = k/(\rho c_p)$ for the thermal diffusivity.

Diffusion within the grid. We solve the heat equation $\partial T/\partial t = \alpha \nabla^2 T$ with a semi-implicit spectral method. To impose Neumann (zero-flux) boundaries, the grid is mirror-padded to twice its size along each axis before a real-input FFT, so the discrete spectrum carries the even extension consistent with $\partial T/\partial n = 0$ on the walls. With wavenumber vector $\mathbf{k}$ and $k^2 = \|\mathbf{k}\|^2$, each Fourier mode is updated implicitly,

$$\hat{T} \leftarrow \frac{\hat{T}}{1 + \alpha\, \Delta t\, k^2}, \tag{4.35}$$

followed by an inverse FFT and a crop back to the original grid; the zero mode is regularized as $k^2 \leftarrow \max(k^2, \epsilon)$. The update is unconditionally stable, so it does not constrain Δt as an explicit diffusion stencil would.

Internal heat source. An optional volumetric source field q (in $\mathrm{W/m^2}$) is integrated with explicit Euler on the same heat equation,

$$\Delta T = \Delta t\, \frac{q}{\Delta z\, \rho c_p}, \tag{4.36}$$

where dividing by the cell thickness Δz converts the surface flux to a volumetric rate before applying the capacity.

Radiation and convection. Surface voxels, those on any face of the grid, additionally lose heat to the environment at ambient temperature T_{amb}. Radiation follows the Stefan–Boltzmann law for a grey body and convection follows Newton's law of cooling,

$$q_{\mathrm{rad}} = \varepsilon\, \sigma\, (T_K^4 - T_{\mathrm{amb},K}^4), \qquad q_{\mathrm{conv}} = h\, (T - T_{\mathrm{amb}}), \tag{4.37}$$

where $\sigma = 5.670 \times 10^{-8}\ \mathrm{W/(m^2\,K^4)}$, h is the convection coefficient, and $T_K = T + 273.15$ converts to Kelvin for the radiative term. The combined loss is applied as $\Delta T = -\Delta t\, (q_{\mathrm{rad}} + q_{\mathrm{conv}})/(\rho c_p)$.

Contact conduction. Heat crosses a contact interface by Fourier's law $q = -k \nabla T$. With the sensor and the contacting body acting as two thermal resistances in series, the interface uses the harmonic-mean conductivity

$$k_{\mathrm{eff}} = \frac{2\, k_a\, k_b}{k_a + k_b}. \tag{4.38}$$

The conduction length scale is taken as $L = V/A$, one cell's volume over the contact area, which is a heuristic stand-in for the unresolved temperature gradient through the interface. For a contacting voxel at temperature T_{cell} opposite a body at T_{other}, the flux, volumetric rate, and temperature change are

$$\mathrm{flux} = \frac{k_{\mathrm{eff}}\, (T_{\mathrm{other}} - T_{\mathrm{cell}})}{L}, \qquad Q_{\mathrm{vol}} = \frac{\mathrm{flux} \cdot A}{V}, \qquad \Delta T = \Delta t\, \frac{Q_{\mathrm{vol}}}{\rho c_p}. \tag{4.39}$$

Contact area estimation. The contact area A is estimated from the contact-manifold points shared by the two links. We project the points onto their mean contact plane, order them by polar angle about the centroid, and take the polygon area by the shoelace formula. When fewer than three manifold points are available, each contact falls back to a disk of area $\pi\,\delta$, where δ is the penetration depth. In the conduction update the area is additionally floored at $\pi\,d_w\,\delta$, where d_w is a depth-weight scalar (default 1.0), so that deep point contacts still carry a plausible area.

Sensor element lag. The instantaneous cell field T is the physical temperature; the reported measurement T_{meas} trails it through a first-order RC low-pass filter integrated with forward Euler,

$$T_{\text{meas}} \leftarrow T_{\text{meas}} + \frac{\Delta t}{\tau}\,(T - T_{\text{meas}}), \tag{4.40}$$

with sensor time constant τ; $\tau \le 0$ disables the lag and reports T directly. This filter is applied only on the measured readout, modeling the finite thermal response of a physical sensing element, while the ground-truth field is left unfiltered.

Task and Success Metric.

Task setup. The hand rummages a tilted bin (side 0.5 m) holding 8 geometrically identical balls (diameter 8 cm), one of which is the hot target. The bin, the hand, and the seven distractor balls start at ambient $T_{\text{amb}} = 22\,^{\circ}C$, while the target ball starts at $T_{\text{tgt}} = 45\,^{\circ}C$. The policy observes proprioception and one `TemperatureGrid` sensor per finger link; it has no other cue to which ball is hot, so the task is only solvable if temperature differences are resolvable through contact. The hand is reset above the bin and acts at 40 Hz (a control decimation of 5 over the 200 Hz simulation) for episodes of up to 20 s.

Temperature progress and heat latch. For finger sensor i we map its reading to a normalized progress

$$\pi_i = \text{clamp}\left(\frac{T_i - T_{\text{amb}}}{T_{\text{tgt}} - T_{\text{amb}}},\ 0,\ 1\right), \tag{4.41}$$

where T_i is the maximum cell temperature over that sensor's voxels. A single sticky *heat latch* fires the first time the selected-finger progress reaches $1 - m$ with margin $m = 0.12$ (equivalently $T_i \ge 42.2\,^{\circ}C$), and remains on for the rest of the episode. The latch marks the moment the hand has thermally identified the hot ball.

Success metric. We score a run by the latched hot-ball contact time: the cumulative seconds for which the heat latch is on *and* at least one fingertip is in contact with the hot ball,

$$S = \sum_t \Delta t \; \mathbb{I}[\text{latched}(t)] \; \mathbb{I}[\text{fingertip contact with hot ball}(t)] \, . \tag{4.42}$$

A configuration counts as a success (a checkmark in Fig. Figure 4.8) when the trained policy sustains this latched contact, $S > 0$ across evaluation episodes. A material configuration whose temperature signal is too weak for the latch to fire, or for which the hand cannot stay in contact once it fires, scores $S \approx 0$ and is marked as a failure.

4.7 Conclusion

Using a GPU-parallel simulator that exposes many tactile abstractions under one interface, this chapter isolated which tactile signals a dexterous policy needs, finding that sensor placement and whole-hand coverage matter more than the type or resolution of any single sensor, with per-taxel force/torque a robust default that transfers to real hardware.

Two directions follow naturally from this design. The first is to vary spatial resolution and placement more finely, and jointly with the policy: rather than choosing among a few fixed sensors, the simulator can treat taxel density and layout as continuous design variables, searching for the coverage a task actually requires and thereby informing what sensors are worth building. The second is multisensory fusion. The audio and temperature channels above, together with proprioception and vision, give a policy several complementary views of the same contact, and how best to combine them—beyond simple concatenation at the input, toward learned, hierarchical fusion—is a question the unified interface is well suited to study.

Both threads connect back to the rodent-brain study: the question of which tactile representation matters, asked here through task success, was asked in Chapter 3 through alignment with the brain. Chapter 5 draws the two together and lays out where their convergence points next.

Chapter 5

Implications and Future Directions for Tactile Robotics

The two studies examined tactile representations from complementary directions: reverse-engineering the rodent somatosensory cortex (Chapter 3) and forward-engineering robot tactile sensing (Chapter 4). Read side by side, they have much to say to each other, and point toward shared next steps.

5.1 Synthesis Across the Two Works

Although one chapter studies a brain and the other a robot hand, both ask which tactile representations matter for behavior, and both answer it by simulating realistic tactile input at scale. Read side by side, three findings line up.

Force and torque are a privileged abstraction. The rodent-brain study found that convolutional recurrent networks operating on whisker force/torque sequences best explain somatosensory cortex, and the robot study independently found per-taxel force/torque to be the most useful and robust observation type for dexterous policies. That two very different optimization pressures—matching neural activity on one side, maximizing task success on the other—converge on the same low-level abstraction is suggestive evidence that force/torque, rather than raw geometry or vision-like images, is the most informative and biologically aligned representation of touch.

Touch is temporal and recurrent. On the brain side, recurrence was the single most important architectural ingredient, consistent with the fact that whisking and contact are inherently dynamic. On the robot side, the tasks are contact-rich and rapidly changing, and the one task where tactile signals failed to close the gap to the teacher—the fast screwdriver gait—was precisely the one we conjectured needs integration over time. Both point to processing touch as a temporal stream within a sensorimotor loop rather than as a static snapshot, echoing the active, closed-loop character of biological haptics emphasized in Chap-

ter 2.

Coverage beats local fidelity. The biological hand spreads receptors of graded acuity across the whole skin surface and reads them out as a population, and the rodent relies on a distributed whisker array; correspondingly, the robot study found that whole-hand coverage matters more than the fidelity or resolution of any single sensor. Where to place sensing, in both biology and engineering, appears to dominate how precisely each site senses.

These parallels are sharpened by a tension worth stating plainly. The two studies measure "what matters" in incommensurate ways—representational alignment with neural data versus sufficiency for task success—and they differ in regime, with the brain models trained on passive contact against a small fixed stimulus set and the robot policies trained by active, task-driven reinforcement learning. That they nonetheless converge on force/torque, recurrence, and coverage is a form of mutual validation: each supplies an existence proof the other lacks. The shared methodological lesson is that biomechanically grounded, scalable simulation of tactile input is what made either result obtainable.

5.2 Limitations

Each study carries constraints that bound these conclusions. The rodent-brain work is limited most by the neural data: the recordings span only six distinct stimulus conditions in a single cortical area, which caps the explainable variability and likely depresses the inter-animal consistency ceiling against which models are judged. The models are trained on passive contact yet evaluated on active whisking, and are compared to the brain through representational similarity rather than by predicting the full temporal dynamics of neural responses. The robot work is bounded above all by its teacher–student design: because students distill from a privileged teacher, they inherit its strategy and cannot reveal what touch *alone*, free to discover new behavior, might achieve. Its contact and noise models are hand-tuned to plausible ranges rather than calibrated to a specific real sensor, the sweep covers a limited set of tasks and hands, and sim-to-real transfer is validated only briefly. Most fundamentally, the platform extracts tactile signals from rigid-body contact queries, so it does not model the skin *folding* around an object—the very deformation that, as Chapter 2 noted, the slowly-adapting afferents of the biological hand rely on to read out local geometry.

5.3 Future Directions

The two threads suggest a shared agenda in which each tool improves the other.

Closing the loop between brain and robot. Most directly, the scalable robot simulator—with its deformable, thermal, and contact-audio sensors—can generate far more

diverse tactile stimuli than any in-vivo experiment, which could be used to design richer neural datasets and lift the stimulus-diversity ceiling that currently limits the brain modeling. Conversely, the recurrent EAD encoders that best matched cortex are natural candidates for the tactile front-end of a robot policy, testing whether biologically aligned representations transfer to better manipulation. Bridging the two more tightly, one could run the classic *exploratory procedures* of human haptics—lateral motion for texture, static pressure for compliance, sifting granular media and fluids, which the underlying physics engine simulates well—through the simulated hand and feed their outputs into the neural models, connecting ethological exploration to both dexterity and cortical modeling.

Co-evolving tactile hardware and software. A recurring theme across both studies is that the sensing surface and the algorithm that consumes it are best studied together. By making sensor type, placement, and resolution *design variables* rather than fixed givens, the simulation platform can be used not only to choose among existing sensors but to inform *what sensors to build*, co-designing the hardware and the policy in a single loop. Realizing this in the real world hinges on tightening the sim-to-real gap, which is best done hand-in-hand with the people building the sensors. Working directly with tactile-sensor hardware developers—calibrating the simulated noise, hysteresis, and deformation models against measurements from their devices—would let the platform serve as a faithful design tool, so that the same experiment that decides which abstraction a policy needs can also feed back into the next generation of sensorized hands.

Multisensory fusion beyond concatenation. Both studies deliberately treat touch largely in isolation, yet biological perception is multisensory, and the haptic system itself already fuses cutaneous, kinesthetic (proprioceptive), and motor streams (Chapter 2). A natural and important next direction is therefore to integrate tactile sensing with proprioception, vision, language, and even the contact audio the simulator can synthesize, and to ask *where* and *how* that fusion should occur. Current multimodal touch models typically join the modalities only at a final embedding layer [70, 71]; inspired by the brain's pervasive feedback across intermediate processing stages, one could instead build architectures with hierarchical fusion operations—cross-attention, co-attention, and rerouting—endowed with the recurrence and long-range feedback that proved decisive in the rodent-brain study. We hypothesize that such models would be more sample-efficient and would generalize better to out-of-distribution objects than late-fusion or unimodal baselines. They could be evaluated precisely on the tasks where vision alone struggles—locating an occluded object in a pile, handling soft or slippery objects, and precise assembly such as cable insertion—both quantitatively on task success and qualitatively on whether a model anticipates, for instance, that supporting the base of a slippery object eases the grasp. Analyzing the error patterns of

vision-only, tactile-only, and fused models would reveal whether multimodal touch inherits the strengths of each channel without their weaknesses, and whether it begins to approach brain-like one- or few-shot efficiency.

Lower-level extensions. Several smaller steps follow from the limitations above: moving beyond rigid-body contact toward finite-element, deformable skin to capture the folding signal that biological slowly-adapting afferents rely on; jointly training sensing and action so that touch is genuinely active rather than a passive readout; and incorporating temperature with the asymmetric, protective dynamic ranges observed in biological thermoreception.

Pursued together, these directions point toward tactile systems—biological and artificial—understood and built on a common, simulation-grounded foundation.

5.4 Broader Impacts

The questions pursued here reach beyond robot learning. *For industry*, many contact-rich, high-precision tasks still depend on human labor; robots equipped with capable visuo-tactile models could extend autonomy into settings such as healthcare and manufacturing where vision alone is insufficient. *For neuroscience*, the rodent-brain study offers a first quantitative characterization of the inductive biases a model needs to match somatosensory cortex—a system far less understood than visual cortex—and a template for asking the same of other tactile areas. *For human–robot interaction*, aligning a robot's *internal* representations of object properties with our own, rather than only its outward behavior, is a step toward machines that handle objects in a natural, predictable, and safe way—a prerequisite for robots operating alongside people in homes and workplaces. *For the research community*, the platform and models provide reusable baselines for tactile sensing across embodiments, lowering the barrier to incorporating touch into multimodal models and helping populate the still-underexplored intersection of computational neuroscience and embodied AI. As with any advance in autonomy, these benefits carry responsibilities—most notably the potential disruption of manual-labor employment—that call for thoughtful deployment and proactive workforce and policy planning.

Bibliography

[1] Chris S. Bresee, Hayley M. Belli, Yifu Luo, and Mitra J. Z. Hartmann. Comparative morphology of the whiskers and faces of mice (mus musculus) and rats (rattus norvegicus). *Journal of Experimental Biology*, 226(19):jeb245597, 10 2023. ISSN 0022-0949. doi: 10.1242/jeb.245597. URL https://doi.org/10.1242/jeb.245597.

[2] Nadina O Zweifel, Nicholas E Bush, Ian Abraham, Todd D Murphey, and Mitra JZ Hartmann. A dynamical model for generating synthetic data to quantify active tactile sensing behavior in the rat. *Proceedings of the National Academy of Sciences*, 118(27): e2011905118, 2021.

[3] Chris C Rodgers. A detailed behavioral, videographic, and neural dataset on object recognition in mice. *Scientific Data*, 9(1):620, 2022.

[4] Yongqiang Zhao, Kun Qian, Boyi Duan, and Shan Luo. FOTS: A fast optical tactile simulator for sim2real learning of tactile-motor robot manipulation skills, 2024. URL http://arxiv.org/abs/2404.19217.

[5] An Dang, Jayjun Lee, Mustafa Mukadam, X. Alice Wu, Bernadette Bucher, Manikantan Nambi, and Nima Fazeli. HydroShear: Hydroelastic shear simulation for tactile sim-to-real reinforcement learning, 2026. URL https://arxiv.org/abs/2603.00446.

[6] Lei Su, Zhijie Peng, Renyuan Ren, Shengping Mao, Juan Du, Kaifeng Zhang, and Xuezhou Zhu. Tacmap: Bridging the tactile sim-to-real gap via geometry-consistent penetration depth map, 2026. URL http://arxiv.org/abs/2602.21625.

[7] Jan 2026. URL https://www.sharpa.com/blogs/news/sharpa-unveils-its-first-autonomous-full-body-robot-with-human-dexterity-at-ces-2026.

[8] Iretiayo Akinola, Jie Xu, Jan Carius, Dieter Fox, and Yashraj Narang. Tacsl: A library for visuotactile sensor simulation and learning. *IEEE Transactions on Robotics*, 2025.

[9] Yuri Burda, Harrison Edwards, Amos Storkey, and Oleg Klimov. Exploration by random network distillation, 2018. URL https://arxiv.org/abs/1810.12894.

[10] Clemens Schwarke, Victor Klemm, Matthijs van der Boon, Marko van der Bjelonic, and Marco Hutter. Curiosity-driven learning of joint locomotion and manipulation tasks. In *Proceedings of the 7th Conference on Robot Learning*, pages 2594–2610, 2023. URL https://proceedings.mlr.press/v229/schwarke23a.html.

[11] Stephane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In Geoffrey Gordon, David Dunson, and Miroslav Dudík, editors, *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pages 627–635, Fort Lauderdale, FL, USA, 11–13 Apr 2011. PMLR. URL https://proceedings.mlr.press/v15/ross11a.html.

[12] Erwin Coumans and Yunfei Bai. Pybullet, a python module for physics simulation for games, robotics and machine learning. http://pybullet.org, 2016–2021.

[13] Susan J Lederman and Roberta L Klatzky. Haptic perception: A tutorial. *Attention, Perception, & Psychophysics*, 71(7):1439–1459, 2009.

[14] Roberta L. Klatzky. Haptic perception and its relation to action. *Annual Review of Psychology*, 76, 2024.

[15] Ewa Jarocka, J. Andrew Pruszynski, and Roland S. Johansson. Human touch receptors are sensitive to spatial details on the scale of single fingerprint ridges. *Journal of Neuroscience*, 41(16):3622–3634, 2021.

[16] J. Andrew Pruszynski, J. Randall Flanagan, and Roland S. Johansson. Fast and accurate edge orientation processing during object manipulation. *eLife*, 7:e31200, 2018.

[17] Hannes P. Saal, Benoît P. Delhaye, Brendan C. Rayhaun, and Sliman J. Bensmaia. Simulating tactile signals from the whole hand with millisecond precision. *Proceedings of the National Academy of Sciences*, 114(28):E5693–E5702, 2017.

[18] 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, March 2024. URL http://arxiv.org/abs/2403.08716. arXiv:2403.08716 [cs].

[19] Hsin-Ni Ho and Lynette A. Jones. Contribution of thermal cues to material discrimination and localization. *Perception & Psychophysics*, 68(1):118–128, 2006.

[20] Eric L. Amazeen and Michael T. Turvey. Weight perception and the haptic size–weight illusion are functions of the inertia tensor. *Journal of Experimental Psychology: Human Perception and Performance*, 22(1):213–232, 1996.

[21] Müge Cavdan, Katja Doerschner, and Knut Drewing. Task and material properties interactively affect softness explorations along different dimensions. *IEEE Transactions on Haptics*, 14(3):603–613, 2021.

[22] Dicle N. Dövencioğlu, F. Sena Üstün, Katja Doerschner, and Knut Drewing. Hand explorations are determined by the characteristics of the perceptual space of real-world materials from silk to sand. *Scientific Reports*, 12(1):14785, 2022.

[23] Jonathan Cheung, Phillip Maire, Jinho Kim, Jonathan Sy, and Samuel Andrew Hires. The sensorimotor basis of whisker-guided anteroposterior object localization in head-fixed mice. *Current Biology*, 29(18):3029–3040, 2019.

[24] Nicholas J Sofroniew and Karel Svoboda. Whisking. *Current Biology*, 25(4): R137–R140, 2015.

[25] Daniel J Felleman and David C Van Essen. Distributed hierarchical processing in the primate cerebral cortex. *Cerebral cortex (New York, NY: 1991)*, 1(1):1–47, 1991.

[26] Laurens WJ Bosman, Arthur R Houweling, Cullen B Owens, Nouk Tanke, Olesya T Shevchouk, Negah Rahmati, Wouter HT Teunissen, Chiheng Ju, Wei Gong, Sebastiaan KE Koekkoek, et al. Anatomical pathways involved in generating and sensing rhythmic whisker movements. *Frontiers in integrative neuroscience*, 5:53, 2011.

[27] Jeffrey D Moore, Nicole Mercer Lindsay, Martin Deschênes, and David Kleinfeld. Vibrissa self-motion and touch are reliably encoded along the same somatosensory pathway from brainstem through thalamus. *PLoS biology*, 13(9):e1002253, 2015.

[28] Jochen F Staiger and Carl CH Petersen. Neuronal circuits in barrel cortex for whisker sensory perception. *Physiological reviews*, 101(1):353–415, 2021.

[29] Ravinder S Dahiya, Giorgio Metta, Maurizio Valle, and Giulio Sandini. Tactile sensing—from humans to humanoids. *IEEE transactions on robotics*, 26(1):1–20, 2009.

[30] Kazuhiro Shimonomura. Tactile image sensors employing camera: A review. *Sensors*, 19(18):3933, 2019.

[31] Jiaqi Li, Jie Li, Zhenmin Ding, Shihao Chen, Kai Yang, Lei Ren, Yan Liu, and Luquan Ren. Flexible tactile sensors toward intelligent perception: Mechanisms, architectures and functional applications. *Chemical Engineering Journal*, 541:177827, August 2026. ISSN 13858947. doi: 10.1016/j.cej.2026.177827. URL https://linkinghub.elsevier.com/retrieve/pii/S1385894726052885.

[32] Yudong Cao, Jiacheng Li, Zihao Dong, Tianyu Sheng, Deyuan Zhang, Jun Cai, and Yonggang Jiang. Flexible tactile sensor with an embedded-hair-in-elastomer structure for normal and shear stress sensing. *Soft Science*, 3(4):N/A–N/A, October 2023. ISSN ISSN 2769-5441 (Online). doi: 10.20517/ss.2023.22. URL https://www.oaepublish.com/articles/ss.2023.22.

[33] Fei Fei, Zhenkun Jia, Changcheng Wu, Xiong Lu, and Zhi Li. Design of a Capacitive Tactile Sensor Array System for Human–Computer Interaction. *Sensors (Basel, Switzerland)*, 24(20):6629, October 2024. ISSN 1424-8220. doi: 10.3390/s24206629. URL https://pmc.ncbi.nlm.nih.gov/articles/PMC11511034/.

[34] Naijia Xu, Shaoyu Liu, Daohui Zhang, Xin Fu, Baisong Han, Xudong Hou, and Xingang Zhao. Three-Dimensional Flexible Tactile Sensor Arrays for Perceptual Grasping: Toward Enhanced Perceptual Robotics. *IEEE Sensors Journal*, 26(10): 14349–14361, May 2026. ISSN 1558-1748. doi: 10.1109/JSEN.2025.3559343. URL https://ieeexplore.ieee.org/document/10965915.

[35] Yi Song, Junwei Wang, Zongke Li, Weilang Hu, Ye Qiu, Ye Tian, Pei Zhao, Aiping Liu, and Huaping Wu. Fingertip-scale six-axis tactile interface with high-precision force sensing and position localization for dexterous human–machine interactions. *Microsystems & Nanoengineering*, 12(1):193, 2026.

[36] Zhuowei Xu, Zilin Si, Kevin Zhang, Oliver Kroemer, and Zeynep Temel. A multimodal tactile fingertip design for robotic hands to enhance dexterous manipulation, 2025. URL https://arxiv.org/abs/2510.05382.

[37] Wenzhen Yuan, Siyuan Dong, and Edward H Adelson. Gelsight: High-resolution robot tactile sensors for estimating geometry and force. *Sensors*, 17(12):2762, 2017.

[38] Raunaq Bhirangi, Tess Hellebrekers, Carmel Majidi, and Abhinav Gupta. Reskin: versatile, replaceable, lasting tactile skins. *arXiv preprint arXiv:2111.00071*, 2021.

[39] Dounia Kitouni, Elie Chelly, Mahdi Khoramshahi, and Veronique Perdereau. Fingertip contact force direction control using tactile feedback, 2024. URL http://arxiv.org/abs/2406.11545.

[40] Elie Chelly, Andrea Cherubini, Philippe Fraisse, Faiz Ben Amar, and Mahdi Khoramshahi. Tactile-based force estimation for interaction control with robot fingers, 2025. URL http://arxiv.org/abs/2411.13335.

[41] 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, 2025. URL http://arxiv.org/abs/2506.14754.

[42] 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.

[43] Depeng Kong, Yuyao Lu, Shuyao Zhou, Mengke Wang, Gaoyang Pang, Baocheng Wang, Lipeng Chen, Xiaoyan Huang, Honghao Lyu, Kaichen Xu, and Geng Yang. Super-resolution tactile sensor arrays with sparse units enabled by deep learning. *Science Advances*, 11(27):eadv2124, July 2025. doi: 10.1126/sciadv.adv2124. URL https://www.science.org/doi/10.1126/sciadv.adv2124.

[44] Yuyang Li, Wenxin Du, Chang Yu, Puhao Li, Zihang Zhao, Tengyu Liu, Chenfanfu Jiang, Yixin Zhu, and Siyuan Huang. Taccel: Scaling up vision-based tactile robotics via high-performance gpu simulation, 2025. URL http://arxiv.org/abs/2504.12908.

[45] 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. In *2025 IEEE International Conference on Robotics and Automation*, pages 5850–5856, 2025. doi: 10.1109/ICRA55743.2025.11127974.

[46] Berith Atemoztli De la Cruz Sanchez, Jennifer Kwiatkowski, and Jean-Philippe Roberge. Tactile contact patterns for robotic grasping: A dataset of real and simulated data. In *2025 3rd International Conference on Control and Robot Technology*, pages 8–13, 2025. doi: 10.1109/ICCRT63554.2025.11072742.

[47] Daniel LK Yamins, Ha Hong, Charles F Cadieu, Ethan A Solomon, Darren Seibert, and James J DiCarlo. Performance-optimized hierarchical models predict neural responses in higher visual cortex. 111(23):8619–8624.

[48] Seyed-Mahdi Khaligh-Razavi and Nikolaus Kriegeskorte. Deep supervised, but not unsupervised, models may explain it cortical representation. *PLoS computational biology*, 10(11):e1003915, 2014.

[49] Alexander JE Kell*, Daniel LK Yamins*, Erica N Shook, Sam V Norman-Haignere, and Josh H McDermott. A task-optimized neural network replicates human auditory behavior, predicts brain responses, and reveals a cortical processing hierarchy. *Neuron*, 98(3):630–644, 2018.

[50] Martin Schrimpf, Idan Asher Blank, Greta Tuckute, Carina Kauf, Eghbal A Hosseini, Nancy Kanwisher, Joshua B Tenenbaum, and Evelina Fedorenko. The neural architecture of language: Integrative modeling converges on predictive processing. *Proceedings of the National Academy of Sciences*, 118(45):e2105646118, 2021.

[51] Aran Nayebi*, Nathan CL Kong*, Chengxu Zhuang, Justin L Gardner, Anthony M Norcia, and Daniel LK Yamins. Mouse visual cortex as a limited resource system that self-learns an ecologically-general representation. *PLOS Computational Biology*, 19, 2023.

[52] Aran Nayebi, Nathan C. L. Kong, Chengxu Zhuang, Justin L. Gardner, Anthony M. Norcia, and Daniel L. K. Yamins. Mouse visual cortex as a limited resource system that self-learns an ecologically-general representation. *PLOS Computational Biology*, 19(10):e1011506, October 2023. ISSN 1553-7358. doi: 10.1371/journal.pcbi.1011506. URL https://journals.plos.org/ploscompbiol/article?id=10.1371/journal.pcbi.1011506.

[53] Aran Nayebi, Rishi Rajalingham, Mehrdad Jazayeri, and Guangyu Robert Yang. Neural Foundations of Mental Simulation: Future Prediction of Latent Representations on Dynamic Scenes. *ArXiv*, page arXiv:2305.11772v2, October 2023. ISSN 2331-8422. URL https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10246064/.

[54] Chengxu Zhuang, Jonas Kubilius, Mitra J Hartmann, and Daniel L Yamins. Toward goal-driven neural network models for the rodent whisker-trigeminal system. *Advances in Neural Information Processing Systems*, 30, 2017.

[55] Aran Nayebi*, Daniel Bear*, Jonas Kubilius*, Kohitij Kar, Surya Ganguli, David Sussillo, James J DiCarlo, and Daniel L Yamins. Task-driven convolutional recurrent models of the visual system. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[56] Albert Gu, Karan Goel, and Christopher Ré. Efficiently modeling long sequences with structured state spaces. *arXiv preprint arXiv:2111.00396*, 2021.

[57] Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces. *arXiv preprint arXiv:2312.00752*, 2023.

[58] Siddhant Haldar, Zhuoran Peng, and Lerrel Pinto. Baku: An efficient transformer for multi-task policy learning. *arXiv preprint arXiv:2406.07539*, 2024.

[59] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PmLR, 2020.

[60] 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.

[61] Bruno A Olshausen and David J Field. Emergence of simple-cell receptive field properties by learning a sparse code for natural images. *Nature*, 381(6583):607–609, 1996.

[62] Chengxu Zhuang, Siming Yan, Aran Nayebi, Martin Schrimpf, Michael C Frank, James J DiCarlo, and Daniel LK Yamins. Unsupervised neural network models of the ventral visual stream. *Proceedings of the National Academy of Sciences*, 118(3), 2021.

[63] Elle Miller, Trevor McInroe, David Abel, Oisin Mac Aodha, and Sethu Vijayakumar. Enhancing tactile-based reinforcement learning for robotic control. In *OpenReview*, 2025. URL https://openreview.net/forum?id=Toy96yYopR.

[64] Irmak Guzey, Ben Evans, Soumith Chintala, and Lerrel Pinto. Dexterity from Touch: Self-Supervised Pre-Training of Tactile Representations with Robotic Play, March 2023. URL http://arxiv.org/abs/2303.12076. arXiv:2303.12076 [cs].

[65] 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, October 2024. URL http://arxiv.org/abs/2410.24090. arXiv:2410.24090 [cs.RO].

[66] 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, May 2025. URL http://arxiv.org/abs/2505.11420. arXiv:2505.11420 [cs.RO].

[67] Youngsun Wi, Jessica Yin, Elvis Xiang, Akash Sharma, Jitendra Malik, Mustafa Mukadam, Nima Fazeli, and Tess Hellebrekers. TactAlign: Human-to-Robot Policy Transfer via Tactile Alignment, February 2026. URL http://arxiv.org/abs/2602.13579. arXiv:2602.13579 [cs.RO].

[68] 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, February 2026. URL http://arxiv.org/abs/2511.09484. arXiv:2511.09484 [cs.RO].

[69] Zhao-Heng Yin, Changhao Wang, Luis Pineda, Francois Hogan, Krishna Bodduluri, Akash Sharma, Patrick Lancaster, Ishita Prasad, Mrinal Kalakrishnan, Jitendra Malik, Mike Lambeta, Tingfan Wu, Pieter Abbeel, and Mustafa Mukadam. DexterityGen: Foundation Controller for Unprecedented Dexterity, February 2025. URL http://arxiv.org/abs/2502.04307. arXiv:2502.04307 [cs.RO].

[70] 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. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 26340–26353, 2024.

[71] Rohit Girdhar, Alaaeldin El-Nouby, Zhuang Liu, Mannat Singh, Kalyan Vasudev Alwala, Armand Joulin, and Ishan Misra. ImageBind: One Embedding Space To Bind Them All, May 2023. URL http://arxiv.org/abs/2305.05665. arXiv:2305.05665 [cs].

[72] Hui Zhang, Zijian Wu, Linyi Huang, Sammy Christen, and Jie Song. RobustDexGrasp: Robust dexterous grasping of general objects, 2025. URL http://arxiv.org/abs/2504.05287.

[73] Pei Lin, Yuzhe Huang, Wanlin Li, Jianpeng Ma, Chenxi Xiao, and Ziyuan Jiao. PP-Tac: Paper picking using tactile feedback in dexterous robotic hands, 2025. URL http://arxiv.org/abs/2504.16649.

[74] Robyn A Grant. Can we study whisker movements to gain insights into the natural sensory behaviours of mammals? *The Journal of Physiology*, 2025.

[75] Alwynelle S Ahl. The role of vibrissae in behavior: a status review. *Veterinary research communications*, 10(1):245–268, 1986.

[76] Guido Dehnhardt, Bjorn Mauck, Wolf Hanke, and Horst Bleckmann. Hydrodynamic trail-following in harbor seals (phoca vitulina). *Science*, 293(5527):102–104, 2001.

[77] Susanne Sterbing-D'Angelo, Mohit Chadha, Chen Chiu, Ben Falk, Wei Xian, Janna Barcelo, John M Zook, and Cynthia F Moss. Bat wing sensors support flight control. *Proceedings of the National Academy of Sciences*, 108(27):11291–11296, 2011.

[78] Samuel Andrew Hires, Diego A Gutnisky, Jianing Yu, Daniel H O'Connor, and Karel Svoboda. Low-noise encoding of active touch by layer 4 in the somatosensory cortex. *elife*, 4:e06619, 2015.

[79] Nicolás Navarro-Guerrero, Sibel Toprak, Josip Josifovski, and Lorenzo Jamone. Visuo-haptic object perception for robots: an overview. *Autonomous Robots*, 47(4):377–403, 2023.

[80] Martin J Pearson, Ben Mitchinson, J Charles Sullivan, Anthony G Pipe, and Tony J Prescott. Biomimetic vibrissal sensing for robots. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 366(1581):3085–3096, 2011.

[81] Tareq Assaf, Emma D Wilson, Sean Anderson, Paul Dean, John Porrill, and Martin J Pearson. Visual-tactile sensory map calibration of a biomimetic whiskered robot. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 967–972. IEEE, 2016.

[82] Nathaniel Simon, Allen Z Ren, Alexander Piqué, David Snyder, Daphne Barretto, Marcus Hultmark, and Anirudha Majumdar. Flowdrone: wind estimation and gust rejection on uavs using fast-response hot-wire flow sensors. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5393–5399. IEEE, 2023.

[83] Teresa A Kent, Hannah Emnett, Mahnoush Babaei, Mitra JZ Hartmann, and Sarah Bergbreiter. Identifying contact distance uncertainty in whisker sensing with tapered, flexible whiskers. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 607–613. IEEE, 2023.

[84] Teresa A Kent and Sarah Bergbreiter. Flow shadowing: A method to detect multiple flow headings using an array of densely packed whisker-inspired sensors. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 17843–17849. IEEE, 2024.

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

[86] 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.

[87] 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. *arXiv preprint arXiv:2402.10211*, 2024.

[88] Michael Armstrong-James, KEVIN Fox, and Ashis Das-Gupta. Flow of excitation within rat barrel cortex on striking a single vibrissa. *Journal of neurophysiology*, 68 (4):1345–1358, 1992.

[89] Jason ND Kerr, Christiaan PJ De Kock, David S Greenberg, Randy M Bruno, Bert Sakmann, and Fritjof Helmchen. Spatial organization of neuronal population responses in layer 2/3 of rat barrel cortex. *Journal of neuroscience*, 27(48):13316–13328, 2007.

[90] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015.

[91] Hayley M Belli, Anne ET Yang, Chris S Bresee, and Mitra JZ Hartmann. Variations in vibrissal geometry across the rat mystacial pad: base diameter, medulla, and taper. *Journal of Neurophysiology*, 117(4):1807–1820, 2017.

[92] Hayley M Belli, Chris S Bresee, Matthew M Graff, and Mitra JZ Hartmann. Quantifying the three-dimensional facial morphology of the laboratory rat with a focus on the vibrissae. *PLoS One*, 13(4):e0194981, 2018.

[93] Per Magne Knutsen, Armin Biess, and Ehud Ahissar. Vibrissal kinematics in 3d: tight coupling of azimuth, elevation, and torsion across different whisking modes. *Neuron*, 59(1):35–42, 2008.

[94] Patricia W Freeman and Cliff A Lemen. A simple morphological predictor of bite force in rodents. *Journal of Zoology*, 275(4):418–422, 2008.

[95] Rosa Cao and Daniel Yamins. Explanatory models in neuroscience, part 2: Functional intelligibility and the contravariance principle. *Cognitive Systems Research*, 85:101200, 2024.

[96] Khaled Mamou and Faouzi Ghorbel. A simple and efficient approach for 3d mesh approximate convex decomposition. pages 3501–3504, 11 2009. doi: 10.1109/ICIP.2009.5414068.

[97] Aran Nayebi, Javier Sagastuy-Brena, Daniel M Bear, Kohitij Kar, Jonas Kubilius, Surya Ganguli, David Sussillo, James J DiCarlo, and Daniel LK Yamins. Recurrent connections in the primate ventral visual stream mediate a tradeoff between task performance and network size during core object recognition. *Neural Computation*, 34:1652–1675, 2022.

[98] 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.

[99] Jasmine Collins, Jascha Sohl-Dickstein, and David Sussillo. Capacity and trainability in recurrent neural networks. In *ICLR*, 2017.

[100] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

[101] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation, 2014. URL https://arxiv.org/abs/1406.1078.

[102] Courtney J. Spoerer, Patrick McClure, and Nikolaus Kriegeskorte. Recurrent convolutional neural networks: a better model of biological object recognition. *Front. Psychol.*, 8:1–14, 2017.

[103] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization, 2016. URL https://arxiv.org/abs/1607.06450.

[104] Léon Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010: 19th International Conference on Computational StatisticsParis France, August 22-27, 2010 Keynote, Invited and Contributed Papers*, pages 177–186. Springer, 2010.

[105] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

[106] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization, 2019. URL https://arxiv.org/abs/1711.05101.

[107] Yang You, Igor Gitman, and Boris Ginsburg. Large batch training of convolutional networks.

[108] Nikolaus Kriegeskorte, Marieke Mur, and Peter A Bandettini. Representational similarity analysis-connecting the branches of systems neuroscience. *Frontiers in systems neuroscience*, 2:249, 2008.

[109] Jenelle Feather*, Meenakshi Khosla*, N Murty*, and Aran Nayebi*. Brain-model evaluations need the neuroai turing test. *arXiv preprint arXiv:2502.16238*, 2025.

[110] Martin Schrimpf, Jonas Kubilius, Michael J Lee, N Apurva Ratan Murty, Robert Ajemian, and James J DiCarlo. Integrative benchmarking to advance neurally mechanistic models of human intelligence. *Neuron*, 108(3):413–423, 2020.

[111] Raunaq Bhirangi, Tess Hellebrekers, Carmel Majidi, and Abhinav Gupta. ReSkin: Versatile, replaceable, lasting tactile skins. In *Proceedings of the Conference on Robot Learning*, November 2021. URL https://arxiv.org/abs/2111.00071.

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

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

[114] Genesis AI Team. The role of simulation in scalable robotics, genesis world 1.0, and the path forward. *Genesis AI Blog*, May 2026. URL https://www.genesis.ai/blog/the-role-of-simulation-in-scalable-robotics-genesis-world-10-and-the-path-forward.