

Low-Cost Multimodal Sensing and Dexterity for Deformable Object Manipulation

Kevin Zhang

CMU-RI-TR-25-08

February 2025

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

Thesis Committee:

Prof. Oliver Kroemer, *Co-Chair*
Prof. Manuela M. Veloso, *Co-Chair*
Prof. F. Zeynep Temel
Prof. Tapomayukh Bhattacharjee (Cornell University)

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

Copyright © 2025 Kevin Zhang

Abstract

To integrate robots seamlessly into daily life, they must be able to handle a variety of tasks in diverse environments, like cooking in restaurants or tidying up around the house. Many of the items in these environments are deformable such as fruits or bed sheets and a certain level of dexterity is necessary to accomplish complex tasks such as making breakfast or folding clothes. However, existing industrial robotic grippers have limitations in terms of sensing and dexterity, and anthropomorphic hands are expensive and challenging to control. This thesis aims to address the challenges of dexterously manipulating deformable objects by presenting work on low-cost multimodal sensing and a novel end-effector design with high dexterity but simple control.

The first part of this thesis focuses on low-cost multimodal sensing. First, we will discuss the abilities of vibrotactile sensing to distinguish material properties between different food items in a self-supervised manner. Afterwards, we introduce work on characterizing the performance of vibrotactile sensing in a factory environment where there are tight tolerances when inserting industrial connectors. Finally, we demonstrate how magnetic sensing can help a robot localize different items such as a key and sense forces less than 10N when manipulating deformable tools such as a sauce dispensing bottle.

In the second part of this thesis, I will introduce our work on utilizing mini-delta robots as robotic fingers that are both dexterous, and simple to control. With three translational degrees of freedom each, we were able to teleoperate a two fingered DeltaHand to perform various dexterous tasks like card picking, grape plucking, and dough rolling. We also developed a more compact DeltaHand with four fingers where the additional fingers help stabilize objects during in-hand tasks and enable the hand to fold cloth, unscrew a cap, and straighten a cable.

Finally, to bring both parts of the thesis together, I will present work on utilizing different sensors to learn dexterous closed loop skills on a DeltaHand using various multimodal skills on a variety of tasks with fine-grained demonstrations. In addition, we augment a parallel jaw gripper with a Delta finger in order to perform tasks that require both force from the parallel jaw fingers as well as dexterity from the Delta finger to succeed such as picking up a soft tortilla, removing a stick of gum from a pack, and lifting up the cap of a vitamin bottle. In summary, this thesis seeks to advance robotic capabilities in handling diverse objects and tasks through innovative sensing methods, dexterous manipulation, and improved control policies.

Acknowledgements

I would like to first thank both of my co-advisors, Professor Oliver Kroemer and Professor Manuela Veloso, for continuously providing me with guidance for my research as well as the opportunity to continue growing my research skills over the years. I first met Manuela as a cheeky Junior in March of 2017 wanting to put some arms on CoBot for RoboCup at Home. Later that day, I ended up joining the RoboCup at Home team and had the privilege to work with Pepper for 2 years. On the other hand I met Oliver for the first time in an elevator helping his Masters student move items. He was not hesitant to help his students with whatever they needed and I have witnessed his dedication to his students continue to this date. Oliver has provided me with extensive insights for my research projects and given me freedom to pursue whatever projects that I was interested in. He has also never hesitated to sit down with me in order to help me finish whatever tasks I needed help with. Without Oliver's or Manuela's input and support, my thesis would not have been possible. In addition, I would like to thank F. Zeynep Temel for valuable feedback in the Delta projects and semi-adopting me into her lab. I have been able to leverage Zoom Lab's resources to solder and 3D print during emergencies when Tech Spark is closed. I would also like to thank Tapomyukh Bhattacharjee for being on my Thesis Committee and chatting with me during various conferences and workshops on food manipulation. I greatly appreciate his kindness and patience with me throughout my Ph.D thesis.

Next, I would like to thank my labmates and friends for helping me through the Ph.D program. In particular, I would like to thank my labmates in both the IAM (Intelligent Autonomous Manipulation) Lab and CORAL (Cooperate, Observe, Reason, Act, and Learn) Group for the lively discussions about research that helped me formulate ideas for my research. In particular, I would like to thank all of my collaborators and co-authors: Ami Sawhney, Steven Lee, Christopher Chang, Shobhit Aggarwal, Tess Hellebrekers, Pragna Mannam, Avi Rudich, Mark Lee, Zilin Si, and Erin Zhang for collaborating with me and making contributions that have been instrumental in the completion of this doctoral thesis proposal. I would also like to thank Mohit Sharma and Jacky Liang for creating Frankapy and Franka-interface with me. Mohit Sharma has been a lifelong friend and addition in my life since our first meeting. He has unlimited knowledge about the latest research papers and experience from life. I don't know what I would have done if I hadn't met him. We would always grab dinner together with Tanya Marwah when we were tired and that would provide a great rest break that I will always cherish. Jacky Liang has also been an invaluable friend for me as well as my greatest long-term roommate. We have had so many adventures over the years from conferences to road trips and random hikes. We spent so much time together during the covid years from cooking dinners and watching movies, and I was happy to spend the time with you. In addition, I would like to thank Sarvesh Patil for always being cheerful, ordering food and snacks for the lab, bringing new ideas to the lab, and helping me babysit Lychee on vacations. Zilin Si has been one of the best collaborators I have had with her assistance being invaluable in all of my later papers. She is always on top of things and so organized while also never saying that things can't be done. In addition, I would like to thank Saumya Saxena for livening up the social scene in the IAM lab and also with helping me clean it up. I would like to thank Mark Lee for accompanying me in the fields and helping keep the lab culture fun. I would also like to thank Shivam Vats for always popping into the lab at snack times to just chat about the latest research or life. I would like to thank Pat Callaghan for always bringing laughter into the IAM Lab and sharing some good

tunes with me. Thank you to Tabitha Lee, Alex LaGrassa, Yunus Seker, Jialiang Zhao, Wuming Zhang, Maximilian Sieb, Austin Wang, Xinyu Wang, Janice Lee, and Lawrence Feng for chatting with me sometimes in the lab about life and your plans for the future. I would also like to thank Shobhit Aggarwal for all of his help at MFI and for the random chats we have on the way home after staying late. I would also like to thank Tanya Marwah again for being a fun friend and getting me and Mohit to get some drinks to chat about life and relationships and they gave me a perspective change that definitely changed my life trajectory from what I had imagined. I would like to thank Arpit Agarwal for being my driving buddy between UIUC and CMU and chatting about research and life in general. Next, I would like to thank Chase Noren for always chilling with me in the A Floor hallways. I would like to thank Kate Shih for being so selfless in helping manage RoboOrg with me during the 2 years we were RoboTsars. I would like to thank Tetsuya Narita for being a great friend to me during his time at CMU as well as when I was in Japan. For friends that have graduated, I am thankful to Samuel Clarke for various discussions and introducing me to contact microphones. I would like to thank Travers Rhodes for being a great friend and my eternal Dim Sum buddy. I would like to thank Anahita Mohseni Kabir for her wisdom and advice over the years as one of Manuela's last students at CMU. Leonid Keselman has always been the best for a great conversation in the halls. I would also like to thank Tess Hellebrekers for letting me be a dog uncle at the park and bantering together.

I would also like to thank all of the faculty and administrators that have assisted in my academic and research journey. Thank you Ashlyn Lacovara for helping with quick reimbursements and organizing our lab scrum. Thank you to Jean Harpley for helping Kate Shih and I with RoboOrg during the covid years. I would like to thank Jean Oh for her support and kindness when I interned at NREC as an undergraduate and afterwards for applying to MSR and the Ph.D. I would also like to thank George Kantor for being on my MSR thesis committee and Ph.D speaking qualifier committee and letting me venture into the fields for a bit in Iowa and Massachusetts. I would also like to thank Tim Angert and Chuck Whittaker for allowing me access to the RI machine shops, which helped me immensely in prototyping all of the hardware used in my papers.

I would like to thank Sony Corporation, NSF, NIST, and the Manufacturing Futures Institute (MFI) for funding my research. I would also like to thank my coworkers and intern friends that I met at Sony AI.

Finally, I would like to thank my parents Zhenyong Zhang and Chunming Li and my sister Anna Zhang for supporting me through my entire CMU and Ph.D journey. I would like to thank my dog Lychee for forcing me to go outside and exercise, but also for accompanying me throughout my daily life. I would also like to again thank my wife Erin Zhang for her patience, understanding, and encouragement by my side on my journey to completing the Ph.D. I look forward to sharing the rest of my lifetime with her!

Contents

1	Introduction	1
1.1	Motivation	1
1.2	Contributions	2
1.3	List of Publications	3
2	Playing with Food: Learning Food Item Representations through Interactive Exploration	5
2.1	Introduction	5
2.2	Related Work	6
2.3	Robotic Food Manipulation Dataset	7
2.3.1	Experimental Setup	7
2.3.2	Data Collection	9
2.3.3	Data Processing	11
2.4	Learning Food Embeddings	11
2.5	Experiments	12
2.6	Conclusions and Future Work	15
3	Vibrotactile Sensing for Detecting Misalignments in Precision Manufacturing	17
3.1	Introduction	17
3.2	Related Work	18
3.3	Setup Description	19
3.4	Outcome and Termination Detection	21
3.4.1	Teach: Data Collection	21
3.4.2	Learn: Outcome and Termination Detection Networks	26
3.4.3	Execute: Model Inference	28
3.5	Experiments and Evaluations	29
3.5.1	Amount of Data	29
3.5.2	Location and Number of Microphones	29
3.5.3	Injection of Noise Vibrations	30

3.5.4	Differing Velocities	31
3.5.5	Different LEGO Block Types	32
3.5.6	Different Number of Surrounding Blocks	32
3.5.7	Terminators	33
3.5.8	Vision Baseline	35
3.6	Conclusions and Future Work	36
4	Localization and Force-Feedback with Soft Magnetic Stickers for Precise Robot Ma-	39
	nipulation	
4.1	Introduction	39
4.2	Related Work	41
4.3	Background Theory	41
4.4	System Overview	42
4.4.1	Robot	42
4.4.2	Magnetometer	42
4.4.3	Soft Magnetic Sticker	43
4.5	Experimental Design And Results	43
4.5.1	Precision Force Measurement Stand	43
4.5.2	Minimum Contact Estimation	44
4.5.3	Localization	45
4.6	Demonstrations	48
4.6.1	Squeezing Ketchup	48
4.6.2	Closing Drawers	49
4.6.3	Unlocking a Door	50
4.6.4	Pipetting	51
4.7	Conclusions and Future Work	51
5	A Low-Cost Compliant Gripper Using Cooperative Mini-Delta Robots for Dexterous	53
	Manipulation	
5.1	Introduction	53
5.2	Related Work	55
5.3	Delta Manipulator Design	56
5.3.1	Setup	56
5.3.2	Delta Actuators and Workspace	57
5.3.3	Fingertip design	58
5.4	Delta Modeling	58
5.4.1	Offset between Revolute Joints	58
5.4.2	Learning Delta Robot Kinematics	60
5.5	Force Profile	61
5.6	Experiments	62
5.7	Discussion and Conclusion	64

6	DELTAHANDS: A Synergistic Dexterous Hand Framework Based on Delta Robots	67
6.1	Introduction	67
6.2	Related Work	69
6.2.1	Robotic hands	69
6.2.2	Delta robots	69
6.2.3	Robotic hands for dexterous manipulation	69
6.3	Methodology	70
6.3.1	Hand design and manufacturing	70
6.3.2	Actuation coupling	70
6.3.3	Hand parametrization	73
6.3.4	Hand kinematics	73
6.3.5	Hand design in simulation	74
6.4	Delta Robot Characterization	74
6.4.1	Kinematics	75
6.4.2	Force profile	76
6.4.3	Durability	76
6.4.4	Workspace	78
6.5	Dexterous grasping and manipulation	78
6.5.1	Evaluation of grasping	78
6.5.2	Teleoperation for dexterous manipulation	80
6.6	Conclusions	81
6.7	Acknowledgements	82
7	<i>Tilde</i>: Teleoperation for Dexterous In-Hand Manipulation Learning with a Delta-Hand	83
7.1	Introduction	84
7.2	Related Work	85
7.2.1	Robotic Hands for Robot Learning	85
7.2.2	Teleoperation Systems for Dexterous Manipulation	85
7.2.3	Learning for Dexterous Manipulation	86
7.3	Methodology	86
7.3.1	Dexterous Hand Design	86
7.3.2	Teleoperation Interface	88
7.3.3	Learning with Diffusion Policies	89
7.4	Experimental Setup	89
7.4.1	Data Collection	90
7.4.2	Training Details	90
7.4.3	Tasks	90
7.5	Experimental Evaluations	93
7.5.1	Experimental results	94
7.5.2	Failure Analysis	95
7.5.3	Effect of using visual observations	96
7.5.4	Data Efficiency Evaluation	98
7.5.5	Comparison of Different Imitation Learning Methods	98
7.5.6	Evaluation on Robustness and Generalization of Policies	99

7.5.7	Performance on Challenging Tasks	99
7.5.8	Limitations and Discussions	101
7.6	Conclusions	102
7.6.1	DeltaHand Comparison	102
7.6.2	Teleoperation Interface Details	103
7.6.3	Comparison with Existing Work	106
7.6.4	Learning Policy Implementation Details	106
8	Mixed Dexterity: Enabling Fine-Grained Manipulation with a Dexterous Third Finger	109
8.1	Introduction	109
8.2	Related Work	111
8.2.1	Three Finger Grippers	111
8.2.2	Teleoperation for Imitation Learning	111
8.2.3	Learning for Dexterous Manipulation	111
8.3	Robot Setup	112
8.3.1	Third Finger	112
8.3.2	Teleoperation	114
8.4	Method	114
8.4.1	Servoing Policy	114
8.4.2	Third Finger Policy	115
8.5	Experiments	116
8.5.1	Soft Tortilla	116
8.5.2	Gum	116
8.5.3	Vitamin Bottle	118
8.6	Limitations	118
8.7	Conclusions and Future Work	119
9	Conclusions and Future Work	121
9.1	Conclusion	121
9.2	Key Takeaways and Insights	121
9.2.1	Sensor Design	122
9.2.2	Robot Hand Design	122
9.2.3	System Design for Tasks	123
9.2.4	Lifelong Learning	124
9.3	Future Research Directions	124
9.3.1	Enhanced Dexterity and Precision	125
9.3.2	Generalization and Robustness	125
9.3.3	Application to Real-World Tasks	126

List of Figures

2.1	Our cutting experimental setup.	7
2.2	Images from the 4 cameras spread around our cutting experimental setup.	7
2.3	Our playing experimental setup.	8
2.4	Examples of Kinect, Realsense, and FingerVision images of a cut tomato.	8
2.5	Food item slices. From left to right, top to bottom: apple, banana, bell pepper, boiled apple, boiled bell pepper, boiled carrot, boiled celery, boiled jalapeno, boiled pear, boiled potato, bread, carrot, celery, cheddar, cooked steak, cucumber, jalapeno, kiwi, lemon, mozzarella, onion, orange, pear, potato, raw steak, spam, strawberry, and tomato.	10
2.6	An overview of our approach. The different features (modalities) defined in Section 2.3.3 are used to form triplets to learn embeddings in an unsupervised manner, which are used for supervised classification and regression tasks (blue text).	12
2.7	Fig. 2.7a visualizes playing audio features using PCA. Fig. 2.7b visualizes the embeddings learned using the playing audio features also in PCA space.	15
3.1	Robot setup. We place three contact microphones on both end-effectors on the front, back, and left as well as a fourth contact microphone underneath the acrylic table surface. Both robots have an ATI Gamma Force Torque Sensor as well as an Intel RealSense D405 on the wrist. Finally, we have an external Orbbec Femto Bolt that allows us to record and view each trial from the side.	20
3.2	Connection Setup. We use Ethernet to interface with the Yaskawa GP4 and the ATI Gamma Force Torque Sensor. We use USB to control the Robotiq Hand E and interface with the Behringer UMC 404HD Audio Interface and the two cameras.	21
3.3	NIST Data Collection Flow Diagram	22
3.4	Successful Perturbations for each connector with a variety of perturbations. The x and y perturbations are in mm and θ is represented by the orientation of the arrow originating from the circle from the horizon parallel to the x axis.	23
3.5	LEGO Data Collection Flow Diagram	25
3.6	Rosbag Data Visualizer with Audio Outcome Processing	27

3.7	Vibrotactile Network Diagram	28
3.8	Real World Performance vs Number of Training Data	30
3.9	Audio Outcome Predictions with External Disturbances	32
3.10	Performance when Models are Trained on 0.01 Velocity Scale Data and Utilized on a novel 0.02 Velocity Scale movement	33
3.11	Lego Outcome Detection Results with Leave One Out Block Type Ablations	34
3.12	Different real-world LEGO constrained environments	35
3.13	Performance on Constrained Environments	35
3.14	Audio vs Vision Results	37
3.15	Lego Side Camera View	37
4.1	A) Overview and B) image of modified Franka Gripper localizing to a soft mag- netic sticker on a key. C) By embedding magnetic microparticles in an elastomer, we can create a soft and deformable magnetic sticker. D) Combined with a 3-axis magnetometer inside the gripper, E) the magnetic flux surrounding the sticker can be used for 3D localization, contact detection, and force estimation.	40
4.2	A) The experimental setup with a precision force-measurement stand (Instron) that records position, force, and magnetic flux. A Franka gripper was attached to the top plate to mimic the applied pressure from a robot gripper. B) Magnetic Field from 40 mm above the magnet to 1/2 depth of varying thickness. C) For a thick- ness of 2 mm, the signal change with compression distance up to 1 mm and D) corresponding forces.	43
4.3	A) Overview of experimental setup for minimum contact experiment. B) For a key, glass bottle, and squeeze bottle, C) the signal change during step-wise gripper motions is shown in solid lines, and the final gripper position without magnetic feedback is marked by a vertical dotted line and a dot with the signal at that location. . . .	45
4.4	A) Experimental setup for localization includes a well-placed camera perpendic- ular to a transparent plate. B) A magnet is placed on the acrylic surface and the camera is used as the ground truth for the C/D) final magnetic position.	46
4.5	3D Gauss localization results from random start points as function of distance vs error. Outliers marked in pink show increased error due to edge-effects or signal decay over distance.	48
4.6	Demonstration of squeezing a ketchup bottle. By continuously monitoring the change in magnetic field, we alternate between minimum contact and a 20N grasp. (A) Signal response measurements for (B,C) dispensing different amounts of ketchup into serving cups.	48
4.7	Demonstration showing the magnetic sensor used for closing drawers. (A) The robot moves to close the third plastic drawer. (B) When the magnetic signal stops changing, the drawer has been closed. We use this signal to stop the robot before it pushes the entire drawer set backwards, about 2 cm earlier than without feedback marked by the black vertical line.	49

4.8	A) First, we localize the key for grasping in 2D by scanning in the X and Y axes. B) The intersection of the raw data shows the centroid of the magnet relative to the centroid of the magnetometer inside the gripper C) Second, we align the key to the center of the lock by scanning in the X and Z axes. The intersection represents the center of the lock relative to the centroid of the magnetometer. A constant offset (depth of magnetometer + 1/2 thickness of key + magnet) is added to insert the key and unlock the door.	50
4.9	By using relative force-feedback provided by the magnetometer and soft magnet, we can fill a pipette with liquid and dispense 5 consistent droplets in sequence with different levels of force.	52
5.1	Novel robotic gripper grasping a coin, a card, and a dough roll with two delta robots, each with three degrees of freedom and made from 3D-printed soft material, polypropylene (PP). The linear actuators, compliant delta links, and fingertips are shown for a pair of delta robots. The delta robot's links move up and down with the linear actuators fixed at the joints.	54
5.2	This diagram illustrates the two orthogonal axes of rotation that approximate a universal joint and the measurements of k , s_p , and L . k is shown as the distance between the two orthogonal axes of rotation, L is shown as the distance between the axes of rotation at the top and bottom of a leg, and s_p is shown as the distance between the the axes of rotation where the center of the parallelogram links attach to the end-effector. Each measurement represents a distance constraint between revolute joints in our simulation.	56
5.3	The prismatic delta workspace, in centimeters, has a dome-like shape, where the robot can reach any of the points in X, Y, and Z axes. The three vertical red lines represent the position of the linear actuators. Colors are mapped to the height for a better visualization. The actuator lengths are limited to 4 cm so that no actuator can be above the delta's end-effector. Higher z values can be reached by adding a constant offset to every actuator.	57
5.4	Effects of varying the offset k on the mean error μ from the path of a standard delta. The path of the delta with offsets is shown in red, and the path of a standard delta is shown in blue. For $k = .7$ cm, some actuator inputs become infeasible and the path is cut short.	59
5.5	(a) Marker-based stereo perception system to track the delta end-effector position using two orthogonal Logitech C920 cameras. The graphs compare measured and desired delta positions on the (b) XY and (c) XZ plane when following a test path (shown in blue) with the PP delta. The neural network trajectory prior to training is shown in red, and the trajectory after training is shown in green.	61
5.6	(a) Force profile experiment setup consisting of a GSO-500 Transducer Techniques Load Cell and TPU delta robot with planar fingertip. The delta robot pushes on the load cell with a displacement of 5mm at various positions in the workspace, along X and Y axes. The mean force exerted by the delta at different values of x and y, and standard deviation are shown for the (b) PP deltas (c) TPU deltas.	62

5.7	Panels (a), (b), and (c) show the delta grippers with planar fingertips 3D-printed using PLA grasping a grape, aligning a pile of coins, and taking a coin from that pile and rotating it in hand, respectively. Panel (d) shows the deltas with spherical TPU fingertips picking a grape from its stem.	63
5.8	Timelapse of the compliant delta gripper sliding the top card and picking it up from the deck, and rolling a flat piece of dough into a spiral.	64
6.1	(a) DELTAHANDS is a synergistic soft dexterous hand framework based on Delta robots. We demonstrate dexterous manipulation tasks such as (b)(c) turning the pages of a book and (d)(e) opening the cap of a water bottle through human teleoperation.	68
6.2	(a) CAD model for a 9-actuator hand where the inner actuators of four Delta robots are coupled as one center actuator (blue). Prototypes of (b) 5-actuator and (c) 9-actuator hands.	71
6.3	(a) and (b): Illustration of actuation coupling for 9-actuator and 5-actuator hands. (c) The hand's workspace varies by adjusting the center actuator's actuation distance.	72
6.4	Simulation of DELTAHANDS including (a) single Delta, (b) twin Deltas, (c) triple Deltas and (d) quadruple Deltas as robotic hands. (e) Integration of a four-finger hand on a robot arm.	74
6.5	(a) Kinematics characterization setup with an OptiTrack system to track a Delta robot's end-effector (EE) pose. (b) Comparing the modeled EE's positions from forward kinematics and recorded positions from the OptiTrack system over a Delta robot's workspace. (c) The EE's position errors and (d) orientation errors. (e) Force profile characterization setup with a UR5e robot arm to execute Cartesian motions, a force-torque sensor to record contact forces, and a Delta robot fixed on the table to actuate towards the contact direction. (f) Correlation between actuation distances and contact forces. (g) (h) Characterization of a Delta robot's force profile in normal and lateral directions.	75
6.6	Characterization of force profile and workspace size by varying the Delta robot's base radius and link length. With a smaller base radius and longer link length, the workspace increases but the force profile decreases, and vice versa.	77
6.7	Grasping gallery with DELTAHANDS. We grasp (d) twelve daily objects with (a) a 9-actuator hand and (c) a 5-actuator hand. We also show (b) in-hand cube and rope repositioning.	80
6.8	Teleoperation of (a) Cloth folding by folding the cloth twice in two perpendicular directions; (b) Cap opening by twisting the cap and then removing the cap from the bottle; (c) Cable arrangement by loosening the cable loop, and rearranging the cable.	81
6.9	Teleoperation setup with a Leap Motion camera. An operator's hand and finger poses are tracked by the camera and then mapped to the robot arm's and a 9-actuator hand's motions.	82

7.1	<i>Tilde: Teleoperation for Dexterous In-Hand Manipulation Learning with a DeltaHand.</i> We introduce an imitation learning-based in-hand manipulation system with a dexterous DeltaHand. We present a kinematic twin teleoperation interface, TeleHand, to collect demonstrations on seven dexterous manipulation tasks, such as shape insertion shown above. By using vision-conditioned diffusion policies, the DeltaHand can autonomously complete the tasks.	83
7.2	(a) A DeltaHand with an in-hand RGB camera. A kinematic twin teleoperation interface including (b) a DeltaHand and (e) a TeleHand. The TeleHand uses linear sliders with potentiometers to record the joint states of each finger. The DeltaHand will reproduce the motions of a TeleHand by using the Telehand’s potentiometer readings as desired joint positions for its linear actuators. (c) The DeltaHand’s fingers have 3D-printed rigid-core embedded links and edged joints, which increase the stiffness of each finger and enable them to exert more force. (d) The TeleHand’s fingers have 3D-printed soft links and curved joints, which induce more compliance in each finger. Therefore less force is required for users to teleoperate the robot, which makes teleoperation easier. (f)-(i) In-hand camera images that capture the object and the DeltaHand’s fingers. (j) The TeleHand’s joint states indicate the movement of each finger during a demonstration.	87
7.3	Experimental setup. We mount a DeltaHand on a Franka robot arm. We pre-set the height and location of the Franka arm on top of the experiment workspace. An external RGB camera is mounted in front of the experiment workspace.	91
7.4	Task gallery. We evaluate our system on seven dexterous manipulation tasks: (a) Grasp (b) Block Slide (c) Block Lift (d) Ball Roll (e) Cap Twist (f) Syringe Push (g) Shape Insert . The goals of tasks are indicated by blue arrows in the initial images of task trajectories. For tasks (a)- (d), we separate the training and additional unseen testing objects with white dashed lines.	92
7.5	Qualitative comparisons between task executions from policies trained before and after DAGger demonstrations. By refining the policies with corrective demonstrations from failure cases, the policies can handle these challenging scenarios.	94
7.6	Qualitative comparisons between task policies conditioned on observations with joint states only versus joint states with in-hand images. We observe that visual observations improve the generalizability of the policies by adapting the contact points of the fingers to the objects’ shapes, sizes, and locations.	94
7.7	Effect of using visual observations. We evaluate the policies trained with observations as 1) joint states only, 2) joint states with in-hand images, and 3) joint states with in-hand and external images. The results show performance improvement with visual observations compared to only using joint states as observations.	95
7.8	Data efficiency evaluation. We evaluate policies trained with 20, 40, 60, 80, and 100 demonstrations of the Shape Insert task and average the performances over five models using different random seeds. The success rates increased with more training data.	96

7.9	Examples for comparison between policies trained with 20, 60, and 100 demonstrations. Shape Insert requires the hand to transport and reposition objects to align with the template hole. With less training data, the task mostly failed at the early transportation stage. When the training data size increased, the policies often failed during the final fine orientation alignment stage where most cases are visually ambiguous.	97
7.10	We evaluate the generalization of policies with additional unseen irregular-shaped objects used on the Block Slide task.	99
7.11	Test examples of four challenging tasks: diamond-shaped block insertion, forceful syringe push, ball roll in the air, and block slide in the air over time. Diamond-shaped block insertion requires orientation alignment of more than 90 degrees. Forceful syringe push requires higher alignment precision to succeed. Both ball roll and block slide in the air have non-recoverable failure risks.	100
7.12	Comparisons between (a) the hand from DELTAHANDS [177] and (b) our adapted hand. We customize 1) the fingertips to an omnidirectional design for larger contact area with a soft coating elastomer layer for increased friction, 2) the finger links to a rigid core-embedded multi-material design and the finger joints to a better defined shape for reducing kinematics error and increasing force profile, 3) an in-hand camera to provide clear visual observations for closed-loop policy learning.	103
7.13	Overview of the teleoperation system. The system can be modularized to the control PC, DeltaHand, TeleHand, and external sensors. The Control PC receives and sends all sensor and control signals as ROS topics. It also serves as the main computing source for policy execution. The DeltaHand has its own microcontrollers for the actuators and sensors. The TeleHand is used to interface with a human to get control signals from the slider potentiometers. All other sensors belong to the external sensors module.	104
7.14	The TeleHand can be teleoperated differently depending on the human's preference. The human can a) directly manipulate the TeleHand by grabbing the links or tips or b) attach and use a customized end-effector such as a ring to constrain their finger to the TeleHand's end-effectors.	105
7.15	Visualization of the desired joint states and joint states of the DeltaHand when being teleoperated with a Leap Motion camera or TeleHand. We observe less noise and simpler motions from teleoperation with the TeleHand. With direct one-to-one joint mapping, the TeleHand does not suffer from kinematics constraints, which are shown as saturated joint states when teleoperating using the Leap Motion.	107
8.1	Illustration of the three tasks that the dexterous third finger can perform. In the left hand image, the third finger lifts up one flour tortilla so that the parallel jaw fingers can grasp it. In the center image, the third finger pushes down and out to extract a stick of gum from a pack. In the right hand image, the third finger lifts up the cap of a vitamin bottle.	110

8.2	Robotic Platform for Online Testing: We conduct our experiments on a Franka Panda robot. In the depicted configuration, we have an Intel RealSense D435 as well as an in-hand Adafruit 640x480 USB spy camera. We have attached 3 piezoelectric microphones, 1 on each of the three fingers. The third finger consists of 5 servo motors, 2 rotational and 3 linear actuators that make up the Delta finger. Finally, under the red finger, we have 4 strain gauge sensors that are connected to an Adafruit HX711 connected through 2 pairs of Wheatstone bridges to sense bending and contact with the fingertip.	112
8.3	Teleoperation Setup: We create a kinematic twin to the third finger on the robot and utilize rotary and linear potentiometers to sense the desired joint state of each actuator and command the third finger directly to mimic the positions. We use the 3D Space Mouse in order to move the entire robot to learn the servoing policy. . . .	113
8.4	Our Multi-modal Diffusion Transformer Network Architecture which takes in camera images from both the Wrist Camera and the Finger Camera as well as the mel spectrogram from the third finger contact microphone. In addition, tokens representing the third finger’s joint states and force are fed into the Denoising transformer to predict the noise that was added to the action.	115
8.5	Ripped Tortilla	117
8.6	Folded Grasp	117
8.7	Small Hole	117
8.8	Synchronized images from the wrist and finger cameras from a successful gum inference trial	117
8.9	Third finger stuck behind the gum sticks	117
8.10	Contact between the Delta Finger platform and the two Fin Ray fingers when the Fin Ray fingers are fairly close prevent the third finger from being able to go between the fingers and help with a M&M singulation task.	119

List of Figures

List of Tables

2.1	Baseline and multi-layer perceptron results on 5 evaluation tasks using different learned embedding networks that were trained on our full dataset.	13
2.2	Results on 3 evaluation tasks for different learned embedding networks that were trained using the auxiliary cooked vs. uncooked dataset.	14
3.1	Ablation study over the number of microphones and their placements. Channel 1 is located on the front of the end-effector, Channel 2 is located on the back of the end-effector, Channel 3 is located on the left side of the end-effector or the left fingertip on the Robotiq Hand E, and Channel 4 is located underneath the table. The background colors were created using conditional coloring using Google sheets to assign colors based on the deviation from the mean of each column.	31
3.2	This table illustrates the mean delay between the Online Force Prediction and Offline Force Prediction time stamps as well as the Offline Force and Online Audio predictions times. Thus we can estimate the Online Force and Online Audio Delay by summing the two mean differences together. We had to do this because when the audio termination handler responds before the force termination handler predicts terminate, we are unable to log the result.	36
3.3	This table illustrates the max force that the robot experiences when using the audio and force terminators in either the perturbed location or the correct insertion location.	36
4.1	Summary of Errors Over Localization Methods	47
6.1	Grasping evaluation of 5- and 9-actuator hands on 12 objects from the YCB dataset. We use the number of force closure grasps found out of 20,000 random searches and the largest-minimum resisted wrench (Q_{LRW}) grasping metric to evaluate the hand's grasping capability. We observe that the 5-actuator hand has better sampling efficiency with lower actuation space dimension while having similar-quality grasping compared with the 9-actuator hand.	79

7.1	Experimental results on six tasks. We show that with less than 50 demos, we can achieve success rates over 60% on all tasks before DAgger. With additional DAgger demonstrations, all tasks have improved results and achieved success rates over 80%.	93
7.2	We evaluate the performance of three Imitation Learning methods: Behavior Cloning (BC) (Robotmimic [118]), Implicit Behavior Cloning (IBC) [59], and Diffusion Policy [33] on two tasks, Cap Twist and Shape Insert , either with joint states only or with joint states and in-hand visual observations. We show Diffusion Policy achieves the best performance.	98
7.3	Frequency of sensor and control signals on ROS.	105
7.4	Teleoperation interface characterization and comparison with visual tracking by using a Leap Motion camera. Our proposed TeleHand has better performance on all metrics when evaluating communication efficiency, demonstration collection time, and mapping errors.	106
7.5	Comparison with other state-of-the-art teleoperation systems for dexterous robotic hands. We show that our system has a relatively lower cost while still preserving high dexterity. The DeltaHand has soft fingertips and a compliant finger structure which assists with adapting to different objects and environments and tolerating deviations from learned policies. *The estimated cost of the RBO Hand 3 is \$250 for manufacturing the hand, \$480 for 16 Freescale MPX4250 pressure sensors, and \$1600 for 16 pneumatic Matrix Series 320 valve controllers based on [50, 152]. . .	108
7.6	Comparison with other state-of-the-art policy learning systems for dexterous robotic hands. We show that our system is evaluated on a variety of tasks including grasping, in-hand translation and rotation tasks. Leveraging end-to-end real-world imitation learning has the benefits of data efficiency and less data distribution shift. . .	108
8.1	Ablation study over the inputs on each of the three tasks.	117

CHAPTER 1

Introduction

1.1 Motivation

Robots are steadily improving in capabilities toward deployment in homes and our daily lives. However, a significant gap remains in robots' ability to perform fine-grained dexterous tasks that humans can do effortlessly. For example, a simple task of preparing breakfast with eggs and toast can be easily done by most humans with a little bit of practice. However, there are many facets of this seemingly simple task that robots still struggle to perform due to the need for precise delicate manipulation with adequate sensing.

For example, to fry an egg, a robot would need to first remove the egg carton from the refrigerator, then carefully open the egg carton, take one egg out, and carefully crack it and pull both sides open to release the egg into the pan. In addition, to make toast, the robot would first need to open a bag of bread and then remove the desired number of slices and put them in the toaster. Many of the tasks I have enumerated above are pick-and-place tasks. However, because the robot would be dealing with delicate objects such as eggs and highly deformable objects such as bread slices, the robot would need to have tactile and force sensing to not crush the items but also detect slip so that it does not drop the items. In addition, while the robot is cooking an egg, it needs to visually monitor the cooking process and determine when it is done to remove it properly before it overcooks.

In addition to sensing, the robot would also need to have dexterous fingers to perform challenging tasks. For example, to remove bread slices from a bag, humans typically reach in and use their fingers to insert between the slices and then pick up the desired number of slices. This skill would not be possible for traditional parallel jaw grippers because they do not have sufficient dexterity to manipulate the bread safely within the confined space of a bag. Thus, it is necessary for robots to have dexterous fingers to perform various fine-grained adjustments for positioning tasks within confined spaces or for thin objects such as a credit card to singulate objects for picking and placing.

Finally, we want future domestic robotics to perform these tasks autonomously without a human teleoperator controlling the robot at all times. Thus, we need to have robust robot learning

pipelines that can take in multimodal sensory inputs and human demonstrations and learn policies that can generalize to different objects. In addition, these policies need to be reactive to changes in contact modes such that if a finger has made contact with an object, but is continuing to apply forces that could knock the object over, the robot would be able to adapt and retract the finger autonomously. Additionally, it would be useful to have the robot detect failures accurately and recover automatically to prevent dangerous situations such as exerting excessive forces which could damage the robot, the objects, and the environment.

1.2 Contributions

This thesis addresses some of these fundamental challenges of dexterous manipulation by pursuing a balanced approach that enhances sensing capabilities while developing novel end-effector designs that combine dexterity with control simplicity. The research presents three interconnected contributions that collectively advance the state of fine-grained robotic manipulation for deformable objects. First, in Chapters 2 through 4, we explore multimodal sensors that augment the robots’ perceptual capabilities beyond traditional force and vision modalities. By integrating vibrotactile and magnetic sensing systems, we demonstrate enhanced material discrimination, improved pose estimation, and precise force control. These capabilities are essential when manipulating deformable objects with variable compliance characteristics.

Second, in Chapters 5 and 6, we introduce an innovative end-effector framework based on mini-delta robot mechanisms. This design philosophy creates a middle ground between simple grippers and complex anthropomorphic hands, offering high dexterity with simplified control. The resulting DeltaHands demonstrate capabilities in manipulating diverse deformable objects, from thin poker cards to soft dough, while being easy to teleoperate. Third, in Chapters 7 and 8, we develop advanced learning methodologies that leverage these hardware innovations to create robust control policies from limited demonstration data. By using diffusion models and multimodal sensing inputs, we establish closed-loop control frameworks that can generalize across variations in object properties and environmental conditions.

The integration of these three research directions: enhanced sensing, novel mechanical design, and advanced learning algorithms, creates a comprehensive approach to robotic manipulation that addresses the practical challenges of handling deformable objects in unstructured environments. Through extensive experimental validation across diverse manipulation tasks, we demonstrate that this integrated approach significantly advances robots’ capabilities in performing the fine-grained, dexterous operations that characterize everyday human activities. As robots continue to expand beyond controlled industrial settings into homes, healthcare facilities, and dynamic workplaces, the ability to safely and effectively manipulate deformable objects represents a critical capability gap. This thesis presents foundational work addressing this gap, establishing new paradigms for robotic end-effector design, multimodal sensing integration, and learning-based control that collectively enable more capable, versatile robotic manipulation systems.

1.3 List of Publications

- Chapter 2: Playing with Food: Learning Food Item Representations through Interactive Exploration, ISER 2020 [166]
- Chapter 3: Vibrotactile Sensing for Detecting Misalignments in Precision Manufacturing, under submission
- Chapter 4: Localization and Force-Feedback with Soft Magnetic Stickers for Precise Robot Manipulation, IROS 2020 [75]
- Chapter 5: A Low-Cost Compliant Gripper Using Cooperative Mini-Delta Robots for Dexterous Manipulation, RSS 2021 [122]
- Chapter 6: DELTAHANDS: A Synergistic Dexterous Hand Framework Based on Delta Robots, RA-L 2024 [177]
- Chapter 7: TILDE: Teleoperation for Dexterous In-Hand Manipulation Learning with a DeltaHand, RSS 2024 [178]
- Chapter 8: Mixed Dexterity: Enabling Fine-Grained Manipulation with a Dexterous Third Finger, under submission

1.3. List of Publications

Playing with Food: Learning Food Item Representations through Interactive Exploration

This chapter is based on [Sawhney, Lee, Zhang, Veloso, and Kroemer, [166]].

Abstract: A key challenge in robotic food manipulation is modeling the material properties of diverse and deformable food items. We propose using a multimodal sensory approach to interact and play with food that facilitates the ability to distinguish these properties across food items. First, we use a robotic arm and an array of sensors, which are synchronized using ROS, to collect a diverse dataset consisting of 21 unique food items with varying slices and properties. Afterwards, we learn visual embedding networks that utilize a combination of proprioceptive, audio, and visual data to encode similarities among food items using a triplet loss formulation. Our evaluations show that embeddings learned through interactions can successfully increase performance in a wide range of material and shape classification tasks. We envision that these learned embeddings can be utilized as a basis for planning and selecting optimal parameters for more material-aware robotic food manipulation skills. Furthermore, we hope to stimulate further innovations in the field of food robotics by sharing this food playing dataset with the research community.

2.1 Introduction

Knowledge of an object’s material properties is important for robots learning to perform tasks that involve physical interactions. However, obtaining material properties for deformable objects can be difficult and time consuming. Food items, in particular, vary widely between and within food types depending on factors such as how they were grown, how they were stored, and whether they have been cooked [58]. As a result of our prior knowledge, humans typically can ascertain the basic properties of many different food items using only vision. For properties that are not always clearly conveyed through vision, humans will touch or interact with objects in order to disambiguate its internal material properties, such as knocking on a watermelon to determine its ripeness. Analogously, we believe that the ability to distinguish properties between food items can

be learned using multimodal sensor data from interactive robot exploration [96].

In this paper, we present a unique multimodal food interaction dataset consisting of vision, audio, proprioceptive, and force data acquired autonomously through robot interactions with a variety of food items. Additionally, we use this dataset to explore a self-supervised method for learning embeddings that encode various food material properties and use visual data as input. The network used to learn these embeddings is trained using a triplet loss formulation, which groups similar and dissimilar samples based on the different types of interactive sensor data. In this manner, the robot can learn complex representations of these items autonomously without the need for subjective and time-consuming human labels.

To demonstrate the utility of the learned representations, we subsequently use the learned embeddings as input features to train regressors and classifiers for different tasks and compare their performances with baseline vision-only and audio-only approaches. Our experiments show that regressors and classifiers that use our learned embeddings outperform similar baseline networks on a variety of tasks, indicating that the visual embedding network encodes additional material property information from the different modalities, without requiring the robot to interact with the object at test time. Our project website is located here¹ along with a link to our dataset here².

2.2 Related Work

Many recent works have utilized simulations in order to learn dynamics models and material properties of deformable objects [124, 202]. For example, Matl et al. [125] collect data on granular materials and compare the visual depth information with simulation results in order to infer their properties. Yan et al. [209] learn latent dynamics models and visual representations of deformable objects by manipulating them in simulation and using contrastive estimation, which is similar to our approach. In comparison, we worked with real-world robots and objects to collect data since representations of food that are learned through a simulation environment may not accurately transfer to real world due to variable behaviors during complex tasks, such as large plastic deformations during cutting. There are simulators that can simulate large elasto-plastic deformation [31], but they are computationally expensive, unavailable to the public, and have not yet shown their efficacy in this particular domain.

Other works also use a variety of multimodal sensors to inform a robot of deformable object properties in order to better manipulate them [19, 86, 102]. Erickson et al. [54] use a highly specialized near-infrared spectrometer and texture imaging to classify the materials of objects. Meanwhile, Feng et al. [57] use a visual network and forces to determine the best location at which to skewer a variety of food items for bite acquisition. Finally, Zhang et al. [217] use forces and contact microphones to classify the hardness of ingredients in order to adjust the cutting parameters for slicing actions. In contrast to these works, we only use overhead images as input to our embedding network during inference time, while still incorporating multi-modal data during training.

Numerous works have focused on using interactive perception to learn about objects in the environment [18]. Katz and Brock [91] used interactive perception to determine the location and type of articulation on a variety of random objects. Sinapov et al. [179, 180, 187] had a robot interact with objects using vision, proprioception, and audio in order to categorize them. Chu et

¹<https://sites.google.com/view/playing-with-food>

²<https://tinyurl.com/playing-with-food-dataset>

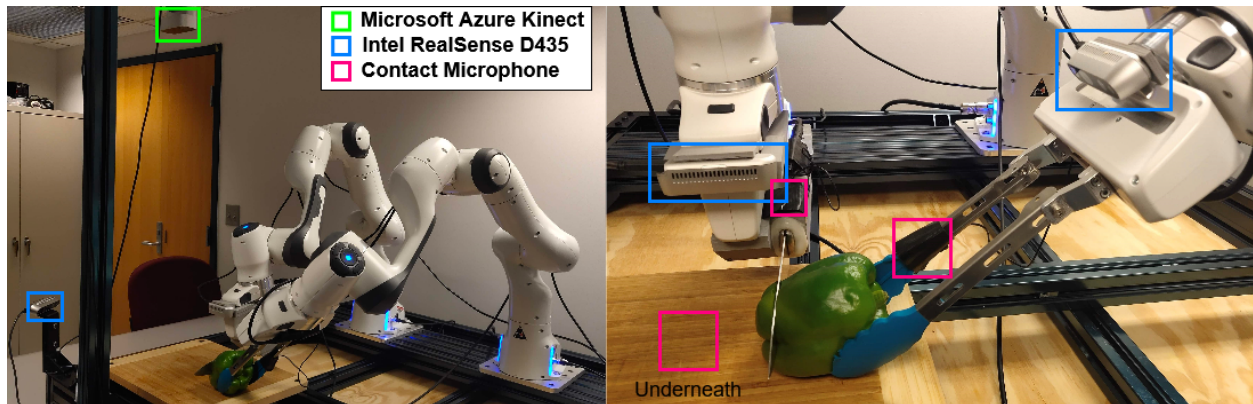


Figure 2.1: Our cutting experimental setup.

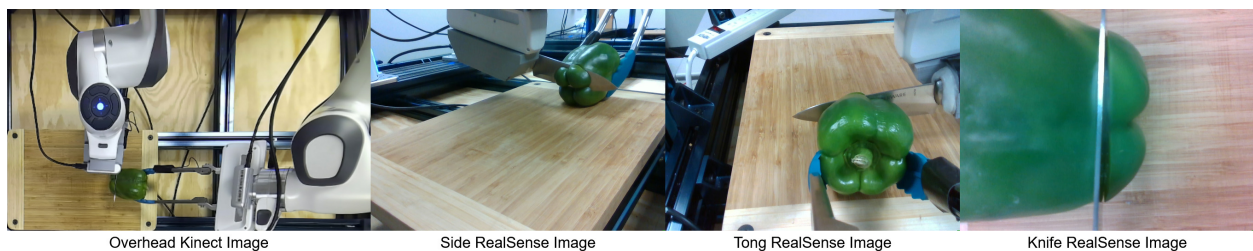


Figure 2.2: Images from the 4 cameras spread around our cutting experimental setup.

al. [37] utilized 2 Biotac sensors on a PR2 along with 5 exploratory actions in order to learn human labeled adjectives from haptic feedback. In our work, we focus our exploratory data collection on deformable food objects instead of rigid objects. Unlike the Epic Kitchens dataset [47], which captures a first person view of humans cooking, we capture proprioceptive information using a robot and repeat consistent interactive actions across food items.

Finally, researchers have been interested in learning deep embeddings for objects in order to create simpler representations for a variety of tasks [16, 205]; however, performing the same task for food items has not been studied extensively. Sharma et al. [173] learn a semantic embedding network to represent the size of food slices in order to plan a sequence of cuts. On the other hand, Isola et al. [81] used human labeled adjectives to generalize the transformation of object states, such as the ripeness in fruit, over time. Although the above works learn embeddings for food objects, both of them focus on using solely visual inputs during both train and test time, while we incorporate additional synchronized multi-modal sensory information during the training of our vision-based embedding networks.

2.3 Robotic Food Manipulation Dataset

2.3.1 Experimental Setup

Our data collection pipeline involves two different experimental setups: one for robotic food cutting and another for food playing, wherein the robot interacts with the food slices that have already been cut. We collected multi-modal sensor data during the cutting and playing data col-

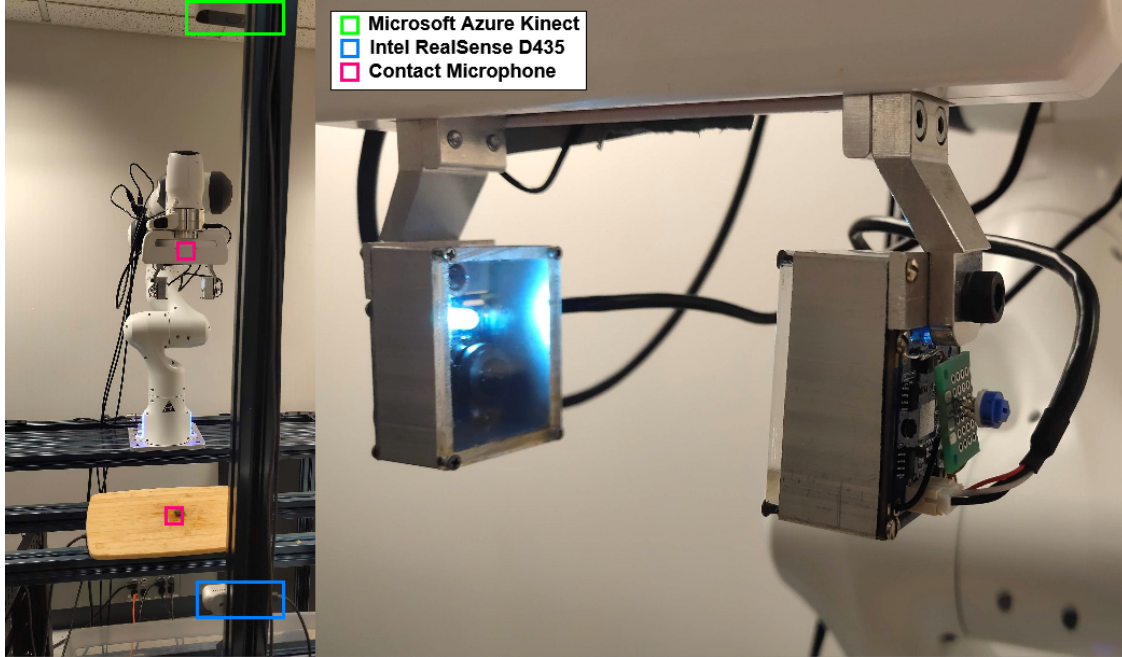


Figure 2.3: Our playing experimental setup.

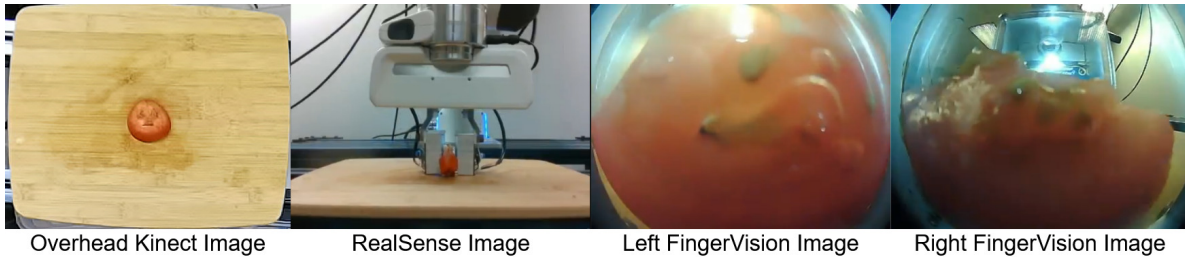


Figure 2.4: Examples of Kinect, Realsense, and FingerVision images of a cut tomato.

lection processes using our Franka robot control framework [216]. The sections below detail the setups for both cutting and playing in more detail.

Robot Cutting:

Our experimental setup for cutting data collection consists of two Franka Emika Panda Arms mounted on a Vention frame with a cutting board in the center, as shown in Fig.2.1. One arm is grasping a custom knife attachment while the other has a set of 8” kitchen tongs mounted to its fingers. There are four cameras attached to the setup: an overhead Microsoft Azure Kinect Camera, a side-view Intel Realsense D435, another D435 mounted above the wrist of the robot holding the knife, and a third D435 mounted on the wrist of the robot holding the tongs. Sample images from each camera are shown in Fig. 2.2. In addition, there are 3 contact microphones: one mounted underneath the center of the cutting board, another mounted on the knife attachment, and the last mounted on the tongs.

Robot Playing:

For our playing setup, we have a single Franka Emika Panda Arm mounted on a Vention frame with a cutting board in the center, as shown in Fig. 2.3. We mounted an overhead Microsoft Azure Kinect Camera and a frontal Intel Realsense D435 that faces the robot. We attach a fisheye 1080P USB Camera³ to each fingertip as in FingerVision [207], except we use a laser-cut clear acrylic plate cover instead of a soft gel-based cover over the camera. This acrylic plate allows us to observe the compression of the object being grasped relative to fingertips. We also added a white LED to better illuminate the object while it is being grasped. Images from all of the cameras are shown in Fig. 2.4. We have 2 contact microphones on the setup: one mounted underneath the center of the cutting board and the other mounted on the back of the Franka Panda hand. The Piezo contact microphones⁴ from both setups capture vibro-tactile feedback through the cutting board, fingers, and tools. The audio from the contact microphones of both the cutting and playing setup are captured using a Behringer UMC404HD Audio Interface⁵ and synchronized with ROS using `sounddevice_ros` [215].

2.3.2 Data Collection

Using our robot cutting setup mentioned in Section 2.3.1, we taught the robot simple cutting skills using Dynamic Movement Primitives (DMPs) [98, 168]. Through ridge regression, we fitted DMP parameters to trajectories collected using kinesthetic human demonstrations as in [217]. Afterwards, we chained DMPs into multiple slicing motions until the specified food item was cut completely through.

In total, we cut 10 slices each from 21 different food types which include: apples, bananas, bell peppers, bread, carrots, celery, cheddar, cooked steak, cucumbers, jalapenos, kiwis, lemons, mozzarella, onions, oranges, pears, potatoes, raw steak, spam, strawberries, and tomatoes. The slices are enumerated from 1 to 14 and were created using similar skill parameters across the food types. The skill parameters vary in slice thickness from 3mm to 50mm, angles from ± 30 degrees, and slice orientation where we had normal vs. in-hand cuts when the knife robot cut between the tongs. There are more than 10 slice types because food items are shaped differently and not all cuts can be executed on every food item, especially the angled cuts. While the robot is cutting the food items, we collect audio, image, force, and proprioceptive data. The resulting slices from various food items are shown in Fig. 2.5.

After the 10 different types of slices have been created, we transfer them to the playing robot setup and begin the data collection process. We run 5 trials on each of the 10 slices in order to capture variations in the objects' behaviors due to differing initial orientations, positions, and changes over time. In each trial, we capture a RGBD image from the overhead Kinect and specify the center of the object. Afterwards, the robot closes its fingers and pushes down on the object until 10N of force is measured. We record the robot's position and forces during this action. Next, the robot resets to a known position, grasps the object, and releases it from a height of 15cm. During the push, grasp, and release actions, we record videos from the Realsense and audio from the two contact microphones as mentioned in Section 2.3.1. Videos are only recorded from the

³<https://www.amazon.com/180degree-Fisheye-Camera-usb-Android-Windows/dp/B00LQ854AG/>

⁴<https://www.amazon.com/Agile-Shop-Contact-Microphone-Pickup-Guitar/dp/B07HVFTGTH/>

⁵<https://www.amazon.com/BEHRINGER-Audio-Interface-4-Channel-UMC404HD/dp/B00QHURLHM>

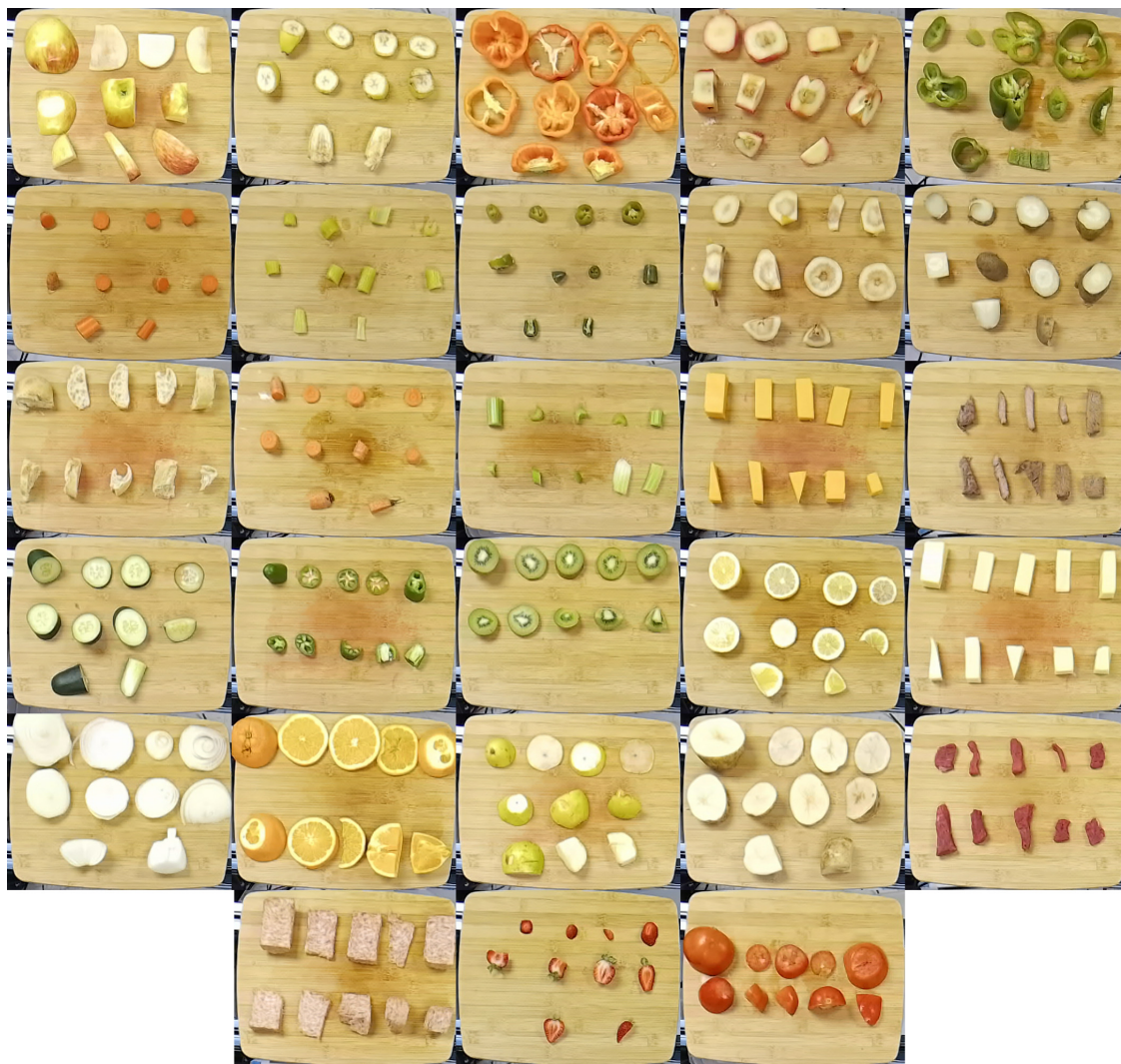


Figure 2.5: Food item slices. From left to right, top to bottom: apple, banana, bell pepper, boiled apple, boiled bell pepper, boiled carrot, boiled celery, boiled jalapeno, boiled pear, boiled potato, bread, carrot, celery, cheddar, cooked steak, cucumber, jalapeno, kiwi, lemon, mozzarella, onion, orange, pear, potato, raw steak, spam, strawberry, and tomato.

FingerVision cameras during the grasp and release actions. Additionally, we record the gripper width when the grasp action has finished and save RGBD images from the overhead Kinect before and after each trial.

Our full dataset is available for download here⁶. The data is located in the appropriately named folders and are sorted first by food type, then slice type, and finally trial number. Additionally, we provide food segmentation masks in the silhouette data folder where we used Deep Extreme Cut (DEXTR) [119] to obtain hand labeled masks of the objects in the overhead and side view images. Then we fine-tuned a PSPNet [219] pre-trained on the Ade20k [223] dataset with our manually labeled masks to generate additional neural network labeled segmentation masks. Finally, we have additional playing data in the old playing data folder, but those slices were all hand cut, and the data was collected in a different environment.

2.3.3 Data Processing

To train the embedding networks, we first extract features from the data. We transform the raw audio data from both the cutting data and playing data (during the release action, push-down action, and grasp action) into Mel-frequency cepstrum coefficient (MFCC) features [109] using Librosa [130]. These features have been shown to effectively differentiate between materials and contact events [217]. Subsequently, we use PCA to extract a lower-dimensional representation of the cutting (\mathcal{A}_{cut}) and playing (\mathcal{A}_{play}) audio features.

To form proprioceptive features (\mathcal{P}), we use the robot poses and forces from the push down and grasp actions. More specifically, for the push down action we extract the final z position (z_f) of the robot’s end-effector once 10N of force has been reached. Using this value, we also find the change in z position between the point of first contact and z_f as Δz , which is an indication of the object’s stiffness. We retrieve the final gripper width (w_g) during the grasping action once 60N of force has been reached. These three values are combined to form \mathcal{P} . Labels for each data sample, such as food class label (\mathcal{F}) and slice type label (\mathcal{S}), are also created according to the food type and slice type performed during cutting.

2.4 Learning Food Embeddings

We train convolutional neural networks, in an unsupervised manner, to output embeddings from overhead images. Our architecture is comprised of ResNet34 [72], which is pretrained on ImageNet [53] and has the last fully connected layer removed. We add an additional three hidden layers, with ReLU activation for the first two, to reduce the dimensionality of the embeddings. We use a triplet loss [171] to train the network, so similarities across food types are captured in the embeddings.

The different modalities of data mentioned in Section 2.3.3 are used as metrics to form the triplets. More specifically, the food class labels (\mathcal{F}), slice type labels (\mathcal{S}), playing audio features (\mathcal{A}_{play}), cutting audio features (\mathcal{A}_{cut}), proprioceptive features (\mathcal{P}), and combined audio and proprioceptive features ($\mathcal{A}_{play} + \mathcal{P}$) are used as metrics. For each sample in the training set, we define the n nearest samples in the PCA feature space (using the L2 norm) as the possible positive samples in

⁶<https://tinyurl.com/playing-with-food-dataset>

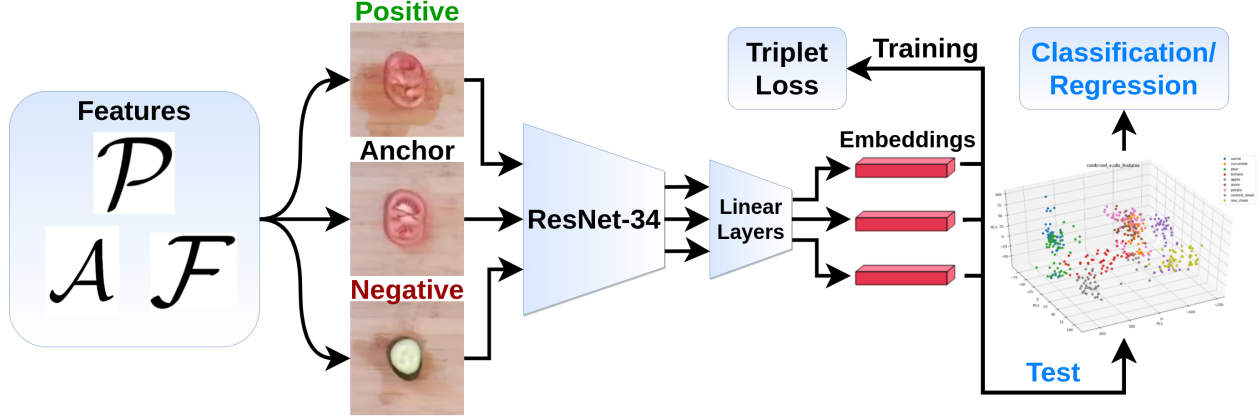


Figure 2.6: An overview of our approach. The different features (modalities) defined in Section 2.3.3 are used to form triplets to learn embeddings in an unsupervised manner, which are used for supervised classification and regression tasks (blue text).

a triplet and all other samples as possible negative samples, where n is a hyperparameter ($n = 10$ was used here). At training time, triplets are randomly formed using these positive/negative identifiers. Fig. 2.6 shows an overview of our approach.

To evaluate the usefulness of these learned embeddings, we train multiple 3-layer multilayer perceptron classifiers and regressors for a variety of tasks, using the learned embeddings as inputs. These results are presented in Section 2.5. When combining multiple modalities, we concatenate the embeddings output from each separate network.

2.5 Experiments

As mentioned in the previous section, embedding networks were trained using the food class label (\mathcal{F}), slice type label (\mathcal{S}), playing audio features (\mathcal{A}_{play}), cutting audio features (\mathcal{A}_{cut}), proprioceptive features (\mathcal{P}), and combined audio and proprioceptive features ($\mathcal{A}_{play} + \mathcal{P}$) for creating triplets. These embeddings were then used to train the multi-layer perceptrons, mentioned in Section 2.4, to predict the labels or values for five different tasks: classifying food type (21 classes), predicting slice width (the width of the gripper after grasping), classifying the hardness (3 human-labeled classes - hard, medium, and soft), classifying juiciness (3 human-labeled classes - juicy, medium, dry), and classifying slice type (14 different classes based on the type of cuts the cutting robot performed to generate the slice). For our two baselines, we trained convolutional neural networks, with a ResNet34 architecture, that use only visual data and another set of 3-layer multi-layer perceptrons that use only \mathcal{A}_{play} data as input to generate predictions for each of the tasks.

To assess the generalizability of our approach, we evaluated the hardness and juiciness classification tasks based on leave-one-out classification, where we left an entire food class out of the training set and evaluated the trained classifiers on this class at test time. We then averaged the results across the 21 leave-one-out classification trials. The performance of all the trained networks on each of the tasks are shown in Table 2.1.

As illustrated in Table 2.1, the purely visual baseline outperforms our embeddings in the food type classification task due to the vast number of labeled images in the ImageNet dataset that

Embeddings	Food Type Accuracy - 21 classes (%)	Hardness Accuracy - 3 classes (%)	Juiciness Accuracy - 3 classes (%)	Slice Type Accuracy - 14 classes (%)	Slice Width RMSE (mm)
\mathcal{F}	92.0	40.7	36.6	12.9	10.9
\mathcal{S}	17.1	37.0	34.9	40.5	11.8
\mathcal{A}_{play}	85.7	35.0	46.0	17.1	9.9
\mathcal{A}_{cut}	93.5	33.5	45.6	16.8	11.3
\mathcal{P}	49.5	47.1	37.0	20.0	7.9
$\mathcal{A}_{play} + \mathcal{P}$	83.8	36.4	40.2	21.4	9.5
ResNet	98.9	34.9	36.5	30.0	13.9
Classifier w/ \mathcal{A}_{play} as input	84.4	40.8	34.0	30.1	34.4

Table 2.1: Baseline and multi-layer perceptron results on 5 evaluation tasks using different learned embedding networks that were trained on our full dataset.

ResNet was pre-trained on. However, ResNet performs worse on the other 4 tasks as ImageNet did not contain prior relevant information on the physical properties that are important for these tasks. Additionally, ResNet was trained to differentiate object classes instead of finding similarities between classes, so when an entire food category was left out of the training dataset, it most likely had no way of extrapolating the correct answer from previous data. Meanwhile, our embeddings contained auxiliary information that encoded the similarity of slices through various multimodal features, without ever being given explicit human labels. This indicates that our interactive multi-modal embeddings provided the neural networks with a greater ability to generalize to unseen data as compared to the supervised, non-interactive baselines.

Additionally, the results show that the audio embeddings provide some implicit information that can help the robot distinguish vegetable types. On the other hand, it makes sense that the proprioceptive embeddings are more useful at predicting hardness and slice width as their triplets were generated using similar information. However, absolute labels were never provided when training the embedding networks, so the learned embeddings encoded this relative information themselves. It should also be noted that in the hardness and juiciness leave-one-out classification tasks, some food types, such as tomatoes, were more difficult to classify when left out of the training dataset than others, such as carrots. This may be due to the small size of our diverse dataset, which has few items with similar properties.

Finally, with respect to the slice type prediction task, there were poor performances across the board due to the inherent difficulty of the task. Due to the variability of shapes between food items, the resulting slices generated by the cutting robot, while executing the same actions, differed greatly at times. Thus, it was to be expected that only the embeddings trained using slice type labels performed relatively well on this classification task. Overall, the results of the evaluations above show that certain embeddings performed better on relevant tasks, which supports the hypothesis

Embeddings	Hardness Accuracy (%)	Juiciness Accuracy (%)	Cooked Accuracy (%)
\mathcal{A}_{play}	98.0	62.9	98.9
\mathcal{P}	63.0	68.4	60.6
$\mathcal{A}_{play} + \mathcal{P}$	99.7	70.6	99.1
ResNet	90.5	66.1	90.4
Classifier w/ \mathcal{A}_{play} as input	82.1	67.4	88.8

Table 2.2: Results on 3 evaluation tasks for different learned embedding networks that were trained using the auxiliary cooked vs. uncooked dataset.

that they are encoding information on different material properties and can be applied in different use cases.

Auxiliary Study with Additional Cooked vs. Uncooked Food Data:

As an addendum to the 21-food class dataset described in Section 2.3.2, we collected an additional dataset of boiled food classes to further explore and evaluate our method’s ability to detect whether a food item is cooked or not through interactive learning. The additional boiled food classes collected were: apples, bell peppers, carrots, celery, jalapenos, pears, and potatoes. Each item was boiled for 10 minutes. Note that for these additional cooked food classes, we did not have the robot cut the food slices due to difficulties grasping the objects. We combined these boiled classes with their uncooked counterparts from the full dataset to form a 14-class dataset and conducted a subset of evaluations on the embeddings learned from this dataset, shown in Table 2.2.

The auxiliary study with the boiled food dataset shows that the playing audio data is effective at autonomously distinguishing between cooked and uncooked food items without being provided any human-provided labels. This is likely because cooking food significantly changes its material properties. The high performance of the $(\mathcal{A}_{play} + \mathcal{P})$ embeddings on the hardness, juiciness, and cooked leave-one-out classification tasks on this smaller dataset demonstrates the ability of our approach to generalize to new data when a food class with similar properties was present during training (apples and pears, potatoes and carrots, bell peppers and jalapenos).

Fig. 2.7a shows the playing audio features (\mathcal{A}_{play}) of the different cooked and uncooked food classes in this dataset using the top 3 principal components. Fig. 2.7b visualizes the learned embeddings based on \mathcal{A}_{play} , also in PCA space. As shown in the plots, there is a distinct separation between the boiled and raw foods in the audio feature space and also in the learned embedding space. Within the cooked and uncooked groupings, there are certain food types that tend to cluster together. For example, the uncooked pear and apple cluster close together, which makes sense given the similarity of these two fruits. Interestingly, they remain clustered close to one another even after they are cooked, even though there is a shift in the feature space between cooked and uncooked.

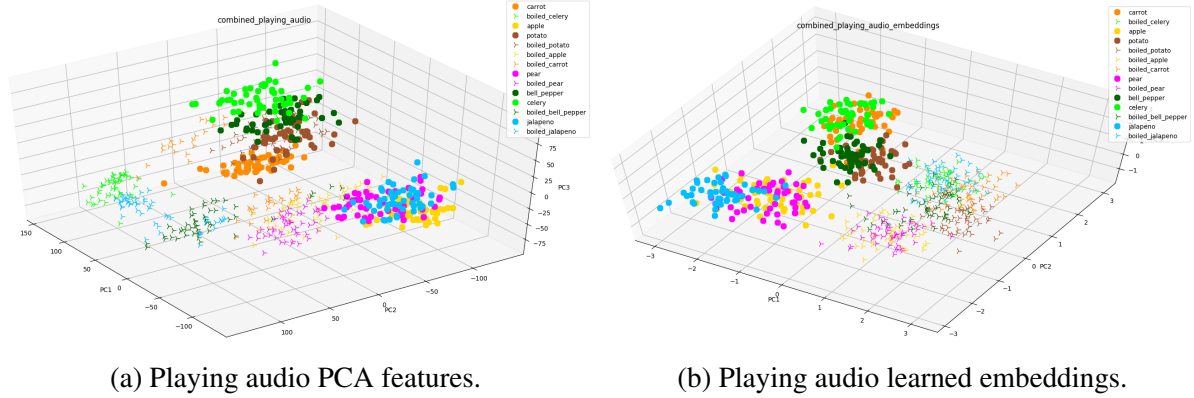


Figure 2.7: Fig. 2.7a visualizes playing audio features using PCA. Fig. 2.7b visualizes the embeddings learned using the playing audio features also in PCA space.

2.6 Conclusions and Future Work

In this work, we have presented a novel dataset consisting of autonomously collected audio, proprioceptive, force, and visual data that was recorded while a robot played with a variety of slices from 21 unique food types that have different shapes and properties. In addition, we learned visual embedding networks that utilized our multimodal dataset to encode properties of food items using a triplet loss formulation. These learned embeddings were shown to encode similarities between food types without explicit human labeling and outperformed normal visual-only and audio-only baselines on a variety of tasks. We hope others can utilize our publicly available dataset to explore novel ways of deciphering and encoding the material properties of food items in order to further propel research on food robotics forward.

For example, an interesting extension to this work would be to apply state of the art computer vision tracking algorithms in order to observe the deformations and movements of the slices using the videos captured from the RealSense and FingerVision cameras. This could serve as an additional metric for creating triplets or possibly be used as a basis for creating dynamics models for different types of food. In addition, since we recorded videos when cutting the slices, it may be possible to predict the shape of resulting slices given an action or vice versa.

For us, the next challenges we hope to tackle using this dataset include: monitoring the progression of food as it is being cooked in order to inform the robot when to intervene, and using the learned embeddings to better execute manipulation tasks such as cutting, flipping, mixing, picking, and placing food items. If we collect data that is complementary to this dataset then we will append the additional data to the original dataset. We also welcome others to contribute as well. Overall, we believe that this work is an exciting step towards autonomously learning about deformable food items through robotic interactions and play.

2.6. *Conclusions and Future Work*

Vibrotactile Sensing for Detecting Misalignments in Precision Manufacturing

This chapter is based on [Zhang, Chang, Aggarwal, Veloso, Temel, and Kroemer] that is currently under submission.

Abstract: Small and medium-sized enterprises (SMEs) often struggle with automating high-mix, low-volume (HMLV) manufacturing due to the inflexibility and high cost of traditional automation solutions. This paper presents a novel approach to robotic manipulation for HMLV environments that leverages vibrotactile sensing. While vision-based systems and force-torque sensors offer valuable information, they can be limited by depth inaccuracies, extensive training requirements, and slower reaction times, respectively. We propose integrating vibrotactile sensors, which capture subtle vibrations and acoustic signals, to provide real-time feedback during manipulation tasks. This approach enables the robot to detect subtle misalignments, which can assist in refining vision-based policies and improving the robot’s overall manipulation skills. We demonstrate the effectiveness of this method in several representative insertion tasks, showing how vibrotactile feedback can be used to predict success or failure of an insertion task as well as predict initial contact between an object grasped in-hand and the placement location. Our results suggest that vibrotactile sensing offers a promising pathway towards more robust and adaptable robotic systems that can better empower SMEs to embrace automation.

3.1 Introduction

Small and medium-sized enterprises (SMEs) form the backbone of many economies, driving innovation and contributing significantly to job creation. However, these businesses often face unique manufacturing challenges, particularly when dealing with high-mix, low-volume (HMLV) production. Unlike high-volume manufacturers who benefit from economies of scale through highly specialized and automated production lines, SMEs must adapt to frequent product changes and smaller production runs. Traditional automation solutions, designed for precisely defined and

repetitive tasks, are often prohibitively expensive and inflexible for HMLV environments. The cost of retooling and reprogramming production lines for each new product iteration can quickly outweigh the benefits of automation, leaving many SMEs reliant on manual labor for complex manufacturing tasks. This reliance on manual labor, however, introduces its own set of challenges, including variability in quality and increased production times.

Thus, HMLV manufacturing would benefit greatly from flexible and cost-effective robotic systems that are capable of performing a diverse set of tasks with minimal downtime and reprogramming effort. A critical component of achieving this adaptability lies in the robot’s ability to reliably and efficiently manipulate objects. Successful robotic manipulation requires not only precise object placement but also careful force control to prevent premature wear or product damage. While vision-based systems have made considerable progress in recent years, offering the potential for generalized object recognition and pose estimation, they still face limitations. These systems can be susceptible to depth inaccuracies, particularly when dealing with materials with varying reflectivity or complex geometries. Furthermore, the performance of vision-based manipulation often relies on substantial training data, which can be time-consuming and expensive to acquire for each new object or task. Force-torque sensors are widely used to provide contact feedback during manipulation; however, they can be slow to react to initial contact which can potentially lead to excessive wear on the robot or damage to the objects if excessive forces are applied due to slight misalignments.

To address these limitations and unlock the full potential of flexible robotic systems in HMLV manufacturing, we propose a novel approach that leverages the rich information provided by vibrotactile sensors. These sensors, typically implemented using contact microphones, capture subtle vibrations and acoustic signals generated during contact and manipulation. They can provide real-time feedback on the success or failure of manipulation attempts, detecting subtle misalignments and other critical events that might be missed by other sensing modalities. This information can be used to refine vision-based policies, allowing the robot to learn from its mistakes and improve its manipulation skills over time. Moreover, the rapid response of vibrotactile sensors can enable more delicate and precise manipulation, reducing the risk of damage to parts and improving the overall robustness of the robotic system.

In this paper, we present the integration of vibrotactile sensors into several representative insertion tasks, a common and crucial operation in many assembly processes. By analyzing the unique insights provided by these sensors, we aim to demonstrate their potential to enhance the precision, reliability, and adaptability of robotic manipulation. We will explore how vibrotactile feedback can be used to predict success or failure of an insertion task as well as predict initial contact between an object grasped in-hand and the placement location. Our goal is to pave the way for more robust and versatile robotic systems that can empower SMEs to embrace automation in HMLV manufacturing environments, ultimately increasing their competitiveness and driving future economic growth.

3.2 Related Work

Recently, there have been a lot of work in leveraging simulation and reinforcement learning to train visual or hybrid force-position policies for insertion tasks. [170] used just a gray scale 32x32 image and reinforcement learning to learn policies to insert the USB, Waterproof, and D-SUB connectors. [15] trained a variable compliance position-force controller in simulation and

evaluated it in the real world on several parallel insertion tasks. [104] used off the shelf vision sensors and a hybrid force position controller to complete various NIST board tasks. However, they utilized expensive structured light ($\sim \$10k-15k$) and time of flight cameras ($\sim \$5k-10k$) that were much more accurate than our consumer grade stereo and time of flight cameras ($\sim \$250-600$). [184] used two wrist mounted Realsenses and forces to learn a visual servoing insertion policy by using a backward learning approach. Finally in one line of research, [137] improved the Isaacgym simulator to reduce the number of contacts between objects with tight tolerances such as a nut and a bolt in order to make the simulation faster and train insertion tasks for sim2real transfer. Then [185] used the learned policies in simulation and transferred them to the real world. Lastly, [186] extended the previous two works to a wider variety of 3D printed assembly tasks from [191].

Researchers have been using vision-based tactile sensors to do a variety of insertion tasks. For example, [103] first used gelsight sensors to insert a USB by calculating its in-hand orientation. [60] used the Digit visuo-tactile sensor to learn a USB insertion policy that is robust to grasp pose variations in a safe self-supervised manner. Recently, [141] used gelsight sensors to precisely rotate an object in hand using a surface, regrasp the item, and then insert it.

Contact microphones and audio have been utilized as an additional modality in a variety of manipulation tasks as well as haptics. [93] explored the uses of vibrotactile sensors for human haptic systems. [40] used contact microphones to determine the amount of granular material that was scooped. [105] used regular microphones to estimate the amount of liquid that was poured into a container without vision. [217] used contact microphones and force torque sensing to adapt cutting parameters for different food items. Finally, [108] used contact microphones to flip a bagel with a spatula, wipe a whiteboard, pour items, and pick and place a piece of tape. By contrast, our focus is on high-precision insertion and assembly tasks.

3.3 Setup Description

The hardware setup is shown in Fig. 3.1. We use a 6-DOF Yaskawa GP4 arm controlled using MoveIt and a Robotiq Hand E two finger gripper to manipulate three NIST Connectors. For the LEGO blocks, we 3D printed a gripper that has stud holes to place a 2x1 LEGO block inside. We utilize the bottom of that 2x1 LEGO block to manipulate other LEGO blocks. This design facilitates easier switching when the bottom of the 2x1 LEGO block has been worn down. Between the gripper and the robot flange, we place an ATI Gamma SI-32-2.5 force torque sensor and a 3D printed camera mount for the wrist mounted Intel RealSense D405 Stereo RGBD camera. On the side of the Vention Frame that the Yaskawa GP4 is mounted on, we place a side mounted Orbbec Femto Bolt RGBD camera in order to detect legos and provide a 3rd person view of the robot while it executes trials.

The LEGO baseplate is hot glued onto a laser cut acrylic pallet that slides in to a fixture that is bolted onto the table surface. The LEGOs to be manipulated are initially placed either on the LEGO baseplate or on the LEGO end-effector. The NIST pallet is also laser cut from acrylic with holes for the NIST sockets to be screwed in. We only utilize three of the NIST connectors: USB, D-SUB, and Waterproof, but our NIST pallet is able to support all of the original NIST assembly task board 1 items with laser cut holes or swappable acrylic subplates that can hold screws as shown on the right side of Fig. 3.1. The NIST connectors are initially placed on the table surface and picked up by the robot before going to the NIST pallet to attempt an insertion trial.

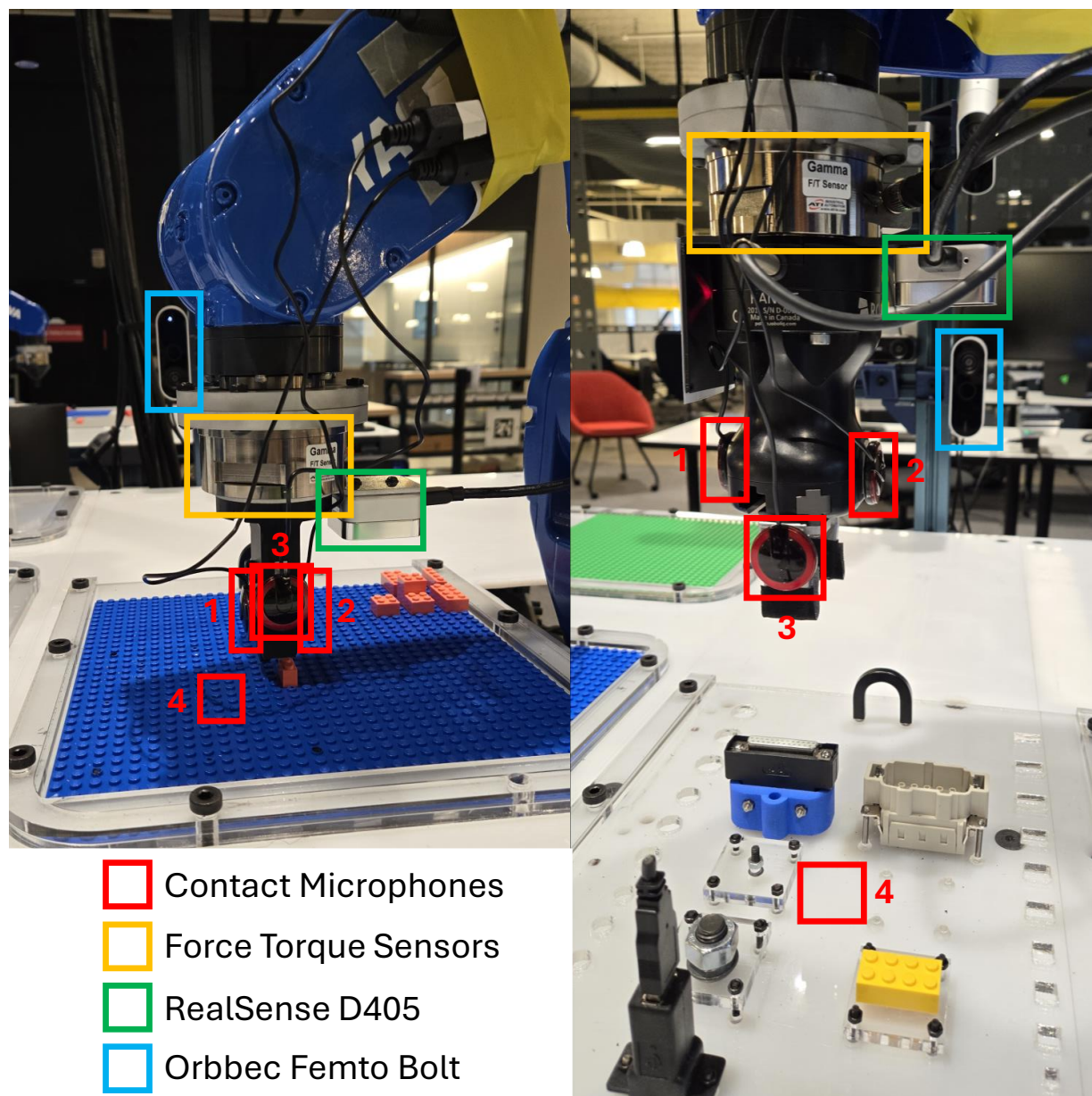


Figure 3.1: Robot setup. We place three contact microphones on both end-effectors on the front, back, and left as well as a fourth contact microphone underneath the acrylic table surface. Both robots have an ATI Gamma Force Torque Sensor as well as an Intel RealSense D405 on the wrist. Finally, we have an external Orbbec Femto Bolt that allows us to record and view each trial from the side.

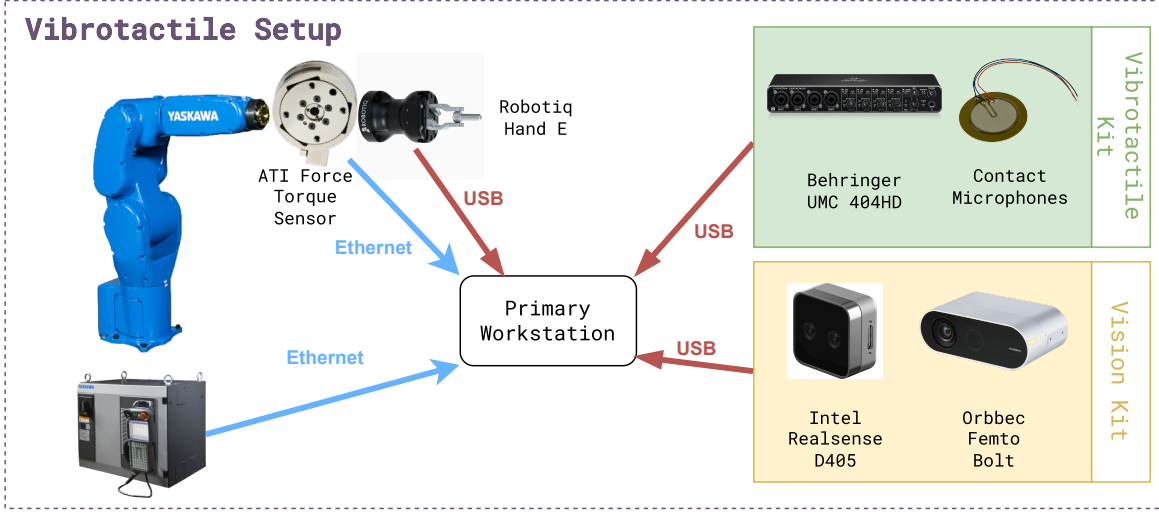


Figure 3.2: Connection Setup. We use Ethernet to interface with the Yaskawa GP4 and the ATI Gamma Force Torque Sensor. We use USB to control the Robotiq Hand E and interface with the Behringer UMC 404HD Audio Interface and the two cameras.

We attach four contact microphones around each robotic setup. Microphone 1 is located on the front of the end-effector, Microphone 2 is on the back of the end-effector, Microphone 3 is on the left side of the LEGO end-effector and on the left finger of the Robotiq Hand E, and finally Microphone 4 is located underneath the table as shown in Fig. 3.1. The Piezoelectric contact microphones are easily available from Amazon. We use 6' 1/4" audio cables from Amazon to connect each contact microphone to a Behringer UMC 404HD USB Audio Interface.

All of the sensors and the robot are connected to a Lambda Workstation with a Nvidia RTX 4090 as shown in Fig. 3.2. The Yaskawa GP4 is connected to the workstation through ethernet and we control it using Moveit with the Pilz Industrial Motion Planner. The ATI Force Torque Sensor is also connected to the workstation through ethernet. Meanwhile the Robotiq Hand E, the Behringer UMC 404HD, and the cameras are all connected to the workstation through USB.

3.4 Outcome and Termination Detection

The contact microphones are used for *outcome detection*, wherein the robot determines after a skill execution has completed if the skill succeeded or failed. We also use the sensors for *termination detection*, wherein, during the skill execution, the robot continuously determines whether to continue or stop the skill.

3.4.1 Teach: Data Collection

To learn the Outcome and Termination Networks for insertion using Vibrotactile Audio, we need to first collect a balanced but varied dataset of successful and failed trials. To do this, we apply a random perturbation on the ground truth insertion pose so that it will frequently cause the robot to fail on the first try. Then the robot will undo the perturbation and attempt the insertion using the

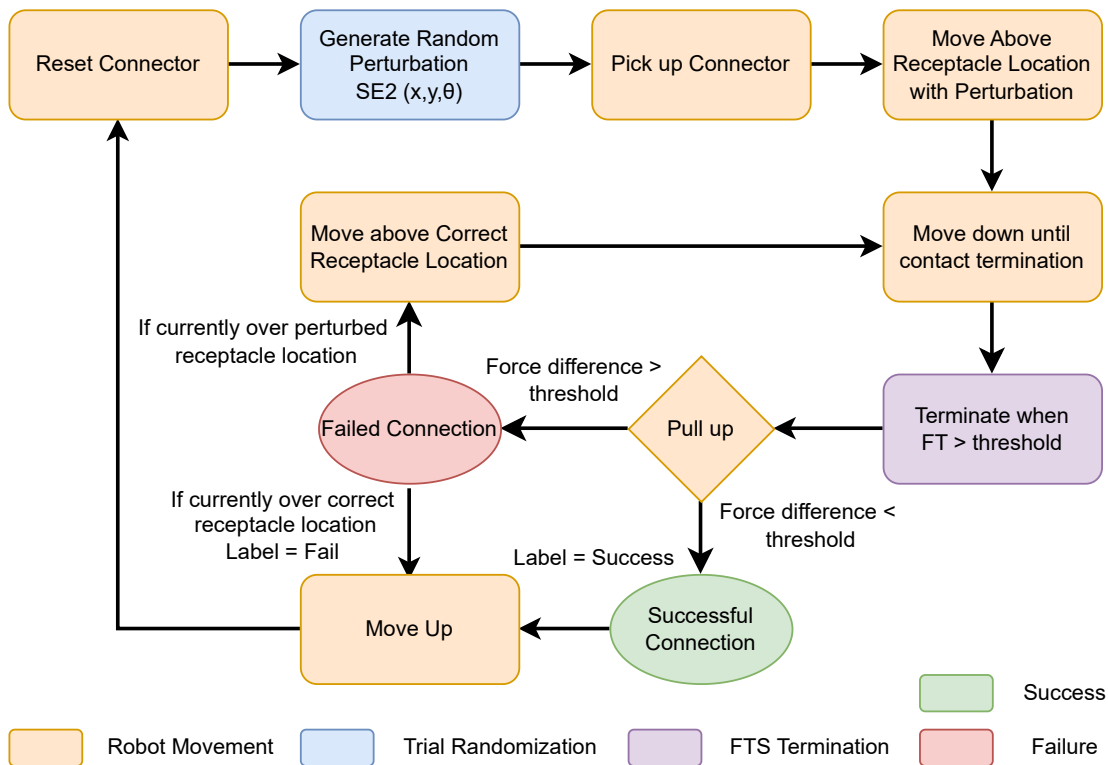


Figure 3.3: NIST Data Collection Flow Diagram

original ground truth pose to collect a more label-balanced dataset. This automated data collection loop consists of three main parts, autonomous resetting, random sampling, and data labeling. The NIST Data Collection Flow Diagram is shown in Fig. 3.3 and the LEGO Data Collection Flow Diagram is shown in Fig. 3.5. The slight differences between the two will be clarified further in each part of the data collection pipeline. All of the trial data is stored in Rosbags that we later extract during the model learning stage.

Autonomous Resetting

NIST Connectors: The first step of autonomous resetting is to record a robot pose where the robot will pick up the connector. Our solution is to laser cut holes into an acrylic plate that fit each connector with a slight amount of extra tolerance. Then we place each connector in a corner of their designated hole so that we can either manually or autonomously reset the connector by pushing it into the corner. Afterwards, we manually jog the robot gripper to be centered around the connector, and then we record that robot pose.

Next we need to record the correct insertion pose, so we first close the fingers around the connector at its pick up pose. Then we manually jog the robot to insert the grasped connector at the desired placement location, and we record the robot’s pose.

If the connector is autonomously resettable, we can also teach 3 more poses to the robot to reset the connector inside its designated hole. A connector is autonomously resettable if it has a

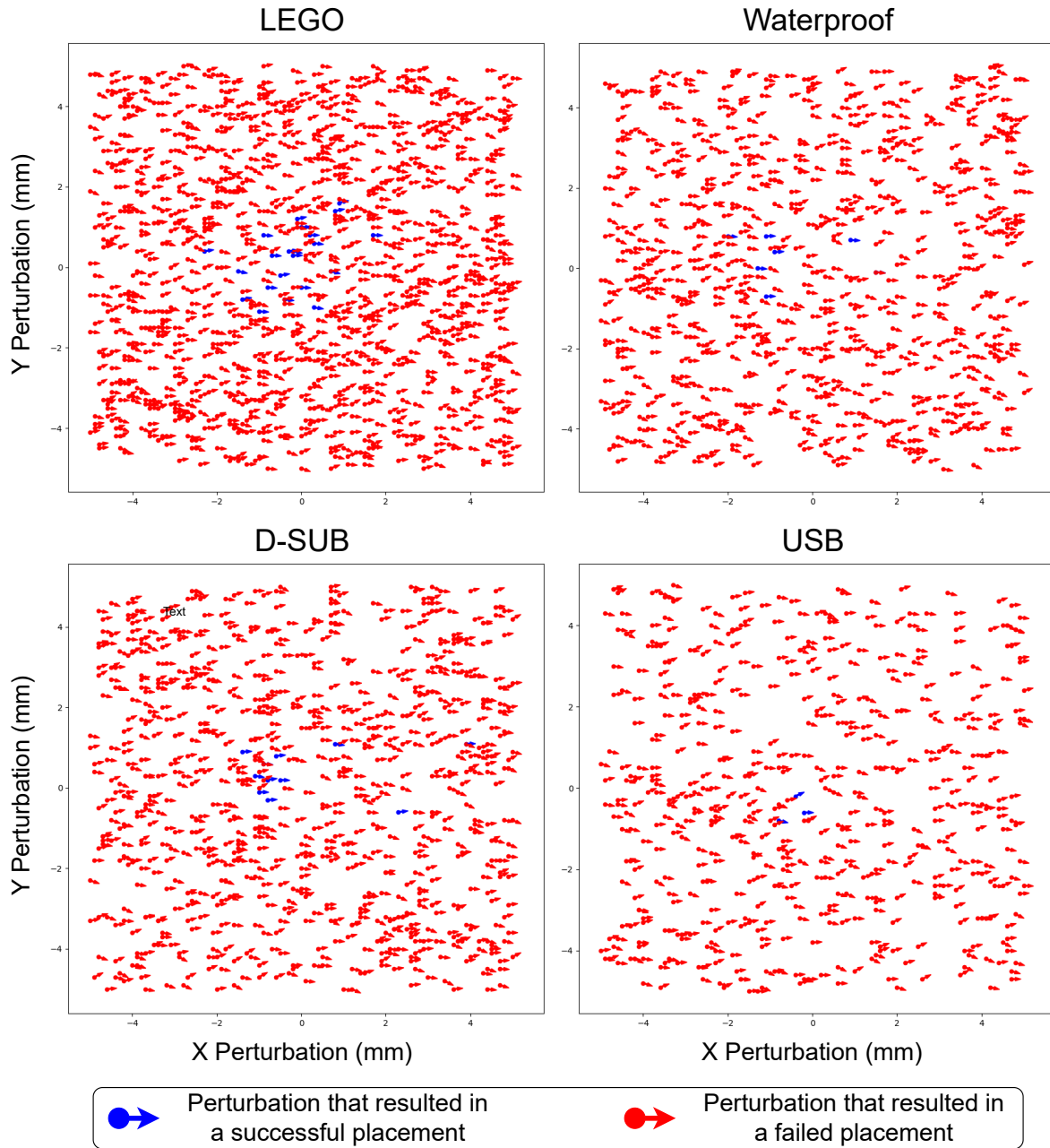


Figure 3.4: Successful Perturbations for each connector with a variety of perturbations. The x and y perturbations are in mm and θ is represented by the orientation of the arrow originating from the circle from the horizon parallel to the x axis.

solid base and will not topple over when released such as the Waterproof and D-Sub connectors. For the USB and Ethernet connectors, they are too thin to stand up on their own and will require a human to help with resetting or a more complex holder.

The 3 poses that need to be taught for autonomous resetting are the release pose, a reset x pose, and a reset y pose. The release pose is required instead of using the original pick pose because when the robot performs a failed insertion, the connector may shift inside the fingers from excessive forces. Then when the robot tries to release the connector in the original pick pose, which is located near the corner of its designated hole, the connector has a chance of falling out when the gripper opens. Hence, the first pose to record is a drop off pose that centers the connector into the middle of the hole. Next we need to teach 2 poses in the x and y directions that will push the connector into the designated corner using the finger tips. We jog the robot to push the connector that is centered in the hole towards the corner in both the x and y directions and then save the two poses when the connector touches the side of the wall. Then, during the data collection, the robot will automatically move the connector from the center release pose to the corner using the reset x and y poses.

LEGO: To perform autonomous resetting of LEGO blocks, we need to record a variety of insertion locations on the LEGO plate with the LEGO gripper. We label each location in terms of its x,y peg location. Then during the data collection, we define a region where LEGO blocks are located and can be placed. Afterwards, the data collection script can randomly select an x,y location to place a LEGO and we extrapolate the ground truth placement location using the closest recorded insertion pose from the previously recorded insertion locations and the default 8mm distance between LEGO pegs in each direction. To pick up LEGOs, we save the previous ground truth location where the LEGO was placed and only overwrite the ground truth location when we place a new LEGO.

Random Sampling

Next to collect the data, we will need to define the range of perturbations. Our data collection script will randomly sample perturbations in the specified ranges (x,y, and θ) and apply them to the correct insertion pose. If the robot does succeed on the first try, then it will not attempt to recover from the perturbed pose. But, if the perturbation does cause the insertion to fail, the robot will pull up and then move above the ground truth pose to attempt a successful insertion.

Fig. 3.4 illustrates the perturbations that result in a success vs. a failure over at least 500 trials of each connector using X and Y perturbation ranges of (-5mm, 5mm) and a θ perturbation range of (-10deg, 10deg). The figure above shows that for some connectors like the D-SUB, the tolerances resulting in success are greater in certain directions (X) vs the other (Y). Overall, none of the connectors have much tolerance to large θ and are typically within 1 to 2 mm in the X and Y directions.

Data Labeling

To autonomously label each trial insertion as either failure or success, we utilize the ATI force torque sensor and an additional verification action. In addition, we use the ATI force torque sensor to stop the robot when it is moving down to insert the connector from above the perturbed location or the correct location. The ATI force torque sensor gives us negative readings when we push down

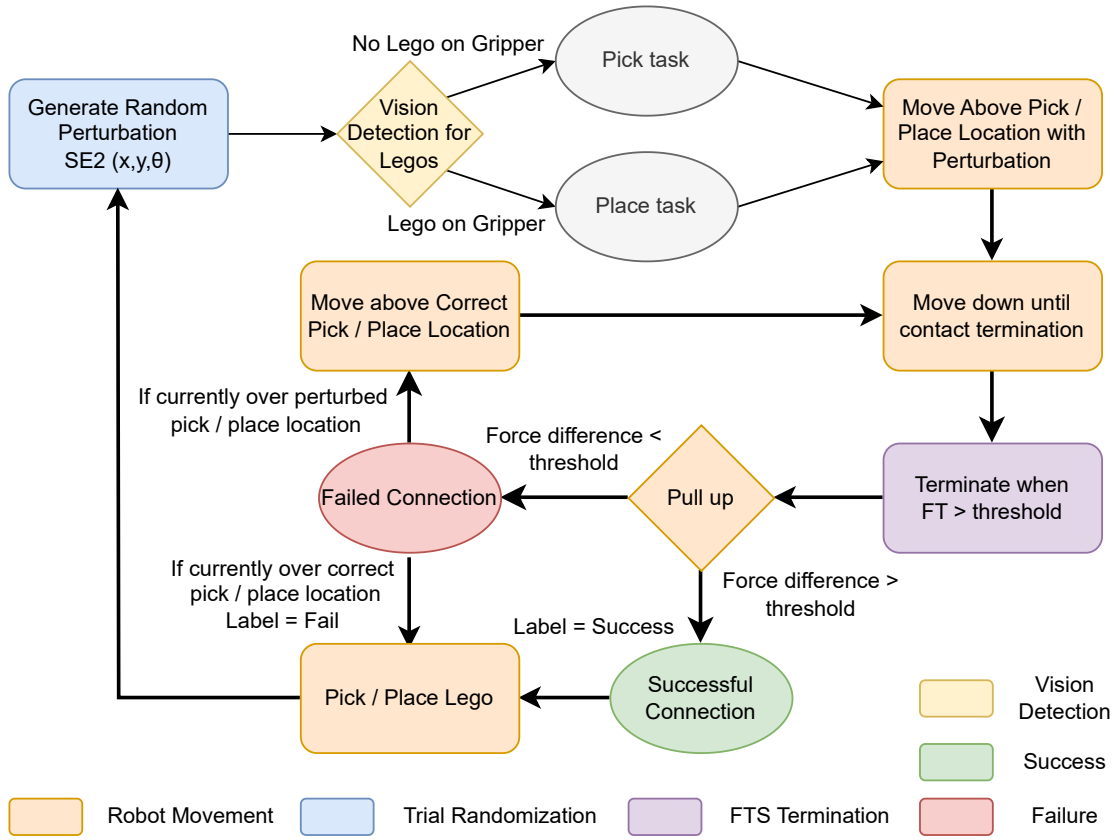


Figure 3.5: LEGO Data Collection Flow Diagram

on an object because positive z is out from the force torque sensor. So we set a force termination threshold of 0N when the robot is moving down to insert a connector or LEGO block. This is because the Robotiq Hand E end-effector and the LEGO gripper weigh around 1kg and 200g respectively, so we observe the force torque sensor readings to be around 8N of positive force for the NIST robot setup and 2N of positive force for the LEGO robot setup when not in contact. While the force torque sensor tries to stop the robot as soon as the force threshold has been breached, it takes a fraction of a second for the robot to decelerate to a stop, which typically results in higher force readings from the sensor depending on the velocity of the robot when it makes contact.

NIST Connectors: To label whether a NIST connector insertion trial was either a success or failure, we take a force reading from when the force torque sensor terminates the move down skill. Then we move the robot up 1 cm and take a second force reading and calculate the difference between the two. If the connector insertion had failed, there would be a large negative z force around -30N during the first sensor reading and a second sensor reading around 8N. When we subtract the first sensor reading from the second reading, the result would be around 40N which is higher than a threshold that we set of around 10N and we would label the trial as a failure. On the other hand, if the connector was successfully inserted, the force reading would be anywhere from 3N to -1N, which would result in a difference below the 10N threshold and a success label.

LEGO: To label whether a LEGO connection trial was either a success or failure, we first take a force reading from when the robot has not started the move down skill and is still in the air, which usually results in a reading of around 2N. Then the robot moves down until the force torque sensor informs the robot to terminate. Afterwards, the robot performs a pull up command at a slow velocity and terminates when a force of 2N is observed. At this point, we take a second force reading and calculate the difference between the two readings. If the LEGO connection had failed, the second reading would be around the same as the first reading at around 2N and the difference would be around 0N and we would label the trial a failure. If the LEGO connection succeeds, the hot glued LEGO baseplate would exert a small force around 0.5N on the LEGO end effector pulling it down and resulting in a sensor reading at around 2.5N after the pull up skill completes. When we take the difference between the two readings, the result would be around 0.5N which is greater than our 0.3N z force difference threshold and then we would label the trial as a success.

In addition, we use Segment Anything [94] to label around 50 wrist and side camera images to train a Mask RCNN [71] network using detectron2 [203] to segment LEGO blocks in each camera view. We then use the trained LEGO segmentation network to predict in real time whether a block is on the LEGO gripper or on the LEGO baseplate, which corresponds to the Vision Detection for Legos step in Fig. 3.5. If the LEGO is on the gripper, we know that the next skill should be a placement skill, but if there are no LEGOs on the gripper, we know that the next skill should be a picking skill. When the LEGO skill has been completed, we use the LEGO segmentation network to check whether the skill was successful based on whether the LEGO block is on the gripper or not.

3.4.2 Learn: Outcome and Termination Detection Networks

After we have finished collecting a sufficient dataset for training the termination and outcome detection networks, we can use the provided scripts in the model training folder to unpack the Rosbags into separate folders with videos, audio wav files, force readings, robot joint states, and robot skill termination logs. Fig. 3.6 shows our visualization tool that allows users to replay a trial using the extracted rosbag folders. We use the skill termination logs for the move down to contact skill to create the outcome and termination datasets for training the neural networks. The skill termination logs contain information on when the Force Torque sensor exceeds the provided threshold as represented as the vertical black line in Fig. 3.6.

Outcome Detection Data Processing: To create the dataset for training the Vibrotactile Outcome detection, we segment the audio data from 0.7 seconds before the force torque threshold has been exceeded and 0.3 seconds after to create 1 second of audio data. We also augment each dataset by sampling twenty 1 second audio segments ± 0.2 seconds from the force torque threshold point for the outcome detection. Thus, if the force torque threshold point is at 3 seconds, the original audio segment is from 2.3 seconds to 3.3 seconds. In addition, we sample an additional twenty 1 second audio segments randomly from the range of 2.1 seconds to 3.5 seconds. We then use the label from the force torque outcome detection from subsection 3.4.1 to label all of the sampled audio segments as either successful insertions or failures.

Termination Detection Data Processing: For Vibrotactile Termination detection, we split the audio data into a nominal section where the robot has started the move down skill but hasn't contacted anything yet and a terminate section around the point where the force torque sensor has crossed the threshold. Then we sample twenty five 0.5 second audio segments from each of the

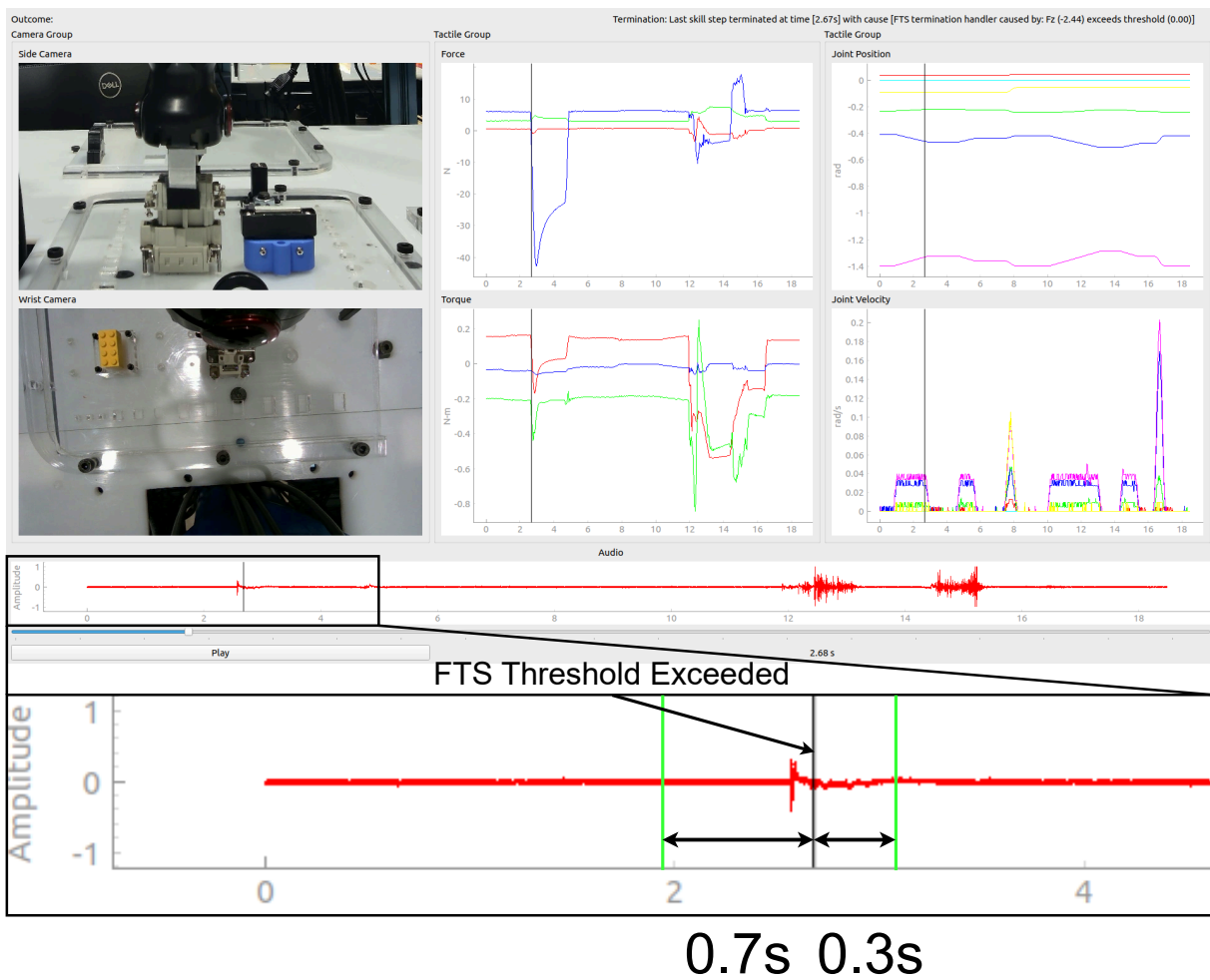


Figure 3.6: Rosbag Data Visualizer with Audio Outcome Processing

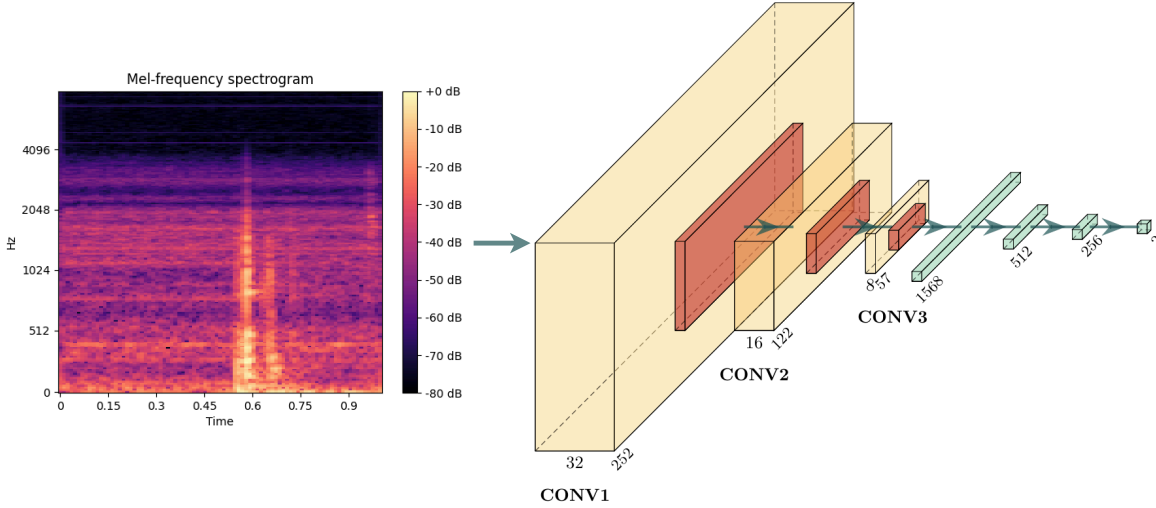


Figure 3.7: Vibrotactile Network Diagram

nominal and terminate audio sections to create the dataset. The nominal section has a range from the start of the skill to 0.5 seconds before the force torque threshold has been reached. Meanwhile the terminate section has a range from 0.6 seconds before the force torque threshold has been reached to the time when the force torque sensor triggers termination. More concretely, if the force torque threshold point is at 3 seconds, the nominal audio section is from 0 seconds to 2.5 seconds, while the terminate audio section is from 2.4 seconds to 3 seconds.

Network Training Fig. 3.7 shows our network architecture for predicting vibrotactile outcomes or termination. We take the labeled audio segments for both outcome detection and termination detection and we create mel spectrogram images for each audio channel using librosa [131]. We then stack the mel spectrogram images from each audio channel and use Pytorch to train each convolutional neural network and save the models to disk when the test accuracy and test loss are better than the previous best test accuracy and loss. We run outcome detection training for 300 epochs while we run termination detection training for 100 epochs. Each saved outcome model is around 3.5 MB while each saved terminator model is around 1.5 MB because the length of the input is 0.5 seconds instead of 1 second. These small models are fast to load and run as compared to larger models that, and our evaluations showed that the larger models did not lead to increased prediction performance. We have tested larger vision models such as ResNet18; however, the test performance does not improve significantly, so we decided to use the smaller and faster models in the experiments we will discuss below.

3.4.3 Execute: Model Inference

After the Vibrotactile Outcome and Termination models have been successfully trained, we can immediately use them for real-world model inference. We use Pytorch’s just in time model loader to load different saved models on the fly given configuration files. We can also vary the number of audio channel inputs that the network uses with the configuration files. This modularity with configuration files can allow new models to be trained and deployed while the robot performs its task and collects more data continuously. The main difference between offline data processing and

training and online model inference is that we have a continuous rolling audio buffer with a size of around 10 seconds. This audio buffer allows the robot to query for an audio segment at any previous specific point of time to be input into the corresponding outcome or termination network. The outcome and termination networks will publish to similar ROS topics as the force termination and outcome detectors that were used to provide the ground-truth labels.

3.5 Experiments and Evaluations

In this section, we will go over the various experiments and ablation studies that were performed using the Vibrotactile sensors on each of the two robot setups. The first section is based on online evaluations with actual robot executions while the second section is based on offline evaluations from data collected during the online evaluations.

3.5.1 Amount of Data

We first wanted to explore the performance of the trained vibrotactile outcome network in the real world given differing amounts of training data to imitate training new models online as more and more data are collected. Thus, we first collected 50 training data and then 50 test data and trained an initial outcome detector for each of the three NIST connectors. Then we used the trained vibrotactile outcome detectors, performed 50 real-world test trials for each connector, and recorded the success rate. Then, we moved the original 50 test data to training data and used the newly collected 50 test trials to train a new vibrotactile outcome detector with 100 training data and 50 test data. We then repeat this process of using 50 newly collected test data as training data for future models until we reach 300 trials at which we start running 100 test trials instead of 50.

The results of this experiment are shown in Fig. 3.8. We see that collecting around 300 training data results in the highest average success across the three NIST connectors. We believe that while performance is still decent with as few as 50 training data, the more data that is collected, the more variety of insertion attempts will be experienced, which can be used to train a more accurate audio outcome prediction model. However, 300 training data seems like a good recommendation for a fairly reliable outcome detector.

3.5.2 Location and Number of Microphones

We next did an ablation study offline on the location and number of microphones to determine which microphones are effective for different connectors because we wanted to do an exhaustive study on each combination of microphone channel inputs. As a reminder, the microphone locations in Fig. 3.1 are located on the robot as follows: Channel 1 is in front of the end-effector, Channel 2 is behind the end-effector, Channel 3 is on the left side of the end-effector or the left finger tip, and Channel 4 is located underneath the table. Table 3.1 shows the results based on each combination of microphone input. For the LEGO blocks, Channel 3 is the most informative of the four microphones. Meanwhile for the NIST connectors, Channel 3 is the least informative and seems to be detrimental to the outcome prediction for each connector. When we combine the audio channels, the performance does improve over just using any 1 microphone.

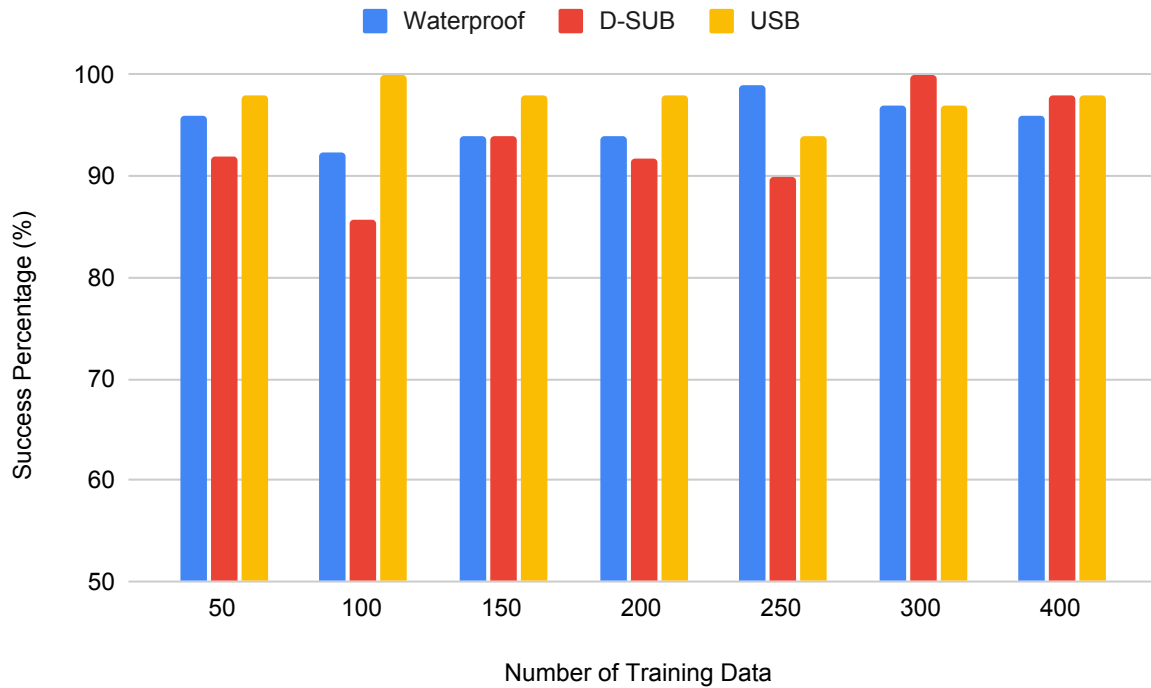


Figure 3.8: Real World Performance vs Number of Training Data

3.5.3 Injection of Noise Vibrations

One key issue we wanted to quantify was the impact that external noise or vibrations would present to industrial setups. Thus, we first trained outcome detectors using the data from subsection 3.5.1 which had 500 trials without external disturbances. Then we directly used the trained models in a setup with a pair of Mackie CR3 Speakers placed on a table across from the robot setups at full volume ($\sim 95\text{dB}$) that was playing audio recorded from real industrial factories in a loop. We found out that there was no performance degradation from external audio from the air. This is because of the properties of contact microphones that almost completely block air vibrations but pick up only contact-based sound. Thus, we next tested placing a dremel running at around 15000 RPM on the tabletop and tested the same trained models on the Waterproof and D-SUB connectors. Fig 3.9 contains the results of this experiment and we can see that there was around a 20 percent drop in real world performance when exposed to the vibration of the dremel.

To test how much the vibration reached the robot, we used only the first 3 channels of the audio (i.e., the 3 contact microphones attached to the end-effector) and observed that for the D-SUB connector it regained most of its performance, while for the waterproof connector it performed worse. This result may be due to a variety of factors, but one factor may be that when the connector is successfully placed into the socket, the socket may transmit the vibrations from the dremel to the end-effector's contact microphones which could affect the outcome prediction.

Input	Waterproof	D-SUB	USB	Lego
Channel 1	99.7%	97.4%	97.3%	83.5%
Channel 2	99.3%	95.6%	98.2%	86.2%
Channel 3	99.0%	89.8%	95.9%	92.6%
Channel 4	99.9%	98.4%	97.9%	82.0%
Channel 1+2	99.8%	97.0%	97.8%	88.7%
Channel 1+3	99.6%	98.4%	97.3%	86.6%
Channel 1+4	100.0%	98.5%	98.2%	91.0%
Channel 2+3	99.2%	98.3%	97.9%	91.5%
Channel 2+4	99.7%	99.0%	98.3%	90.9%
Channel 3+4	99.9%	98.6%	98.2%	93.5%
Channel 1+2+3	99.6%	99.5%	98.1%	89.2%
Channel 1+2+4	100.0%	98.7%	98.3%	90.5%
Channel 1+3+4	99.7%	99.0%	98.2%	91.8%
Channel 2+3+4	99.9%	98.7%	98.3%	90.9%
Channel 1+2+3+4	99.9%	98.8%	98.2%	92.8%

Table 3.1: Ablation study over the number of microphones and their placements. Channel 1 is located on the front of the end-effector, Channel 2 is located on the back of the end-effector, Channel 3 is located on the left side of the end-effector or the left fingertip on the Robotiq Hand E, and Channel 4 is located underneath the table. The background colors were created using conditional coloring using Google sheets to assign colors based on the deviation from the mean of each column.

3.5.4 Differing Velocities

Another evaluation we performed was whether the velocity of the robot may affect the prediction of the vibrotactile models. The previous data collected in subsection 3.5.1 was using a MoveIt velocity scaling parameter of 0.01 when moving down to insert the connector. However, we wanted to see how different velocities would affect vibrotactile outcome prediction performance. Thus, we tested the vibrotactile outcome detectors trained with 500 trials of 0.01 velocity scaling and tested it directly in the real world with a velocity scaling of 0.02 for faster insertions. The results of this experiment are located in Fig. 3.10. For some of the blocks and connectors, there was little to no degradation in performance such as the 2x1 block, 4x2 block, and D-Sub connector. However, for the 2x2, 4x1, and waterproof connector, there was a decrease in performance of around 10-25%. This may be due to differences in what vibration sounds are created when additional force is exerted during insertions due to the higher velocity which may cause outcome predictions to be misclassified.

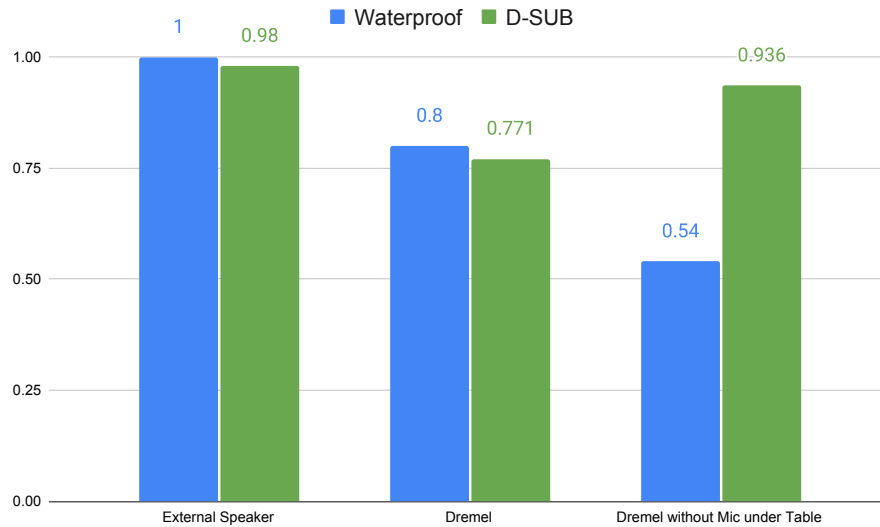


Figure 3.9: Audio Outcome Predictions with External Disturbances

3.5.5 Different LEGO Block Types

Since LEGO blocks are modular with similar structures but different shapes, we wanted to see how well the vibrotactile outcome detector would perform on block types that were left out of the training data. This would mimic using a pre-trained outcome detector on a fairly similar part without collecting additional data and training a new outcome detector for the new part. Thus, we trained individual outcome detectors for each block type by leaving that block type out of the training data and evaluated their performances in the real world.

The results in Fig. 3.11 show that the performances are mixed. For 2 of the LEGO block types (2x1 and 4x1), the performance actually improves by 3-5% when the outcome detector is trained without data on those specific block types. On the other hand, for the 2x2 and 4x2 block types, the performance decreases around 10% for the 2x2 block and around 25% for the 4x2 block. We believe that this is due to a difference in shape as the 2x2 and 4x2 blocks are wider, which results in differing sounds due to additional contacts between the block and the LEGO plate. Meanwhile, the 2x1 and 4x1 blocks may be closer in sound to each other so if one or the other was in the training dataset, the resulting outcome detector was able to generalize to the other connector.

3.5.6 Different Number of Surrounding Blocks

In addition, since LEGO structures can be constructed in a modular fashion, we wanted to see whether having different constraints around a desired pick or place location would affect vibrotactile outcome predictions. Thus, we tested the LEGO outcome detector in different environments where a LEGO block of a specific size would be surrounded on 1, 2, or 3 sides as shown in Fig. 3.12. We couldn't surround the block on four sides because our LEGO end-effector has a fingernail-like protrusion that we use as a lever to pick up LEGO blocks in front of the 2x1 LEGO block on the end-effector.

The results of this experiment are shown in Fig. 3.13. The 4x1 and 4x2 LEGO blocks experi-

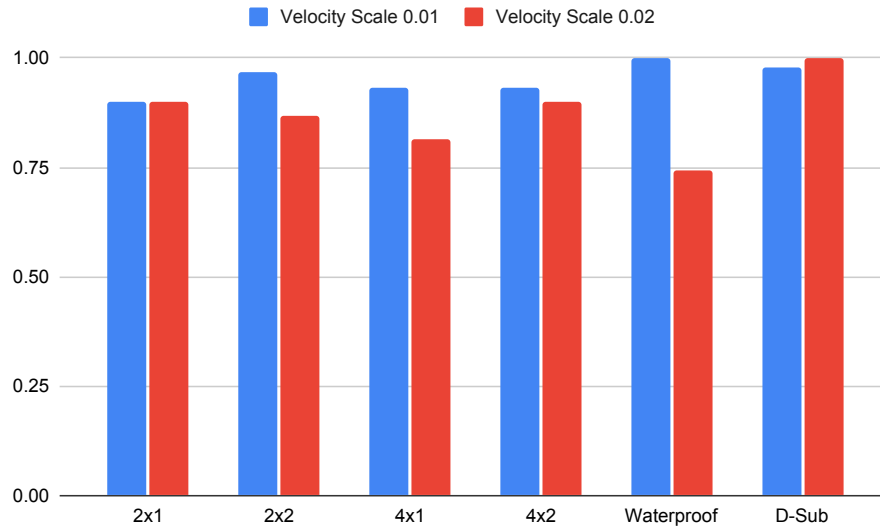


Figure 3.10: Performance when Models are Trained on 0.01 Velocity Scale Data and Utilized on a novel 0.02 Velocity Scale movement

enced the highest degradation in performance with minimal constraints because these two blocks are longer than the 2x1 and 2x2 blocks, so they stick out on either side of the 2x1 LEGO block on the end effector. Thus, when the end-effector was just slightly misaligned, the parts of the block that extend out would become a lever when it contacts the surrounding blocks, which would cause the block to fall off. Then the outcome predictor would falsely predict success because while it did successfully detach from the end-effector, it did not correctly place the block securely onto the LEGO plate. This failure case did not affect the 2x1 or 2x2 LEGO blocks as much because the side blocks did not cause the blocks to fall off the end-effector very often. However, when the 3rd block constraint was added under below the placement location, more hard contacts were made with that block due to slight misalignments and sometimes the constraint blocks would pop off the LEGO plate, or the block on the end-effector would fall off. We did not conduct experiments for the 4x1 and 4x2 blocks on the 3 constraints environment because it was difficult to get good trials as they already suffered in the 1 and 2 constraint environments. Because we did not have any prior training data on each of these failure scenarios, there were many more mispredictions compared to the other experiments above.

3.5.7 Terminators

For our final online evaluation, we wanted to test the performance of the vibrotactile audio based termination compared to force termination that we used as ground truth. We hypothesized that the vibrotactile termination would terminate faster than the force termination because it takes time between when the robot initially makes contact with an object before the force torque sensor will register a sensor value higher than a defined threshold. This can be seen in Fig 3.6 where the force-threshold-exceeded time, noted as the vertical black line, is a few milliseconds after the spike in the audio signal. However, there is still some latency from processing the audio data into spectrograms and then passing it into the neural network, thus sometimes the online force

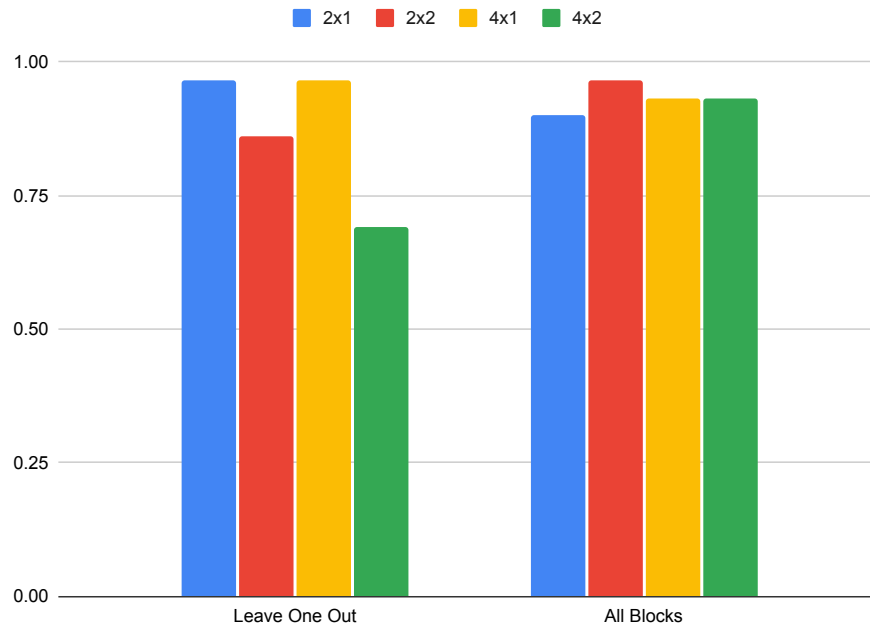


Figure 3.11: Lego Outcome Detection Results with Leave One Out Block Type Ablations

terminator reacts faster than the audio-based terminator. However, when the robot is perfectly aligned for insertion, there is less force until it reaches the bottom, but the vibrotactile audio-based terminator can stop as soon as it senses contact between the robot and the placement location. The results of our vibrotactile audio-based termination trials are shown in Tables 3.2 and 3.3 below. We averaged the results from around 40 trials for both the Waterproof and D-SUB connectors.

First we will go over the latency calculations in Table 3.2. The first row represents the average latency between when the online force threshold is breached and predicts the robot should terminate minus the actual time the force torque sensor published the force that would exceed the threshold. Between the two connectors, this force threshold latency for online prediction is consistently around 5.35 milliseconds. The second row represents the latency between the offline force prediction versus the online audio prediction. We had to calculate this average for each connector because sometimes only the audio termination predicts success when the connector is actually aligned whereas the force threshold did not predict success so we don't have the online force and online audio direct comparisons. Finally we sum up the latency from row 1 and row 2 to get the online force and online audio latency of around 9.36 milliseconds for the waterproof connector and 3.83 milliseconds for the D-SUB connector. This aligns with what we witness from the actual real-world trials where the vibrotactile audio-based termination often predicts contact faster than the force termination except when the force spikes heavily due to a misalignment between the connector and its receptacle.

Next we will go over Table 3.3 which covers the average stopping force when using the Vibrotactile Audio Terminator vs the Force Termination. There was not much difference between the Audio and Force termination final forces when the insertion location was perturbed because the robot is still moving when the terminator tries to stop the robot, so it takes some time for the robot to completely stop and thus the force will continue to rise above the threshold. However, for the

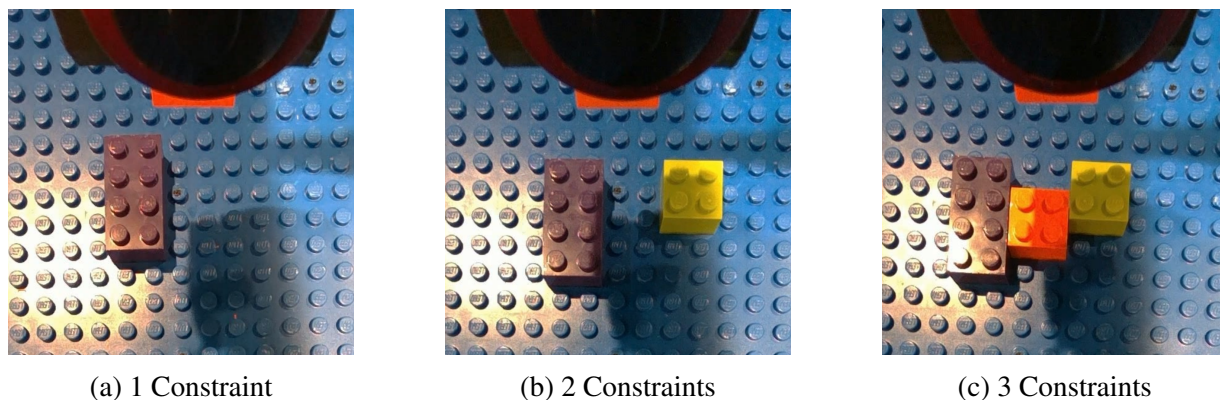


Figure 3.12: Different real-world LEGO constrained environments

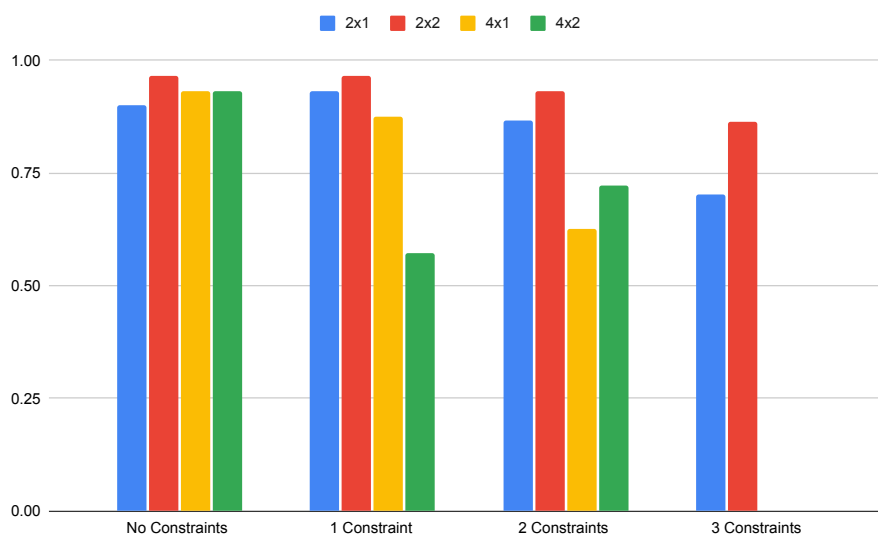


Figure 3.13: Performance on Constrained Environments

cases where the robot is inserting the connector over the correct location, there is around a 8N difference for the Waterproof connector and a 3.5N difference for the D-SUB connector when using the Audio Terminator, which demonstrates that the Vibrotactile Audio Terminator can reduce the forces experienced by the robot when inserting connectors due to its earlier termination prediction.

Finally, as a word of caution, the vibrotactile audio-based terminator can occasionally have false positives without any contact. For the D-SUB connector, there were no false positives, but for the Waterproof connector there were 4 false positive terminations out of 17. Additional tuning of the audio termination model can potentially reduce this false positive rate; however, it may not be as fully reliable as a force torque sensor.

3.5.8 Vision Baseline

We also wanted to know how well our vibrotactile outcome detector would compared to a vision only outcome detector. Since we have a side camera and a wrist camera, we evaluated

Prediction Delay	Waterproof	D-SUB
Online Force - Offline Force	0.00538s	0.00535s
Offline Force - Online Audio	0.00398s	-0.00152s
Online Force - Online Audio	0.00936s	0.00383s

Table 3.2: This table illustrates the mean delay between the Online Force Prediction and Offline Force Prediction time stamps as well as the Offline Force and Online Audio predictions times. Thus we can estimate the Online Force and Online Audio Delay by summing the two mean differences together. We had to do this because when the audio termination handler responds before the force termination handler predicts terminate, we are unable to log the result.

Termination Type	Perturb Location		Correct Location	
	Waterproof	D-SUB	Waterproof	D-SUB
Audio	46.371N	65.607N	9.397N	4.5N
Force	46.905N	64.538N	17.884N	7.953N

Table 3.3: This table illustrates the max force that the robot experiences when using the audio and force terminators in either the perturbed location or the correct insertion location.

the performance of a Resnet18 outcome detection model trained on images from each individual camera as well as both cameras together. The results are located in Fig. 3.14. The side camera’s outcome predictor performs poorly on the LEGO blocks because the view is not optimal to see whether a good connection was made for smaller objects as shown in Fig. 3.15. Meanwhile the wrist camera performs much better on average because it is closer to the connector so if there are many features it is better for providing a close-up view of the final insertion pose. However, these experiments were run in an ideal environment with constant lighting, high-contrast colors, and little to no occlusions or obstructions. Having additional blocks or obstacles that block either camera view could cause vision to fail. On the other hand, vibrotactile sensors could work without any light or even under heavy occlusion if provided sufficient training data.

3.6 Conclusions and Future Work

In this work, we have discussed our vibrotactile robotic setup, our data collection pipeline, model learning, and deployment strategies, and finally our results in various online and offline experiments.

Our experiments verified that the Vibrotactile audio-based outcome predictors are invariant to airborne noise, but external vibrations may reduce the accuracy of the prediction depending on how strong or close the vibration is to the contact microphones. In addition, we saw that there may be some slight decreases in performance when the robot is moving at a different velocity compared to the ones used during data collection. When the robot manipulates a LEGO block that was not seen in the training data, the vibrotactile outcome detector was able to extrapolate to the new block without losing performance in two of the four blocks but experienced a loss of around 10-20%

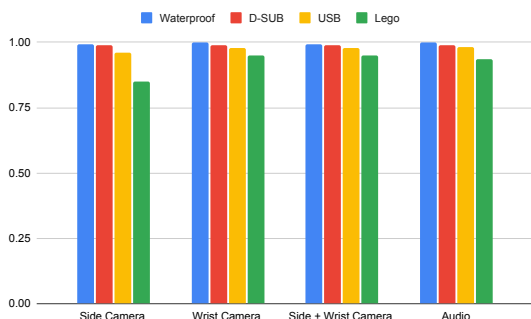


Figure 3.14: Audio vs Vision Results

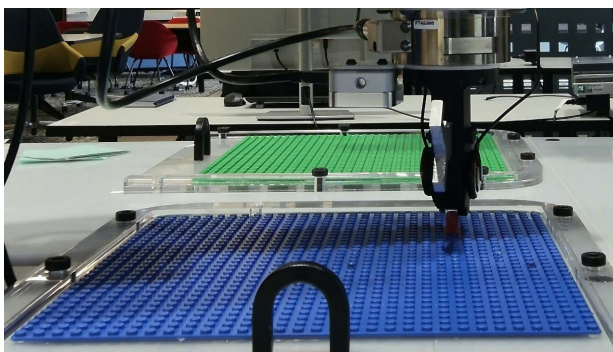


Figure 3.15: Lego Side Camera View

in accuracy with the other two blocks. When we changed the environment around the pick and place location for the LEGO blocks using different constraints, the smaller 2x1 and 2x2 blocks didn't suffer too much until there were 3 constraints because they didn't extend out much from the end effector while the larger blocks such as the 4x1 and 4x2 blocks had large drops in accuracy with as low as 1 or 2 constraints because of the torque caused by moving down when the block is misaligned. Finally, we were able to determine that the vibrotactile audio terminators were able to predict termination approximately 9.4 and 3.8 milliseconds faster than force which results in a decrease in the amount of force experienced by the robot end-effector by around 8N and 3.5N for the waterproof and D-SUB connectors respectively.

We discovered that collecting 300 or more training data resulted in consistently-high performance. When we tested which microphones performed the best, we found out that the microphone at the fingertip was the most important for the LEGO task, and multiple microphones were needed generally needed for the NIST connectors. The optimal microphone setup can thus depend on the specific task. In addition, when we compared our Vibrotactile outcome detector with only a vision-based outcome detector, we saw that the results were comparable, although for the LEGO blocks, the side camera only view was insufficient for reliably predicting the outcome.

Overall we introduce the use of vibrotactile outcome and termination detection so that SMEs can use our findings to help integrate these sensors into their flexible robotic manufacturing tasks. While we only presented work on insertion, we believe that vibrotactile sensing can be utilized for a wide variety of tasks such as verifying whether a connector has successfully clicked into place, if packaging has ripped, or if a cable is being pulled too tightly and is caught on something. However, in these cases the data collection pipeline may need to be modified in order to create a balanced dataset for training, evaluation, and deployment. All in all, we believe that this work provides an insightful foundation for the expansion of vibrotactile sensing as a tool to support the adoption of robots by SMEs for HMLV manufacturing.

Acknowledgements

This work was funded by NIST.

3.6. Conclusions and Future Work

Localization and Force-Feedback with Soft Magnetic Stickers for Precise Robot Manipulation

This chapter is based on [Hellebrekers, Zhang, Veloso, Kroemer, and Majidi, [75]].

Abstract: Tactile sensors are used in robot manipulation to reduce uncertainty regarding hand-object pose estimation. However, existing sensor technologies tend to be bulky and provide signals that are difficult to interpret into actionable changes. Here, we achieve wireless tactile sensing with soft and conformable magnetic stickers that can be easily placed on objects within the robot’s workspace. We embed a small magnetometer within the robot’s fingertip that can localize to a magnetic sticker with sub-mm accuracy and enable the robot to pick up objects in the same place, in the same way, every time. In addition, we utilize the soft magnets’ ability to exhibit magnetic field changes upon contact forces. We demonstrate the localization and force-feedback features with a 7-DOF Franka arm on deformable tool use and a key insertion task for applications in home, medical, and food robotics. By increasing the reliability of interaction with common tools, this approach to object localization and force sensing can improve robot manipulation performance for delicate, high-precision tasks.

4.1 Introduction

Uncertainty affects almost every area of robot manipulation. For example, inaccuracies in determining an object’s position can affect the robot’s ability to execute successful grasps. Seemingly minor pose changes within valid grasps can prevent the robot from properly using tools. Repeating the same pre-programmed path can still include variance due to small errors in the robots kinematics and motion. To reduce uncertainty in the workspace, sensors can be integrated into the robots directly or in the surrounding environment [90, 169]. Vision is the gold standard for finding objects in the workspace [148, 176, 222]. However, robot arms often occlude the manipulation target, increasing uncertainty. In addition, vision-based approaches can often fail to recognize objects due to reflective or transparent surfaces [46]. Multi-camera systems can mitigate occlusion

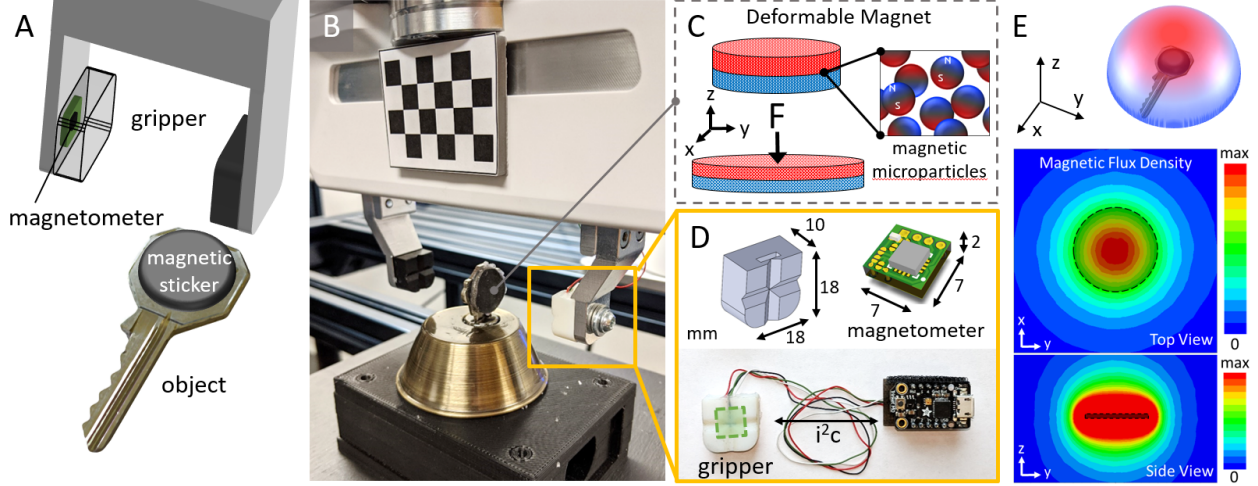


Figure 4.1: A) Overview and B) image of modified Franka Gripper localizing to a soft magnetic sticker on a key. C) By embedding magnetic microparticles in an elastomer, we can create a soft and deformable magnetic sticker. D) Combined with a 3-axis magnetometer inside the gripper, E) the magnetic flux surrounding the sticker can be used for 3D localization, contact detection, and force estimation.

while approaching objects, but often remain limited during manipulation. Still, vision is often able to get the robot in close proximity (1-2cm) of the target object.

To overcome these existing challenges and limitations of vision-based sensing, tactile sensors capable of measuring contact forces are employed for pose estimation and object localization in the presence of occlusions [188, 212, 227]. A tactile sensor aims to convey information about its surroundings through touch. Often, robots have tactile sensors on the robot end-effector to estimate contact location or force. However, these kind of tactile sensors only receive information after the grasp, and do not help during the approach. They are often difficult to integrate into systems, especially if a high spatial density is required. Some vision-based tactile sensors place cameras in the fingers in order to visualize the approach and contact forces [77, 103, 207]. However, these sensors are quite bulky, sample slowly (< 15 fps), and require feature matching algorithms for pose estimation that are still subject to lighting and material conditions.

We propose a tactile sensing system with a magnetometer in the robot gripper and soft magnetic stickers placed in the workspace (Figure 4.1A, B). We are able to encode good grasps, through correct placement of the deformable magnets on objects, and help the robot consistently localize to a repeatable grasp and object pose even before contact. Additionally, the soft magnet provides a soft surface that enables compliant contact as well as contact and force information from magnetic flux changes (Figure 4.1C, E). The magnetometer is available in a very small format (7x7x2 mm), provides fast sampling rates (> 100 Hz), and easily integrates into systems via serial communication (Figure 4.1D). For frequently used objects and precision tasks, the system helps to reduce the uncertainty of contact, grasping, and object pose estimation in a semi-structured environment.

We characterize the proximity and contact signals of the soft magnetic stickers with a precision force measurement stand and 7-DOF robot arm, as well as outline several straightforward algorithms to localize the magnetic sticker. Finally, we present demonstrations that highlight the enabling features of the magnetic stickers for culinary, domestic, and biomedical applications.

4.2 Related Work

Numerous tactile sensors are commercially available with varying principal techniques that provide force, pressure, and other tactile information to robots for precise manipulation tasks [34]. BioTac sensors use conductive fluid and electrodes to determine the pressure on a robot’s fingertips. Optoforce sensors use light emitters and sensors as well as a reflective layer in order to sense 3-axis forces. Tactile sensors also draw from an increasingly diverse number of underlying transduction methods. Liu et al. [107] use small 6-axis ATI Nano17 force/torque sensors underneath spherical fingertips in order to estimate the friction and normal forces of surfaces. Romano et al. [161] use fingertip pressure sensors (TactArrays from PPS) on the Willow Garage PR2 to measure the grip force to lift everyday objects. [95], [40], and [217] use contact microphones near robots’ end-effectors to encode tactile information in the form of audio signals. Capacitive sensors are able to detect both contact for any object and pre-contact information, but only for conductive objects.

Previous work on sensors have expanded their functionality to include object localization for precise robot manipulation. For example, Li et al. use a GelSight sensor to insert a USB by localizing the position and orientation of the USB after the grasp [103]. Yamaguchi and Atkeson [207] use FingerVision in order to both see the external scene as well as sense forces in order to detect slip when cutting vegetables. However, both of these methods are vision-based and therefore suffer from the same issues of durability, bulkiness, lower sampling rates, and onerous data processing. Capacitive sensing is largely limited by object compatibility.

Magnetic transduction methods have also been explored for robot tactile sensors and typically includes a magnetometer in conjunction with a permanent magnet suspended in a compliant substrate [39]. This method has been shown to capture both normal and shear forces upon contact deformation [100, 146, 162]. Magnets have also been used for localization of mobile robots [21, 36, 210] and medical devices [115, 189]. Previously, we demonstrated that elastomers with embedded microparticles can also be used as a soft continuous tactile skin [73, 74].

This work builds upon previous magnetic tactile sensors by expanding their application to include 3D localization. By separating the soft magnet from the magnetometer circuit, the robot is able to move freely and measure the surrounding magnetic flux changes due to both motion and deformation. We characterize how to approximate the magnetic flux to find the centroid of a disc magnet. This system is complementary to vision-based object localization, which is required to get the robot within range of the magnetic field. Once nearby, the magnetometer measures the magnetic flux changes along the robot’s path due to the soft magnetic sticker’s proximity and deformation.

4.3 Background Theory

Localization and force-feedback with soft magnets is governed by Maxwell’s equations for electromagnetism. For magnetostatic conditions, the magnetic flux density \mathbf{B} associated with the magnet can be calculated as $\mathbf{B} = \mu_o(\mathbf{H} + \mathbf{M})$, where μ_o is the permeability of free space, \mathbf{H} is the magnetic field intensity, and \mathbf{M} is the magnetization.

For a point outside of a disc magnet uniformly magnetized in the z-direction, the magnetic field can be calculated by [62]:

$$\mathbf{B}(\chi) = \frac{-\mu_o}{4\pi} \int_{z_1}^{z_2} \int_0^{2\pi} \nabla \left(\frac{\sigma_m}{|\chi_c - \chi'_c|} \right) R d\phi' dz' \quad (4.1)$$

where χ is some point outside the disc in cylindrical coordinates (r, ϕ, z) , R is the radius of the disc, and σ_m is the surface charge density. $|\chi_c - \chi'_c|$ represents the distance between the point χ_c outside of the disc and current point χ'_c inside the disc. Qualitatively, Equation 2 represents adding all the contributions through the magnet's volume at one point outside the disc magnet.

Equation 2 has no analytical solution and must be solved using finite element methods. In order to localize the magnet from the current position, we need the inverse equation. To simplify this model for robotic applications, we chose to roughly estimate the shape of the field above a disc magnet as a 2-D Gaussian (Figure 4.1E). Because collaborative robots are often unable to take precise sub-mm steps, we found good results with this approximation because it provided fast and sufficiently accurate solutions.

Limiting the points along the centerline of the magnet (i.e. $r=0$) leads to an analytic solution to the magnetic field strength due to azimuthal symmetry [25]:

$$B_z = \frac{B_r}{2} \left(\frac{z+T}{\sqrt{(R^2 + (z+T)^2)}} - \frac{z}{\sqrt{R^2 + z^2}} \right) \quad (4.2)$$

where B_z is the magnetic field at some distance z along the centerline of a disc magnet with radius R and thickness T . The remanence field B_r is a property of the magnetic material and can be estimated as follows:

$$B_r = \frac{2\sqrt{R^2 + T^2}}{T} B_z(0) \quad (4.3)$$

where $B_z(0)$ represents the z-component of magnetic field at the surface of the magnet, which can be easily measured with the gripper. For a thickness $T=2$ mm, we found our magnets' B_r can range from 3500 to 4500 μ T and serves as bounding values for a least-squares fit.

4.4 System Overview

4.4.1 Robot

We use a 7-DoF Franka Panda Research arm mounted on a frame with a wooden table surface. The setup contains two Microsoft Azure Kinect cameras (3840x2160 resolution; 0.25-2.21 depth range): one mounted above the robot, which is used to classify objects in the robot's environment, and another mounted on the left side of the setup to give us a vision baseline for localization experiments in subsequent sections. Both cameras are calibrated with respect to the base of the robot and provide RGB and depth images at 15 fps.

4.4.2 Magnetometer

The 3-axis magnetometer (MLX90393; Melexis) is mounted on a custom circuit board (7x7x2 mm) with four input wires for SDA, SCL, 3.3V, and GND. These four wires allow the magnetometer to communicate with a small microcontroller (Trinket M0; Adafruit) attached to the end-effector

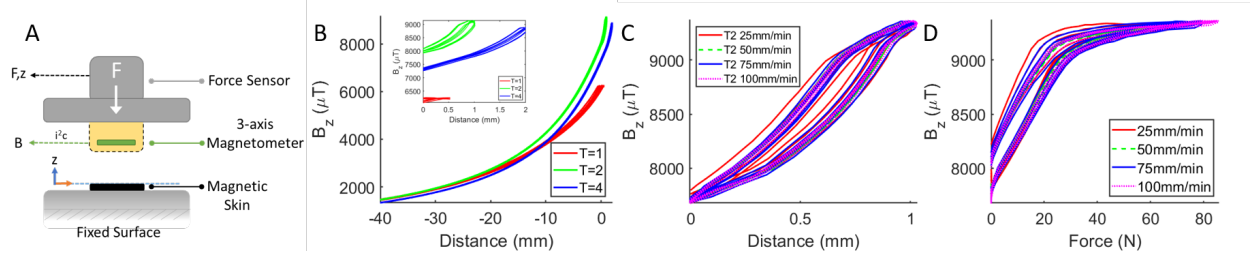


Figure 4.2: A) The experimental setup with a precision force-measurement stand (Instron) that records position, force, and magnetic flux. A Franka gripper was attached to the top plate to mimic the applied pressure from a robot gripper. B) Magnetic Field from 40 mm above the magnet to 1/2 depth of varying thickness. C) For a thickness of 2 mm, the signal change with compression distance up to 1 mm and D) corresponding forces.

using i^2c . The 3-axis magnetometer is inserted into a slot (7x12.5x2 mm) of a modified 3D-printed Franka fingertip (Fig. 4.1D). The slot places the chip at the center of the gripper, 2 mm below the surface. The microcontroller continuously samples the surrounding magnetic field at 100 Hz and sends the data over USB serial to the computer controlling the Franka.

4.4.3 Soft Magnetic Sticker

The soft magnet is fabricated by mixing prepolymers and magnetic microparticles and then curing under a magnetic field. In this work, we mix a silicone elastomer (Ecoflex 00-30; Smooth-On) with magnetic micro-particles ($d \approx 100\mu m$, MQP-15-7; Magnequench) in a 1:1 ratio. Then, we pour the mixture into a 3D-printed mold (Objet30; Stratasys) and degas it in a vacuum chamber for 3 minutes. Next, we place a plastic film on top of the filled mold to push excess mixture out. Finally, the filled mold and film are left to cure on top of a permanent magnet (N48; Neodymium) at room temperature for 2 hours [74]. We adhere the soft magnet to objects using either double-sided tape or a quick-setting epoxy. Ecoflex 00-30, with a Young's modulus of ≈ 100 kPa, was selected for being both soft enough for conformal contact and tough enough for contact displacement detection.

4.5 Experimental Design And Results

4.5.1 Precision Force Measurement Stand

First, we characterize the magnetic signal of the sticker during contact and proximity with a precision force-measurement system (Instron). We adhere the modified Franka gripper with an internal magnetometer to the top plate. Next we place three soft disc magnets ($R=10$ mm) of varying thickness ($T=1, 2$ or 4 mm) on the bottom plate. Then we move the Instron vertically down in the z -axis towards the magnet while continually recording the force and position (Figure 4.2A). We conduct two experiments: one for proximity and contact detection, and another to examine force response during contact.

In the first experiment, we run 5 trials of the Instron starting from a height of 40 mm above the magnet to 1/2 depth of the magnet ($D=0.5, 1$, or 2 mm, respectively). These distances represent

the maximum distance between the Franka grippers and a reasonable compression of the magnets to avoid plastic deformation. As shown in Figure 4.2B, the magnetic flux density increases exponentially across all thicknesses. Once the magnets are compressed (inset), the signal continues to increase for the 2 and 4 mm thick magnets where we observe that a larger compression distance results in a larger signal change. The 1-mm magnet appears to be too thin to give a change in signal during contact.

For the second experiment, we explore how the magnetic signal changes as a function of applied force and distance. We use the $T=2\text{mm}$ magnet due to its combination of conformability and signal strength. We record 5 compression trials from the top surface of the magnet to 1 mm compression at different speeds ($s=25, 50, 75, 100\text{mm/min}$). Looking at the data in Fig. 4.2C and D, the magnet responds similarly across different speeds, and there is a clear hysteresis in the signal response, which is expected for the viscoelastic host material. Furthermore, the signal change is greatest in the range of 0-20 N. For forces $>20\text{N}$, we see only modest change in magnetic field with increased force. We primarily attribute this to the nonlinear stiffness and incompressibility of the elastomer. We also note that magnetic particles settle on one side of the mold due to gravity during the curing process. This results in one side of the magnet with more particles and greater stiffness, contributing to even greater nonlinearity in the force response. This operating range of 0-20N is ideal for complementing the on-board force sensing of robot arms, which generally excel at a larger force range (20-80N), but at the cost of limited sensitivity.

4.5.2 Minimum Contact Estimation

For this experiment, we are interested in responding to the minimum contact the robot can perceive between its gripper and a soft magnetic sticker. This is important because the minimum force that the Franka gripper can sense is 20 N, which would crush fragile or deformable objects. We want to leverage the magnetic signals from the fingertip and soft magnet to stop the gripper when initial contact has been made (Figure 4.3A). We use the same disc magnet of $T=2\text{mm}$ and $R=10\text{mm}$ as in the previous experiment.

We chose three objects to demonstrate this task (Fig. 4.3B): a small metal key, a transparent glass bottle, and a very soft squeeze bottle. They all represent challenges for vision-based grasping approaches, such as being small, transparent, or reflective. The objects are placed in a known location in the robot's workspace, and the robot positions its gripper on the centerline of the magnet. We close the gripper iteratively in small steps ($s=2\text{mm}$) while monitoring the magnetic signal in the z-direction. As the gripper approaches the soft magnet, it is either looking for a large gradient change in the signal or a near zero gradient. When the stopping criteria is satisfied, the robot lifts up the object.

The large gradient change observed in Fig. 4.3C for both the key and glass bottle is a result of the compression of the soft magnet after contact. On the other hand, a near zero gradient change is caused by deformation in the object itself, which results in little compression in the soft magnet and little to no change in the magnetic signal. As a baseline, we allow the gripper to completely close around each object using its internal 20N force feedback, which is indicated by the vertical dotted lines in Fig. 4.3C.

For hard objects like the key and glass bottle, we show that the object can still be lifted without slipping given a wider final grasp location. Across three trials, the glass bottle and key were grasped at positions 1.31 and 1.03 mm wider on average than without magnetic force feedback.

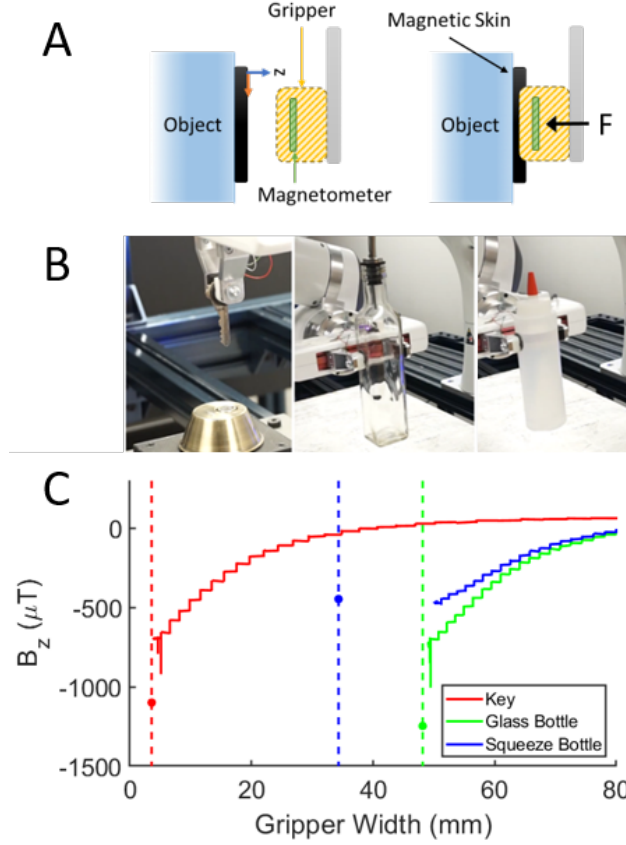


Figure 4.3: A) Overview of experimental setup for minimum contact experiment. B) For a key, glass bottle, and squeeze bottle, C) the signal change during step-wise gripper motions is shown in solid lines, and the final gripper position without magnetic feedback is marked by a vertical dotted line and a dot with the signal at that location.

However, the real strength of our system is highlighted in the deformable squeeze bottle data. With our additional magnetic feedback, the gripper position is 15.79 mm wider on average. In this case, our system makes all the difference between lifting the bottle and crushing it.

4.5.3 Localization

In this section, we explore how the magnetometer fingertip can localize to a soft magnetic sticker without contact. We verify the location with a well-placed camera looking through a transparent plate (Figure 4.4). We present a variety of localization methods in order to adapt to different levels of prior information. For each method, we repeat the magnet localization process 10 times from a *constant* initial starting point and 10 times from a *random* starting point within 2 cm in each axis. We again use the same soft magnetic sticker with $T=2\text{mm}$ and $R=10\text{mm}$. Notably, the localization methods presented below would also apply to rigid magnets, given the magnetometer range is large enough.

In the *ID scan* localization experiments, we assume that only one-axis needs to be localized, and that the robot passes over the magnet during a sufficiently long scan. In these conditions, it is sufficient to collect magnetic data over the scan and store the robot's end-effector position that

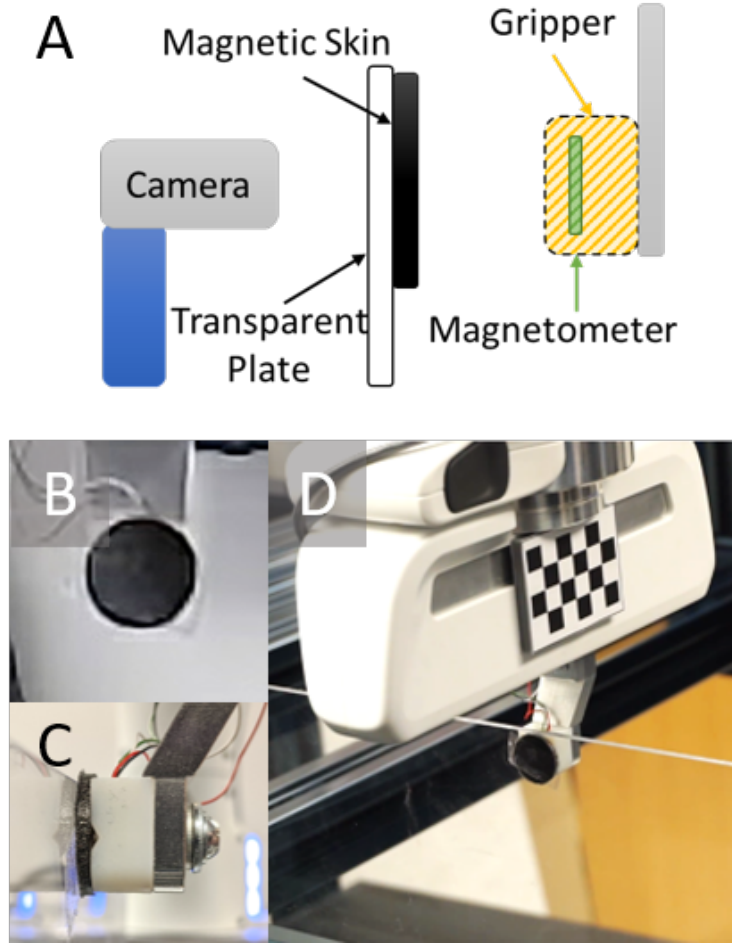


Figure 4.4: A) Experimental setup for localization includes a well-placed camera perpendicular to a transparent plate. B) A magnet is placed on the acrylic surface and the camera is used as the ground truth for the C/D) final magnetic position.

encountered the max magnetic field. Moving to the corresponding robot location will result in centering the magnet in that axis. Similarly, this approach extends to a *2D scan* localization task if the robot knows the direction to travel in 2D to pass over the magnet. By repeating the scan in each axis, the robot can converge to the correct position.

More commonly, vision will be able to place the robot in the proximity of the object and soft magnetic sticker, but is unaware of which direction to scan. In this case, we recommend scanning over a short distance in any direction and fitting a 1D Gaussian to the data points via a non-linear least squares optimization. In our case, we used the curve fit function in the *scipy* optimization library in order to fit our observed magnetic signals B_i at each robot position x_i in an axis to the following Gaussian equation:

$$B_i = B_z e^{-\frac{1}{2}(\frac{x_i - \mu}{\sigma})^2} + B_e \quad (4.4)$$

We optimize for the variables B_z (the maximum z-component of the magnetic signal at the centroid of the magnet), μ (the location of the center of the Gaussian), σ (the standard deviation

of the magnetic signal), and B_e (the ambient magnetic field). It is important to properly initialize each variable. We typically set the starting values to be $400 \mu T$ for B_z , the location of the max value in the scan for μ , the radius R for σ , and around $80 \mu T$ for B_e .

After the optimization is complete, we send the robot to the estimated location of the peak of the Gaussian (μ). We recommend repeating this process twice in each axis because qualitatively, if the gripper starting location is more than 2 cm away, the first Gaussian fit will be inaccurate. When you repeat the Gaussian fit with additional and stronger information from a second scan, the least squares optimizer will be able to more accurately predict the peak location of the Gaussian. Results from repeating this method in 2D are reported as *2D Gauss*.

Finally, once the robot has localized to the central axis, we can use Equation 3 to estimate the location of the surface of the magnet. First, we scan a short distance in the last axis and fit Equation 3 to the data by again using the *scipy* curve fit function. Then, we calculate the distance between the robot and the surface of the magnet, and move halfway there. We repeat this process until the calculated step size is less than 1 mm, which is an appropriate stopping point for the Franka. This experiment is reported at *3D Gauss*.

Starting at the same point in space, all the methods reliably find the centroid of the magnet within a standard deviation of 0.5 mm (Table 4.1). This is comparable to the robot’s motion accuracy, and further improvement in processing may not be possible due to inaccuracies in the robot’s sub-mm movements. With random starting points, the accuracy and precision are still repeatable and reliable. For 3D localization, the moving along the y-axis represents moving towards the surface of the magnet. We observe the largest error and standard deviation in this axis most likely because our magnets are not representative of the model outlined in Equation 3. Each magnet has a unique surface magnetic flux density, as well as a random, nonuniform particle distribution.

Finally, the most important factor of the quality of a magnetic signal is distance. In Fig. 4.5, we compare the start-to-end distance to the final error from the *3D Gauss random* experiment. Marked in pink, we see that starting too close (<5 mm) may lead to undesired edge effects during a scan. On the other hand, starting more than 2 cm away also leads to a large jump in overall error because the magnet has a range of about 2 cm and its magnetic field decays rapidly as distance increases. To reliably start beyond 2 cm, it is necessary to increase the size of the magnet in any dimension, to improve its ‘region of convergence’. Finally, Fig. 4.5 shows that our model for approaching the

Table 4.1: Summary of Errors Over Localization Methods

Method	Error (mm)				
	N=10	X	Y	Z	Overall
1D scan constant	0.6±0.4	—	—	—	0.6±0.4
1D scan random	0.9±0.6	—	—	—	0.9±0.6
2D scan constant	0.6±0.4	—	—	0.6±0.6	0.6±0.4
2D scan random	0.5±0.4	—	—	0.5±0.5	0.5±0.3
2D Gauss constant	0.4±0.1	—	—	0.7±0.3	0.6±0.1
2D Gauss random	0.7±0.7	—	—	0.7±0.5	0.7±0.3
3D Gauss constant	0.3±0.0	1.5±0.5	0.9±0.3	0.9±0.3	0.9±0.2
3D Gauss random	0.3±0.3	1.3±0.4	0.7±0.5	0.7±0.5	0.8±0.2

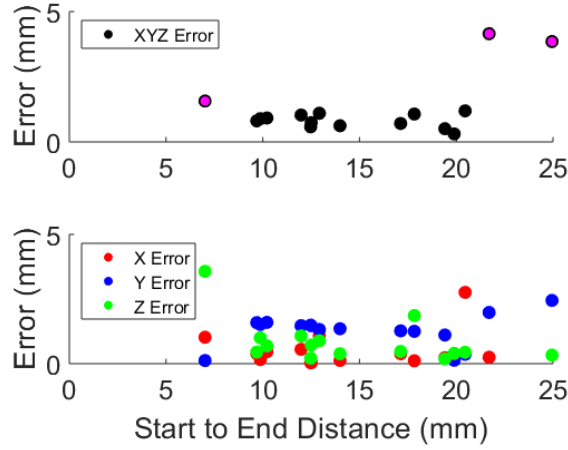


Figure 4.5: 3D Gauss localization results from random start points as function of distance vs error. Outliers marked in pink show increased error due to edge-effects or signal decay over distance.

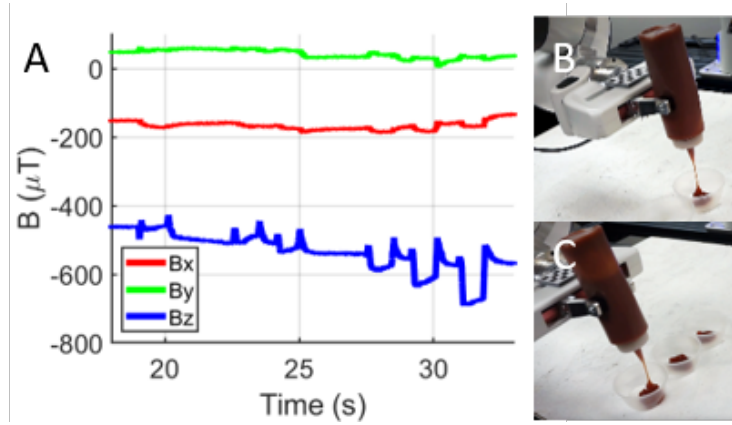


Figure 4.6: Demonstration of squeezing a ketchup bottle. By continuously monitoring the change in magnetic field, we alternate between minimum contact and a 20N grasp. (A) Signal response measurements for (B,C) dispensing different amounts of ketchup into serving cups.

magnet may need to be improved as the y-error is generally higher than the others.

4.6 Demonstrations

In the following demonstrations, we place the soft magnetic stickers on objects that are repeatedly used in the workspace and mimic scenarios that require precise localization and/or grasping while using a tool.

4.6.1 Squeezing Ketchup

In the squeeze bottle demonstration, we combine the earlier technique of grasping the squeeze bottle with minimum contact and couple it with the force feedback from the robot in order to

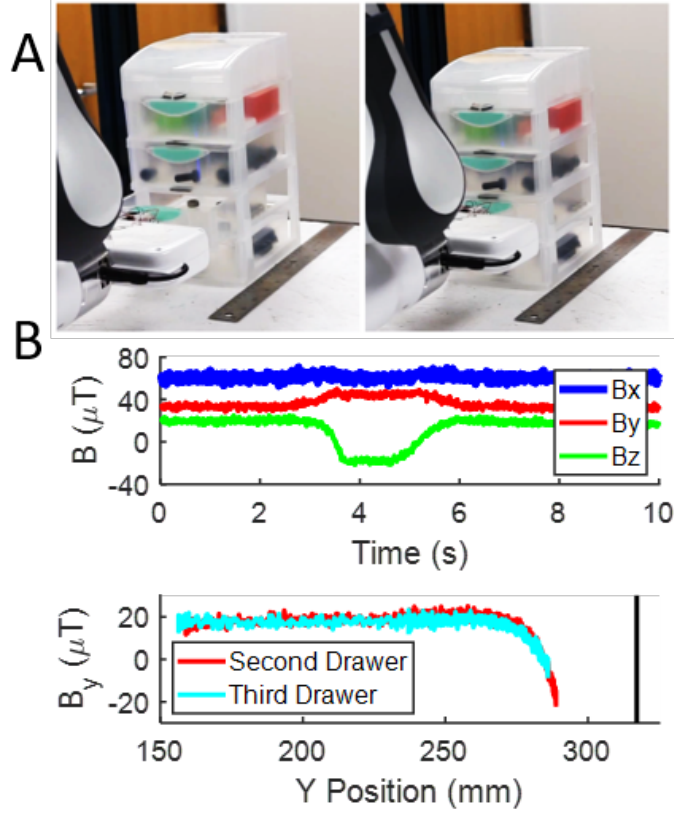


Figure 4.7: Demonstration showing the magnetic sensor used for closing drawers. (A) The robot moves to close the third plastic drawer. (B) When the magnetic signal stops changing, the drawer has been closed. We use this signal to stop the robot before it pushes the entire drawer set backwards, about 2 cm earlier than without feedback marked by the black vertical line.

dispense a more viscous material (ketchup) consistently across three condiment containers. We dispense 1, 2, or 3 squeezes into each container, where each squeeze is approximately 4 grams. Figure 4.6 shows that magnetic flux changes most consistently in the z-axis. This is due to the magnet face being parallel to the z-axis of the magnetometer. Additionally, there is a downward trend in the signal over time. We attribute this to the heavy weight of the bottle that causes slippage during the sequential squeeze tasks. For heavier objects like the ketchup bottle, users can close beyond the minimum contact detection to increase grasp strength and avoid slippage. In the future, it would be interesting to characterize this slip and rotation during the squeezing motion.

4.6.2 Closing Drawers

In this demonstration, the robot is tasked with closing an open drawer in a small light toolbox as shown in Fig. 4.7A. Each drawer contains different objects with varying weights that the robot has no prior knowledge of. The goal is for the robot to close the drawer softly with just enough force so as to not push the entire toolbox. To complete this task, we provide the robot with remote magnet feedback by adhering small rectangular soft magnetic stickers (25mm x 8mm, $T=3\text{mm}$) above each drawer.

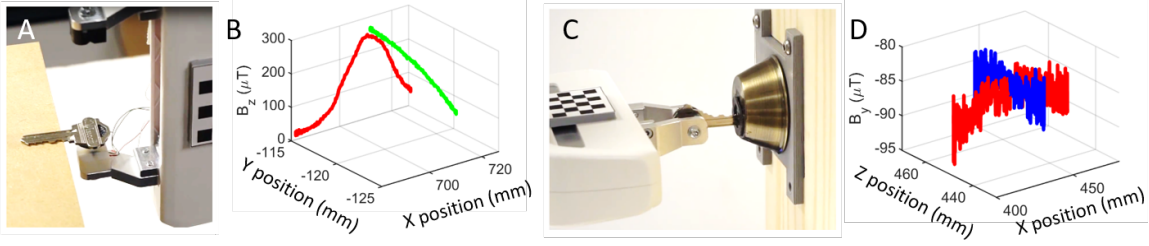


Figure 4.8: A) First, we localize the key for grasping in 2D by scanning in the X and Y axes. B) The intersection of the raw data shows the centroid of the magnet relative to the centroid of the magnetometer inside the gripper C) Second, we align the key to the center of the lock by scanning in the X and Z axes. The intersection represents the center of the lock relative to the centroid of the magnetometer. A constant offset (depth of magnetometer + 1/2 thickness of key + magnet) is added to insert the key and unlock the door.

When the robot is pushing the drawer closed, it is getting closer to the magnet and will continually see a decrease in signal. When it has fully closed the drawer and is starting to push the entire light toolbox, the magnetic signal becomes constant because the distance between the robot’s gripper and the magnet are no longer changing. We program the robot to execute a 8 cm push motion to close the second and third drawers fully. Without the magnetic feedback, the robot is unable to detect the contact of closing the relatively light toolbox and subsequently pushes the entire box backwards until it completes the full 8cm motion. With the magnetic feedback, we are able to detect when the signal stops decreasing and stop the robot before the toolbox moves, regardless of the weight of the drawers (Figure 4.7B).

4.6.3 Unlocking a Door

To highlight the precise localization capabilities of the magnetic sensing approach, we performed a demonstration in which we insert a key into a door knob and unlock a door. We adhered a soft disc magnet ($R=10\text{mm}$, $T=2\text{mm}$) to the end of the key and placed a ring magnet around the key hole ($ID=16\text{mm}$, $OD=26\text{mm}$, $T=1.5\text{mm}$). We begin the experiment by placing the key on the edge of a table that is located close to the door. We train YOLOv3 [156] to recognize the key and lock at different locations in the robot’s workspace. Then we use a depth image from the overhead Azure Kinect camera and the bounding box from YOLOv3 to get the 3D position of the lock and key in terms of the robot’s world coordinates. Afterwards, we command the robot to the key’s location and use 2D Gaussian localization in the X and Y axes to center the gripper with the soft magnet (Figure 4.8A, B). Since the key is rigid and inserting the key requires relatively high forces, we simply grasp it with a force of 20N without feedback from the magnet.

Next, the robot moves with the key to the vision-calculated location of the lock. Since this location is again not precise, we localize in the X and Z axes to move the center of the magnetometer to the center of the ring magnet (Figure 4.8C and D). At this point, the magnetometer itself is aligned at the center of the lock. To align the key, we add a constant offset of 4mm to account for the distance between the magnetometer chip and the gripper (2 mm) and the width of the key and magnet (2 mm). We add this offset to the final calculated location and move the key forward to

insert it. The robot then rotates the key 180 degrees in order to unlock the door (Supplementary Video).

4.6.4 Pipetting

The final demonstration involves dispensing liquid with a pipette. We selected pipetting since it is a very time consuming task that requires very precise control of forces on a lightweight and easily deformable tool. In addition, it illustrates the use of the force response previously shown in the Instron data to enable the Franka gripper to respond to forces < 20 N.

We adhered a rectangular soft magnet around the surface of a 10 mL plastic pipette in the shape of a ring (ID=13mm, OD=17mm, H=14mm). The robot approaches the pipette and closes the gripper in steps until minimum contact. It lifts up the pipette and moves it towards a beaker with blue water as shown in Fig. 4.9. To fill the pipette, the gripper closes completely and then returns to the minimum contact point.

Next, the robot moves to place 5 drops of water in a plastic dish. The relative size of the drops is controlled by the relative increase in magnetic flux. In this experiment, we show three droplet sizes that correspond to 15, 30 and 50 μT increases in signal. Specifically, each position the gripper moves to decreases the signal by the same amount for five consecutive drops. For the medium sized droplets, the five drops correspond to signal changes of 30, 60, 90, 120, and 150 from the minimum contact. The maximum increase is represented by the full compression during refill, which peaks around -200 μT .

Qualitatively, the droplets along each column are approximately the same size and correlate to 1, 2, and 3 drops from the pipette (≈ 0.05 mL, 0.1 mL, and 0.15 mL). Even though the force does not increase linearly with magnetic flux, there is a clear relative pattern that can be relied on. The shape of the magnet is also important. We found that for a ring magnet around a symmetric object, the signal decreases with applied force and distance. In addition, we found that the variance between droplets is increased due to the high-surface tension of water. When the robot attempts to dispense small amounts, the surface tension may unintentionally hold the water outside the pipette tip. This can make the next droplet larger. One way to combat this effect is to select magnetic signal changes that are much larger, or tune the change to the material properties of your fluid.

4.7 Conclusions and Future Work

In conclusion, we have demonstrated a two-part sensing system composed of a 3-axis magnetometer and soft magnetic stickers that can be remotely mounted to objects. By scanning areas of interest, we show the ability to localize the magnet with sub-mm accuracy. Because sensing is performed remotely over a magnetic field, this approach avoids traditional limitations of tactile sensors as well as vision-based problems of occlusion and lighting. By making the magnet out of a soft elastomer, the magnet can adhere and conform to a variety of surfaces and provide information about interfacial forces. The added soft magnets creates a semi-structured environment that enables safe motion and grasping with simple and fast integration. However, magnetic sensing is also susceptible to magnetic noise in the environment.

In this work, the robot always scans in one axis. Moving forward, we will extend the localization technique to allow for linear motion in any plane. This extra degree of freedom will

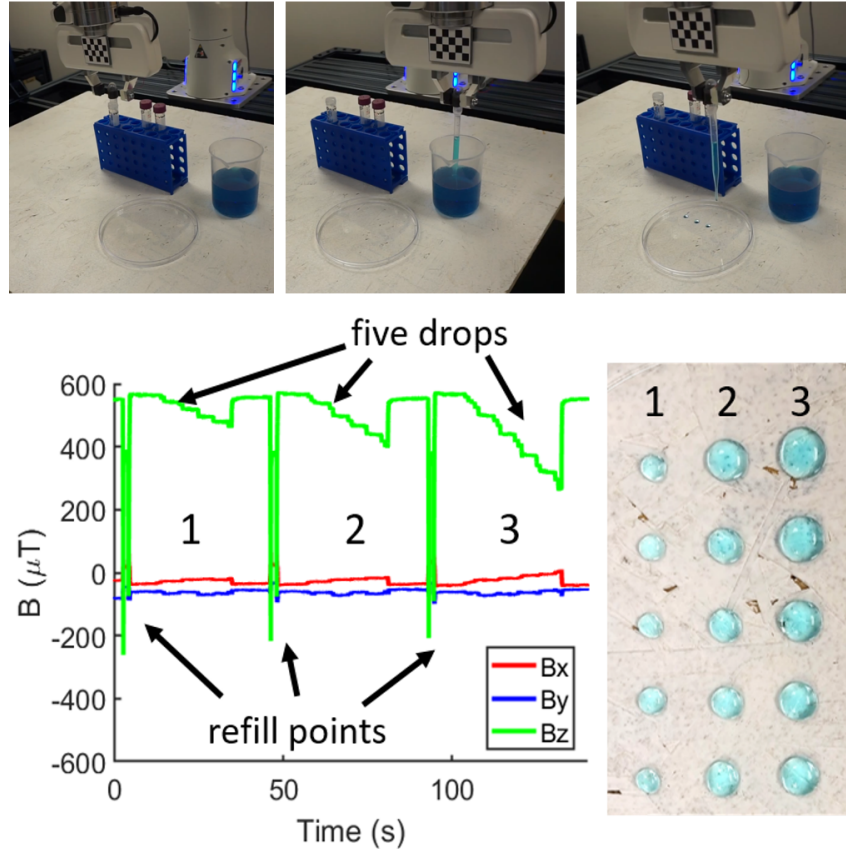


Figure 4.9: By using relative force-feedback provided by the magnetometer and soft magnet, we can fill a pipette with liquid and dispense 5 consistent droplets in sequence with different levels of force.

require filtering out environmental magnetic noise and the added linear offset in the soft magnet's signal. Previously, we have shown that an additional magnetometer board and simple calibration can successfully remove Earth's magnetic field [73]. Variations on this method will be explored to effectively filter these types of additional noise. Finally, this system also has the potential to explore object-environment contacts, in addition to the robot-object contacts explored here, to further inform robot motion. These improvements will help further generalize the technique for use in unstructured environments.

A Low-Cost Compliant Gripper Using Cooperative Mini-Delta Robots for Dexterous Manipulation

This chapter is based on [Mannam, Rudich, Zhang, Veloso, Kroemer, and Temel, [122]].

Abstract: Traditional parallel-jaw grippers are insufficient for delicate object manipulation due to their stiffness and lack of dexterity. Other dexterous robotic hands often have bulky fingers, rely on complex time-varying cable drives, or are prohibitively expensive. In this paper, we introduce a novel low-cost compliant gripper with two centimeter-scaled 3-DOF delta robots using off-the-shelf linear actuators and 3D-printed soft materials. To model the kinematics of delta robots with soft compliant links, which diverge from typical rigid links, we train neural networks using a perception system. Furthermore, we analyze the delta robot’s force profile by varying the starting position in its workspace and measuring the resulting force from a push action. Finally, we demonstrate the compliance and dexterity of our gripper through six dexterous manipulation tasks involving small and delicate objects. Thus, we present the groundwork for creating modular multi-fingered hands that can execute precise and low-inertia manipulations.

5.1 Introduction

In unstructured settings like hospitals and homes, robots require the ability to execute dexterous manipulation tasks like handling delicate and small objects such as pills and coins. Many existing robotic end-effectors are designed for industrial applications where the focus is on repeatable and robust manipulation of large and rigid objects. However, these end-effectors can exert significant forces that can damage smaller, delicate, and nonrigid objects, like berries and playing cards. Interest in soft manipulators has grown recently because of their advantages in safety and compliance [99]. To leverage these desirable properties in dexterous manipulation, we propose a novel compliant gripper composed of cooperative 3-DOF mini-delta robots that are made using soft 3D-printed materials.

Delta robots are highly effective and accurate for pick and place tasks in a variety of industrial

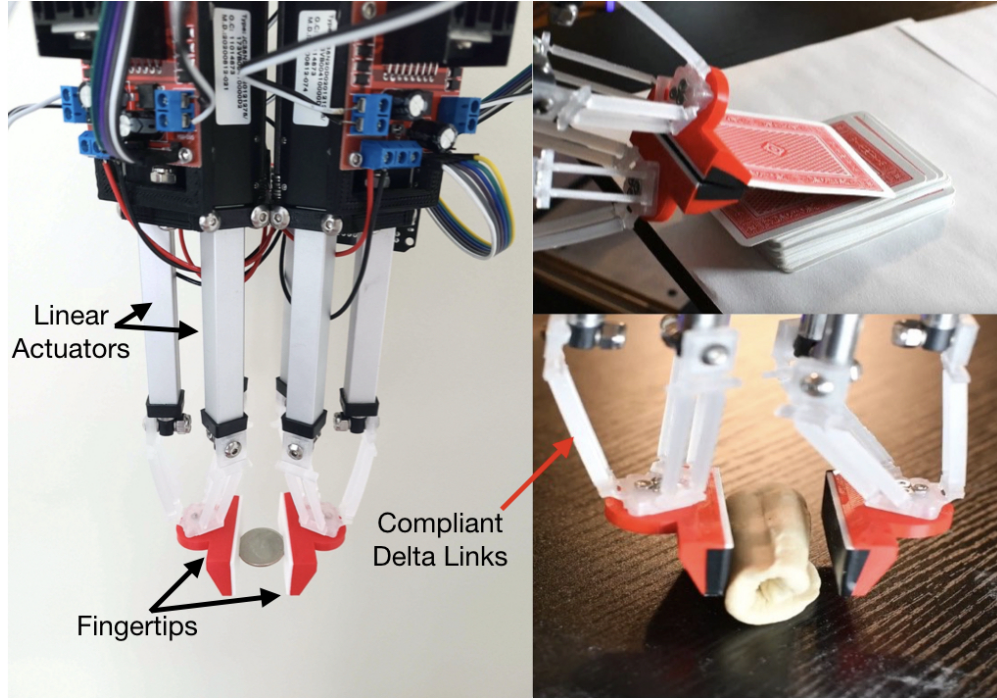


Figure 5.1: Novel robotic gripper grasping a coin, a card, and a dough roll with two delta robots, each with three degrees of freedom and made from 3D-printed soft material, polypropylene (PP). The linear actuators, compliant delta links, and fingertips are shown for a pair of delta robots. The delta robot’s links move up and down with the linear actuators fixed at the joints.

manufacturing and packaging processes [133]. However, utilizing them for other purposes has not been widely studied. In particular, many collaborative robot (cobot) arms used in both academia and industry are outfitted with either two finger parallel jaw grippers or vacuum grippers. Some labs and companies have used significantly expensive and complicated anthropomorphic (human-like) hands such as the shadow hand, which are typically difficult to control autonomously due to their high degrees of freedom [9]. Other researchers have developed their own hands such as soft pneumatic grippers [79] or jamming grippers [7], but they usually have a smaller workspace and less accuracy.

Our gripper, presented in this paper, consists of an end-effector with two 3-DOF delta robot modules as shown in Figure 5.1. In contrast to other grippers, our mini-delta robots use closed-form inverse kinematic solutions and soft materials which achieve high accuracy while still providing compliance. Furthermore, our end-effector is accessible through the use of 3D-printing and readily available off-the-shelf parts. The modular parts can be easily replaced and produced at a low cost. The price for our delta gripper is approximately \$300 (\$150 per 3-DOF finger), which is significantly cheaper than the cost of some off-the-shelf grippers. For instance, the anthropomorphic Shadow Dexterous Hand starts from \$50,000 [172] and Robotiq’s dexterous 3-finger adaptive robot gripper is \$18,000. Even a two-fingered gripper from Robotiq costs around \$5,000 [160]. The dexterous dynamixel claw from BAIR lab [225], which also has 3-DOF per finger, costs around \$2500 for 3-fingered manipulation. From our total delta gripper cost of \$300, the cost of the actuators (\$40 each) is the largest cost out of all the delta robot materials. We therefore consider it a low-

cost gripper when comparing the magnitude of cost with commercial dexterous grippers. Unlike these grippers, the delta robot gripper with its parallel mechanism is compliant, low-cost, easily manufactured, and modular.

Our main contribution in this paper is the design and modeling of a novel gripper composed of 3-DOF compliant delta robots. We start by presenting a design of the gripper and learn its kinematic model using neural networks as the traditional rigid delta model is inaccurate for compliant links. Subsequently, we construct a force profile of the compliant delta robot in various starting configurations using a one-axis force sensor. Finally, we conclude by using the gripper with the learned kinematic model on several dexterous manipulation tasks including manipulating a grape, aligning a pile of coins, picking up one coin from a pile, picking up a card from a deck, plucking a grape off of a stem, and rolling dough. Through these manipulation demonstrations, we present a multi-fingered hand design that can execute precise and low-inertia manipulations.

5.2 Related Work

Delta robots are three translational degrees of freedom (DOF) parallel mechanisms that can be used for manipulation. The advantage of such a robot compared to a serial manipulator is that the inverse kinematics can be computed in closed-form, allowing for fast and easy control. Additionally, the motors are stationed at the base of the delta robot, creating a light end-effector that can move precisely with low inertia. Such low inertia mechanisms combined with compliant materials have a lower chance of harming objects and humans upon interaction. These qualities can be enhanced with the use of soft materials for a safe robotic gripper.

However, soft robots require a departure from classical methods for design, fabrication, and control [164]. The design of our compliant delta robot is similar to the laminate millimeter-scaled delta robot [129] [43] and compliant parallelogram links characterized in [121]. Unique from these two approaches, we use 3D-printing for fabrication and linear actuators to create a prismatic delta rather than rotary motors for a revolute delta. Thus, we use three linear actuators per 3-DOF delta robot to move the end-effectors as shown in Figure 5.1 which behaves practically the same as the revolute delta [20].

Actuation and control of soft robots is an ongoing challenge. Some relevant works have tackled this by using learning-based methods. Truby et. al. [193] use deep learning to map piezoresistive sensor readings to 3D configuration of a complex soft robot. Homeberg et. al. [76] also use proprioceptive sensors but to distinguish objects in-hand. It is common to deduce where the robot is in space through external or internal sensors and in some cases both [183]. In our work, we exploit rigid delta kinematics to build a prior of where the delta end-effector is in space. Then, using a neural network, we create a robust model relating actuator positions to end-effector positions for delta robots of two different soft materials.

In our previous work on 3D-printed compliant delta manipulators [121], we characterized thermoplastic polyurethane (TPU) and polypropylene (PP) parallelogram links for optimal design parameters. The main characteristic of a delta robot mechanism is that the end-effector stays parallel to the base. In our work, we use the optimal dimensions found to maintain this parallel relationship in [121] to design and fabricate a delta robot manipulator using the two materials, TPU and PP.

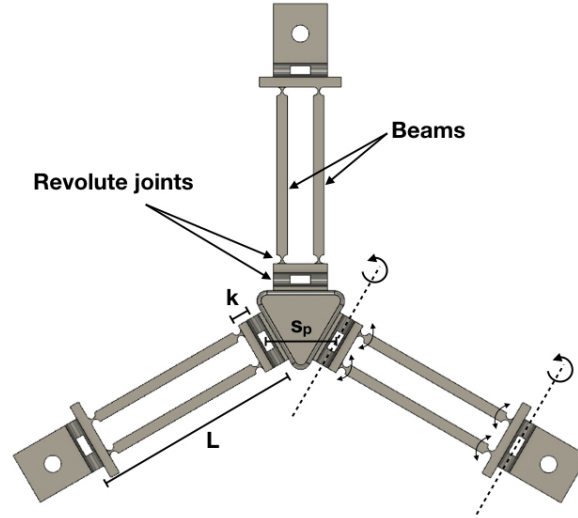


Figure 5.2: This diagram illustrates the two orthogonal axes of rotation that approximate a universal joint and the measurements of k , s_p , and L . k is shown as the distance between the two orthogonal axes of rotation, L is shown as the distance between the axes of rotation at the top and bottom of a leg, and s_p is shown as the distance between the the axes of rotation where the center of the parallelogram links attach to the end-effector. Each measurement represents a distance constraint between revolute joints in our simulation.

5.3 Delta Manipulator Design

Using 3D-printing and soft materials significantly impacted our design of the delta robot gripper. Our compliant delta links made from PP and TPU require a different design when compared to rigid delta links. The delta design, as shown in Figure 5.2, is also dependent on the gripper workspace and actuators. Additionally, fingertips made from PLA are mounted on the delta robot’s end-effector, as shown in Figure 5.1. For comparison, out of the materials used, TPU is the most compliant, followed by PP and then PLA with tensile moduli of 26 MPa (using ASTM D638), 220 MPa (using ISO 527), and 2,346.5 MPa (using ISO 527), respectively [194]. We used an Ultimaker S5 to create all of our 3D-printed parts, but similar fused deposition modeling (FDM) printers are widely accessible, which greatly lowers the barrier of entry to make these low-cost compliant delta robotic manipulators.

5.3.1 Setup

In this work, we test delta robots made from two soft materials, PP and TPU, so we design two compliant delta links accordingly. We use compliant parallelogram links with living hinges to 3D-print delta robots as characterized in previous work [121]. The two living hinges rotate along orthogonal axes to approximate a universal joint in the delta links, as shown in Figure 5.2. Each leg is composed of two beams that move as a four bar parallel linkage mechanism, which transfer the motion from the linear actuators to the end-effector, as shown in Figure 5.1. These parallelograms have 0.375mm hinges, and 2.5mm and 4.5mm thick beams for PP and TPU deltas, respectively.

We use these values to ensure that the delta robot end-effector remains as parallel to the base, throughout as much of the robot's configuration space, as possible [121].

The delta links are attached to the ends of three ECO LLC Mini Electric Linear Actuators with 76.2 mm stroke and 20 N maximum load. The actuators are controlled through ROS serial with an Arduino Mega and L298N motor controllers. The total weight of the gripper with two delta modules is 1.03kg. The gripper is mounted on a Franka Panda Robot Arm for additional mobility during task executions. Further dimensions such as the length of the delta parallelogram beams (as shown in Figure 5.2) and distance between the beams are chosen according to the desired workspace of the delta gripper.

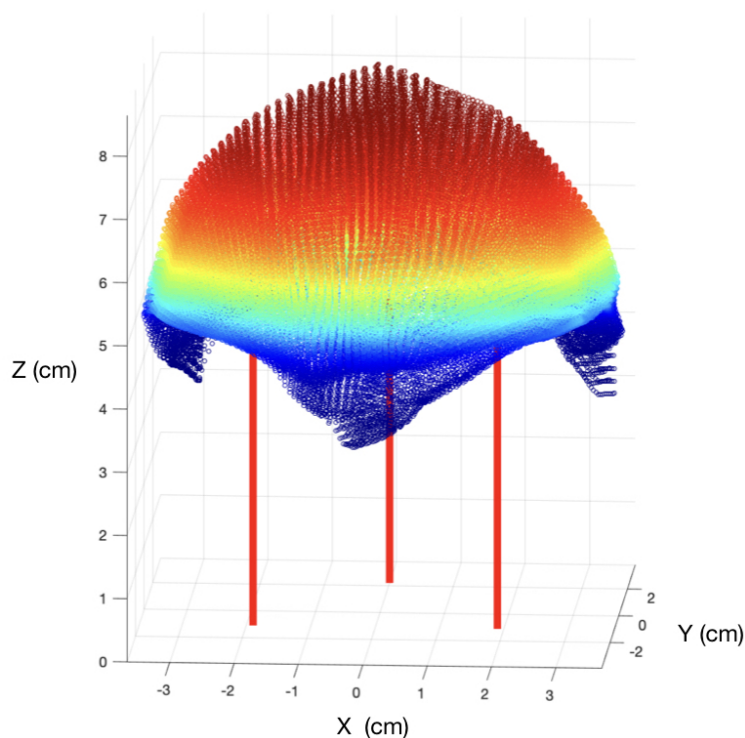


Figure 5.3: The prismatic delta workspace, in centimeters, has a dome-like shape, where the robot can reach any of the points in X, Y, and Z axes. The three vertical red lines represent the position of the linear actuators. Colors are mapped to the height for a better visualization. The actuator lengths are limited to 4 cm so that no actuator can be above the delta's end-effector. Higher z values can be reached by adding a constant offset to every actuator.

5.3.2 Delta Actuators and Workspace

The parallelogram links characterized in our previous work [121] were evaluated on revolute delta robots while our delta gripper uses prismatic deltas actuated by linear actuators. While the delta robot mechanism remains the same, the prismatic delta design allows more freedom in the packing of delta robots. Our gripper only features two delta robots as shown in Figure 5.1, although the design framework can be easily extended to more delta robots adjacent to the existing ones. Revolute delta robots would require more space around the rotary actuators, hence using

linear actuators instead allows us to pack deltas closer together and enable cooperative capabilities between the robots.

The prismatic delta workspace is generally close to a hemisphere shape as shown in Figure 5.3. The workspace shown only accounts for joint angles until the delta links are perpendicular to the linear actuators, as significant deformation of the compliant links will occur after this point. We design adjacent delta robots such that they share a section of their workspace to enable cooperative manipulation of an object. Parameters such as the leg length L , distance between actuators at the base s_b , and distance between where the parallelogram links attach to the end-effector s_p are chosen to create overlapping workspaces between adjacent deltas while taking into account the size of the linear actuators and how close we can pack them. Additionally, we use similar delta structure dimensions characterized in related work [121] except that the parallelogram beams were adjusted to be 6mm apart from each other and 37mm in length. These changes allow for larger joint angles and overlapping workspaces.

5.3.3 Fingertip design

For our delta gripper, we consider two types of fingertips 3D-printed with 10% infill. First, the planar fingertips made from red Tough PLA shown in Figure 5.1 mimic a parallel jaw gripper in that the contact surfaces are flat and opposable. The second fingertip is spherical in shape and made from blue TPU as shown in Figure 5.7(d), making the compliance at the contact points significantly higher at 10% infill. Additionally, the planar fingertips are padded with 2mm thick foam and then electrical tape to increase friction and compliance between the fingertip and object. The main compliance of the gripper comes from the soft delta links as the foam padded fingertips are fully compressed after 1mm of deformation. We exploit the compliance of our delta robots in conjunction with our fingertips to manipulate small and delicate objects dexterously.

5.4 Delta Modeling

There is extensive literature analyzing rigid delta robots [110, 132], but the manufacturing of flexible delta robots through 3D printing introduces significant changes to the kinematics. We approximate each universal joint in the delta with an orthogonal pair of revolute joints that are separated by a small offset k as labelled in Figure 5.2. We analyzed the effect of the offset between revolute joints on the delta workspace by modeling a rigid version of our delta in Simulink. To model the compliant delta kinematics, we use the rigid model as a prior and learn the residual correction using a distal learning approach with data obtained from a marker-based stereo visual tracker.

5.4.1 Offset between Revolute Joints

In order to approximate universal joints, our delta has orthogonal revolute joints that are separated by a small offset. We studied the effect of these offsets on the workspace by simulating a rigid version of our flexible delta robot. We measured four values from our delta robot to parameterize the simulation: leg length $L = 4.8$ cm, distance between linear actuators $s_b = 4.3$ cm, distance

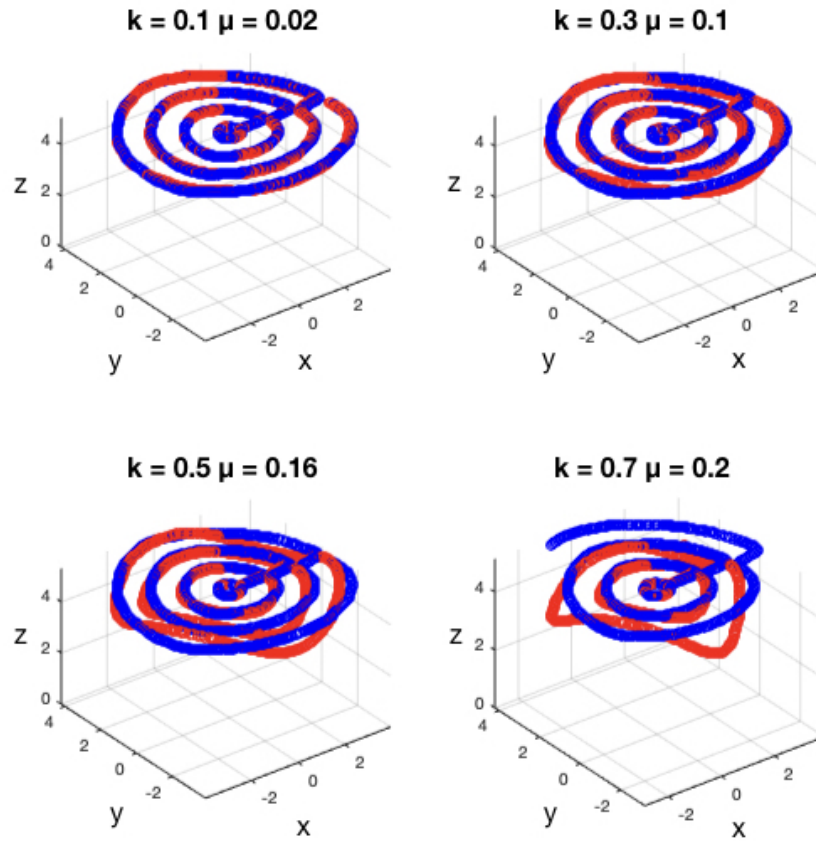


Figure 5.4: Effects of varying the offset k on the mean error μ from the path of a standard delta. The path of the delta with offsets is shown in red, and the path of a standard delta is shown in blue. For $k = .7$ cm, some actuator inputs become infeasible and the path is cut short.

between where the legs attach to the end-effector $s_p = 1.6$ cm, and an offset $k = .5$ cm (see Figure 5.2).

In Figure 5.4, we studied the effect of varying k by measuring the deviation from the kinematics of a standard delta robot on a test trajectory. The test trajectory is a spiral that moves from the center of the workspace to its edge in the XY plane. The z coordinate is selected to maximize the width of the workspace. This path is then representative of the workspace as a whole because all points can be reached at other values of z by adding a constant offset to each of the three linear actuators. Figure 5.4 shows that for a large k , some actuator inputs become infeasible when the delta is near the edge of the workspace. Our value of $k = 0.5$ warps the standard delta workspace by an average of .16 cm on our test path, and does not cause any actuator inputs to become infeasible.

The most significant source of error when compared to the standard rigid delta is the deformation and twisting of the links. Each revolute joint in the delta applies a torque towards its rest position when actuated. In a rigid delta, this force is counteracted by the actuators, but for flexible deltas, equilibrium can be reached by changing the shape of the robot itself. Therefore, deformation occurs as joint angles become large. There are multiple ways that the robot is able to deform, and the type of deformation that occurs is determined by the robot’s design parameters. Since we chose a high ratio of beam link width to joint hinge width in the parallelograms, the links themselves do not bend significantly. Instead, the revolute joints are able to twist a small amount, which can lead to large changes in the position of the end-effector.

5.4.2 Learning Delta Robot Kinematics

We used a marker-based stereo perception system to track the position of the real world delta and collect data to learn the forward and inverse kinematic models for the flexible delta robots as shown in Figure 5.5. To acquire the accurate kinematics, we trained a neural network that was pretrained to match the kinematics for our rigid delta model. We trained two different neural networks to model forward and inverse kinematics. Each network has 3 densely connected ReLU layers with 256 hidden units each and linear activation at the output.

The networks were trained using a distal teacher approach [88] where the forward kinematic model was trained to match input actuator positions with measured end-effector positions, and the inverse kinematics model learned inputs to the forward network that would reduce the error between its prediction and the measured position of the robot. This structure ensures consistency between the network outputs, and it makes it easy to identify and target areas where the kinematics are not well-known because the two networks will produce conflicting results. The training data was also augmented using the symmetry of the delta robot. Observed end-effector positions were copied and reflected over the y -axis, corresponding to switching the heights of the two rightmost actuators in Figure 5.5. The actuator and end-effector positions for each observed point were also rotated by ± 120 degrees about the z -axis. Finally, each observed end-effector position was given multiple z offsets, corresponding to adding a constant to the height of every actuator. These changes ensured that the learned workspace would be symmetrical.

Separate models were trained for both the TPU and PP deltas. The pretrained rigid delta model network had a mean error of 1.3cm and 0.72cm from the test path for TPU and PP deltas, respectively. To improve our pretrained network, it was sufficient to teach the network the flexible delta kinematics by running 100 trajectories that took approximately 20 seconds each. The models were then evaluated based on accuracy and repeatability when following a test path 50 times, as

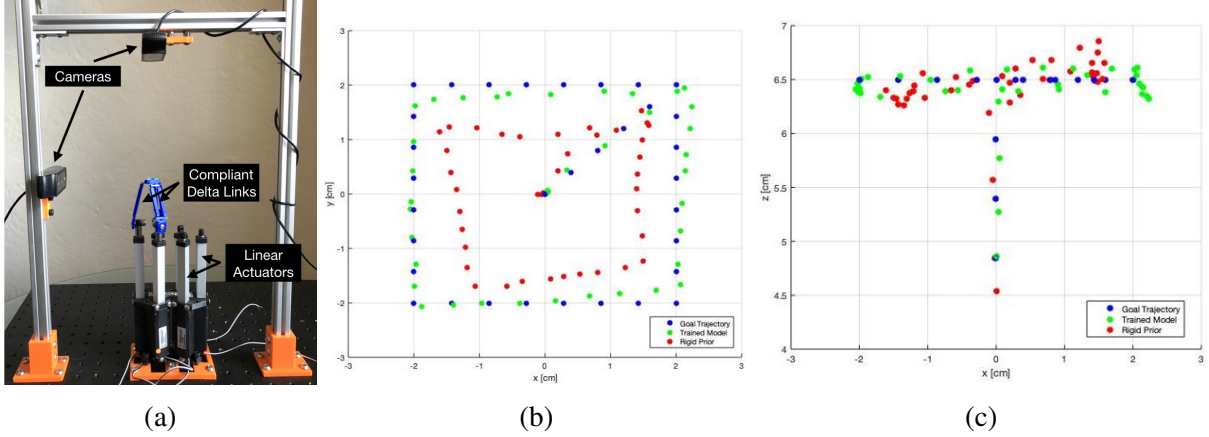


Figure 5.5: (a) Marker-based stereo perception system to track the delta end-effector position using two orthogonal Logitech C920 cameras. The graphs compare measured and desired delta positions on the (b) XY and (c) XZ plane when following a test path (shown in blue) with the PP delta. The neural network trajectory prior to training is shown in red, and the trajectory after training is shown in green.

shown for PP deltas in Figure 5.5(b)-(c). After training on the TPU delta data, the mean error from the goal path was 0.33 cm, and the mean pairwise error over 50 trajectories was 0.13 cm. The mean error from the goal path for the PP delta was 0.28 cm, and the mean pairwise error over 50 trajectories was 0.09 cm. By fitting models to each type of delta, we were able to decrease the kinematics model error significantly for both deltas and confidently deployed them during our robot experiments.

5.5 Force Profile

As discussed in Section 5.4, deformation of the delta links occurs as joint angles become large, which happens towards the edges of the workspace. To determine whether this effect weakens the payload capacity of the delta in certain configurations, we displace the end-effector by a fixed distance along X and Y axes, and measure the resulting force to create a force profile of the delta.

For a given end-effector position (X, Y, Z_1) , another end-effector position (X, Y, Z_2) may be achieved by offsetting all three of the linear actuators, as shown in Figure 5.6(a), by $Z_2 - Z_1$. We selected the Z value that maximized the width of the delta workspace and calculated the force exerted by the delta gripper on the resulting XY plane. Any force measurement at a point (X, Y, Z_1) is representative of the delta's force output at any other point (X, Y, Z_2) . We sampled the x-axis in 4mm increments and the y-axis in 5mm increments from the point $x = -1.5\text{cm}$, $y = -3.6\text{cm}$ to the point $x = 1.5\text{cm}$, $y = 3.6\text{cm}$. Accounting for symmetry, only positive x-axis values are taken into consideration for measurements.

To test the force at a certain position of the workspace, the delta robot end-effector with a planar fingertip was moved to contact the load stem of a GSO-500 Transducer Techniques Load Cell. Then, we recorded the blocked force exerted by pushing the delta at the center of the fingertip surface 1,2,3,4, and 5mm into the load cell, as shown in Figure 5.6(a). Testing both TPU and PP delta robots, we observed that the force exerted grows linearly with the increased displacement. To

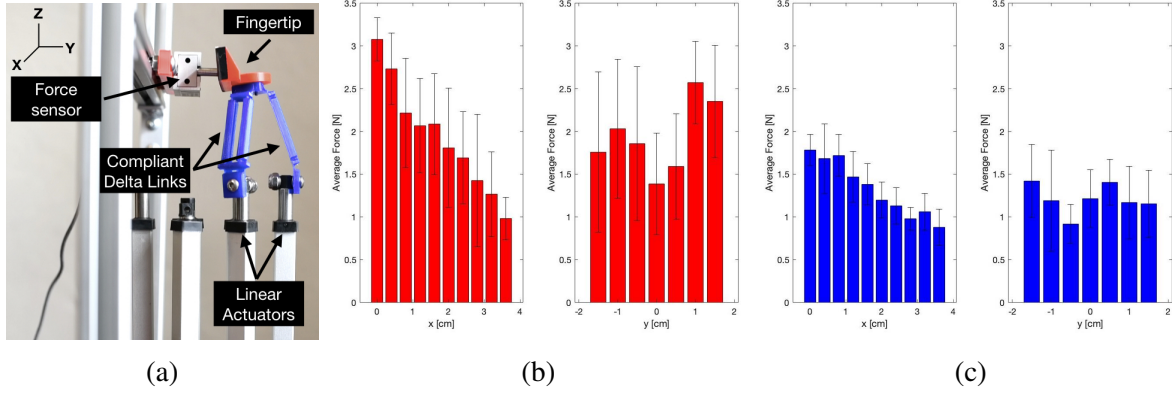


Figure 5.6: (a) Force profile experiment setup consisting of a GSO-500 Transducer Techniques Load Cell and TPU delta robot with planar fingertip. The delta robot pushes on the load cell with a displacement of 5mm at various positions in the workspace, along X and Y axes. The mean force exerted by the delta at different values of x and y, and standard deviation are shown for the (b) PP deltas (c) TPU deltas.

measure the linearity, we calculated the R^2 value after linear regression for the five force measurements at each coordinate. On the TPU delta, the mean R^2 value across all measured points was 0.9541, with a standard deviation of 0.1183. The mean R^2 value for the PP delta was 0.9730 with a standard deviation of 0.0393. This linear relationship allows us to control the force exerted by the delta robot through its displacement.

We grouped the data based on the x and y coordinates, and reported the mean blocked force when displacing the delta 5mm in the direction of the load cell. Figure 5.6(b)-(c) shows that increasing the value of x (moving parallel to the plane of the fingertip away from the center of the workspace) decreases the force output of the delta end-effector. There is no clear trend between the y coordinate of the delta and the mean force that can be exerted. This may be due to the orientation of the delta end-effector changing as it moves forward or backwards along the y-axis.

5.6 Experiments

As a result of our work in Section 5.4, we are able to execute delta manipulator trajectories with precision. To further test the manipulator, we evaluate the success of manipulating various small objects with open loop control or human teleoperation using a PS4 Dualshock Controller. The six tasks we executed are as follows, 1) in-hand manipulation of a single grape, 2) aligning a pile of coins, 3) picking up a coin and rotating it in-hand, 4) slide-to-grasping a card from a deck, 5) twisting a grape off of its stem, and 6) rolling up dough between the fingers on a table. Unlike rigid manipulators, our soft delta gripper can exploit contacts, similar to a human hand, to execute tasks precisely. We chose these tasks to demonstrate the compliance of the deltas and their ability to manipulate delicate objects. While existing grippers may be able to execute these tasks using additional DOFs, we present a unique gripper that can perform all six tasks as a proof of concept. Due to our force profile experiments in Section 5.5, we used the PP delta in all of our demos due to the higher force it can exert. All of the demos are shown in the accompanying supplementary

video¹.

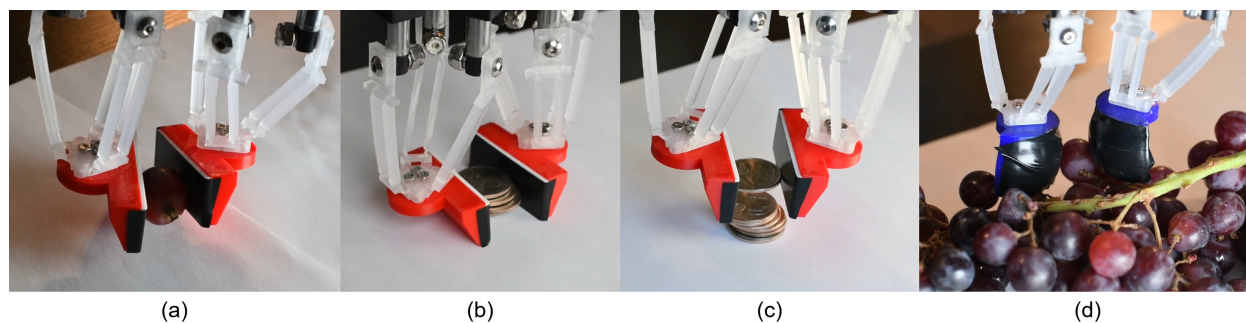


Figure 5.7: Panels (a), (b), and (c) show the delta grippers with planar fingertips 3D-printed using PLA grasping a grape, aligning a pile of coins, and taking a coin from that pile and rotating it in hand, respectively. Panel (d) shows the deltas with spherical TPU fingertips picking a grape from its stem.

In Figure 5.7(a), we show the delta gripper using planar fingertips to grasp a grape. Even when the delta robots use their maximum force to squish the grape, the compliance in the deltas prevent it from being crushed. Instead, the delta fingertips twist, while still holding on to the grape.

Next, in Figure 5.7(b), the deltas arrange a pile of coins by executing two parallel grasps that are orthogonal to each other. While this demo could also be completed by a parallel jaw gripper, there is a chance that the coins would fly out of the gripper if too much force was exerted on the pile. Our deltas gently align the pile of coins in order to create the precisely aligned pile. After the gripper aligns the coins in the pile, it grasps the top coin as in Figure 5.7(c) and is able to rotate the coin in hand. This task illustrates the ability of our robot to move in an additional axis that normal parallel jaw grippers cannot.

Using spherical TPU fingertips the delta gripper picks a grape off of a stem in Figure 5.7(d). Taking a grape off a stem requires the robot to twist the grape in order to apply the necessary pressure on the stem to get it to release without damaging the grape. The spherical fingertips allowed the grape to roll in between the fingertips, resulting in a twisting motion. Afterwards, the robot was able to remove the grape from the stem. This motion required human teleoperation as it involved positioning the fingertips so as to not allow other neighboring grapes to impede the motion of the deltas.

The final two tasks are illustrated in Figure 5.8. In the card pickup task, the top delta robot uses a stroking motion in order to slide the top card from the rest of the deck. Afterwards, the bottom delta lifts up and pinches the card together with the top delta to pick up a single card. The slide-to-grasp motion is made possible by the gripper’s additional degrees of freedom and compliance. The sliding motion was programmed to execute autonomously, although it heavily depends on the initial positioning and orientation of the deltas relative to the cards.

The last task involved rolling a flat piece of dough into a spiral roll. This task also required human teleoperation due to the inherent compliance of the dough itself. One fingertip was used to mainly hold the dough in-place while the other was executing a scooping motion in order to get under the dough and push it. Without the degrees of freedom provided by the deltas, this task would likely be difficult for most grippers.

¹<https://youtu.be/yciJn3rgFHw>

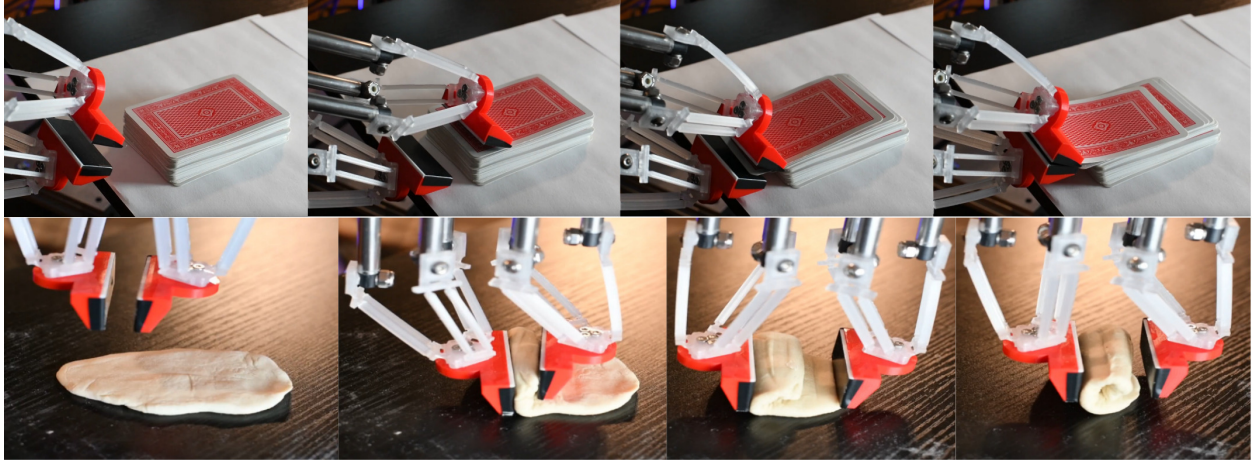


Figure 5.8: Timelapse of the compliant delta gripper sliding the top card and picking it up from the deck, and rolling a flat piece of dough into a spiral.

Throughout all of the demos, the compliance of the deltas and added degrees of freedom enabled a wider range of motion that normal parallel jaw grippers would not afford. In addition, as we had the inverse kinematics for the delta robots, we were able to quickly translate a desired trajectory into commands to the linear actuators. This direct mapping allowed us to easily teleoperate the robot with a PS4 Controller to complete tasks that would typically require a motion tracking hand setup in order to give the robot demonstrations [70]. In the future, we plan to explore more delicate and dexterous tasks with added sensors to provide feedback when interacting with objects.

5.7 Discussion and Conclusion

Through kinematic modeling, force profile characterization, and manipulation task executions, we explored the capabilities of compliant delta grippers made from two soft materials, TPU and PP. While the two materials vary significantly in compliance, the learned kinematic models for the TPU and PP deltas did not differ significantly in performance. Additionally, the force profiles were similar in their ability to exert maximum forces at the center of the workspace. We expect similar trends to extend to delta robots made from materials similar to TPU and PP.

Our kinematic model learning error and force profile experiments show that the delta gripper is easy to control. The robot experiments show that dexterous manipulation tasks such as rolling dough and picking a grape off its stem can be executed with the degrees of freedom provided by each delta. Additionally, the compliance of the delta robot avoids damaging items like the grape. Thus, we can ensure that the delta robot can manipulate delicate objects and interact with its environment safely.

We present the groundwork for creating multi-fingered hands that can execute precise and low-inertia manipulations. Future extensions of this work can explore grasp planning and increasing the number of cooperative deltas to handle larger objects. In addition, we plan on incorporating internal and external sensors to the deltas in order to use visual and haptic feedback for more precise autonomous manipulation.

Acknowledgments

The research presented in this work was funded by the National Science Foundation under Grant No. CMMI-2024794, Sony Group Corporation, and an Amazon Research Award. The research above does not reflect the opinions of our sponsors.

CHAPTER 6

DELTAHANDS: A Synergistic Dexterous Hand Framework Based on Delta Robots

This chapter is based on [Si, Zhang, Kroemer, and Temel, [177]].

Abstract: Dexterous robotic manipulation in unstructured environments can aid in everyday tasks such as cleaning and caretaking. Anthropomorphic robotic hands are highly dexterous and theoretically well-suited for working in human domains, but their complex designs and dynamics often make them difficult to control. By contrast, parallel-jaw grippers are easy to control and are used extensively in industrial applications, but they lack the dexterity for various kinds of grasps and in-hand manipulations. In this work, we present DELTAHANDS, a synergistic dexterous hand framework with Delta robots. The DELTAHANDS are soft, easy to reconfigure, simple to manufacture with low-cost off-the-shelf materials, and possess high degrees of freedom that can be easily controlled. By leveraging hand synergies, DELTAHANDS' dexterity can be adjusted for different applications with further reduced control complexity. We characterize the Delta robots' kinematics accuracy, force profiles, and workspace range to assist with hand design. Finally, we evaluate the versatility of DELTAHANDS by grasping a diverse set of objects and by using teleoperation to complete three dexterous manipulation tasks: cloth folding, cap opening, and cable arrangement. We open-source our hand framework at <https://sites.google.com/view/deltahands/>.

6.1 Introduction

As robots advance closer to assisting humans at home, the design of their end-effectors becomes crucial in ensuring safety and functionality for complex household manipulation tasks. While parallel jaw grippers are widely used for industrial tasks due to their simple one degree of freedom (DoF) control, many household tasks require hands that are dexterous and can safely adapt to various objects. Anthropomorphic robotic hands [9, 48, 49, 152] possess dexterous fingers

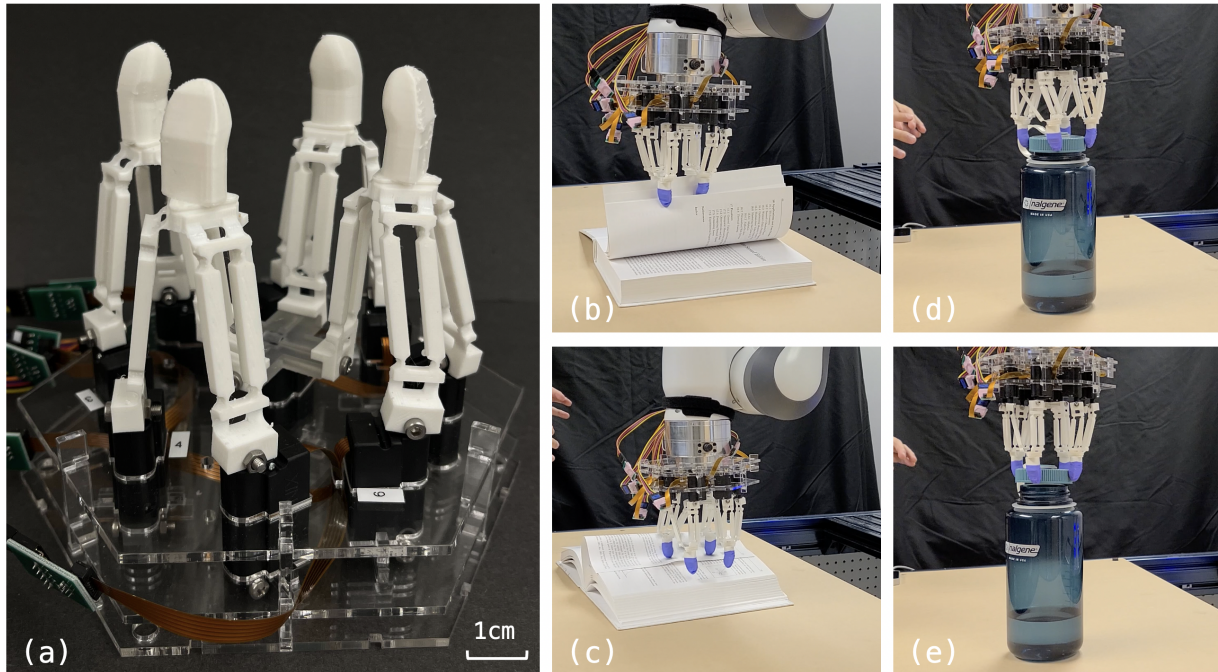


Figure 6.1: (a) DELTAHANDS is a synergistic soft dexterous hand framework based on Delta robots. We demonstrate dexterous manipulation tasks such as (b)(c) turning the pages of a book and (d)(e) opening the cap of a water bottle through human teleoperation.

capable of accomplishing complex in-hand manipulation tasks; however, they tend to be difficult to control, bulky, and costly to manufacture. We aim to introduce a framework for developing hands that are easy to configure, build, and control for dexterous manipulation.

To achieve this goal, we need highly dexterous modular fingers. We propose utilizing linear Delta robots, which consist of three parallel actuators connected by soft 3D-printed links and have three translational DoF with closed-form kinematics. Multiple soft linear Delta robots are arranged in a planar layout as fingers to create dexterous hands with parallel fingertips. The parallelism and translational movements allow for easy control of the fingertips, while the soft links provide a level of compliance. To be adaptable for different applications, we define a flexible design space that allows us to quickly and easily configure hands with different numbers of fingers and layouts, as well as different finger dimensions, workspaces, and force profiles. To reduce control complexity, we introduce hand synergies inspired by previous works [38, 51, 165] that adapted human postural synergies to robotic hands. Due to the inherent parallelism and modularity in linear Delta robots, we can easily configure different hand synergies by combining actuators across different fingers.

We present DELTAHANDS, a synergistic robotic hand framework for dexterous manipulation. By using modular Delta robots, it is easy to configure and manufacture DELTAHANDS with low-cost components and off-the-shelf materials under \$800 in less than a day. In addition, DELTAHANDS can compliantly interact with the environment due to its soft Delta links. We provide the workspace range, kinematics accuracy, and force profile characterization data of Delta robots, which can guide the reconfiguration process for specific applications. We successfully demonstrated grasping daily objects using hands with two different synergies and teleoperating a hand with a robot arm for dexterous manipulation tasks. Our results show that we can easily con-

trol DELTAHANDS by leveraging synergies while maintaining the necessary dexterity. Our main contributions are:

- Hand design, manufacturing process, and simulation with configurable synergies in Section 6.3.
- Kinematics, workspace, and force profile characterization of 3D-printed soft Delta robots to assist configuring hands in Section 6.4.
- Dexterous grasp evaluations and manipulation demonstrations using teleoperation in Section 6.5.

6.2 Related Work

6.2.1 Robotic hands

Robotic hands have a variety of form factors ranging from anthropomorphic hands [42, 142] to underactuated hands [112, 126], from rigid [83, 159] to soft [1, 134], from tendon driven [55, 112] to pneumatically actuated [48, 49], and from 3D-printed [14, 113, 208] to molded [152]. However, most of these hands are difficult to reproduce or modify such as varying the number of fingers and DoF due to their complicated designs and kinematics. For instance, hands with tendon-driven mechanisms [26] might require an entirely new wrist design and cable routing. In contrast, DELTAHANDS supports a wide range of configurations including the number of fingers (1 – 6), actuators (3 – 18), and DoF (3 – 18), along with various synergies. The reconfiguration of hands can be easily realized by updating the acrylic board frames and actuation coupling bars, and reusing all other components as shown in Fig. 6.2. DELTAHANDS, to the best of our knowledge, is the most flexible hand framework.

6.2.2 Delta robots

Delta robots [41] have been deployed for industrial pick-and-place tasks [80] and explored in research [52, 129, 144] because of their high speed, low inertia, and accurate kinematics. With these advantages, researchers have started using Delta robots as fingers for dexterous manipulation. A two-fingered six-DoF gripper with two Delta robots [122] and a large array of 64 Delta robots [145] were introduced for dexterous and distributed manipulation. Similarly, we adapt the modularized soft Delta robots as fingers for dexterous hands. We further extend it to a hand design space and propose hand synergies to reduce cost and control complexity. DELTAHANDS are optimized for dexterous manipulation with simple control, rather than forceful grasps that can be achieved by tendon-driven mechanisms [113].

6.2.3 Robotic hands for dexterous manipulation

Most soft robotic hands, especially anthropomorphic hands, are evaluated with a grasp taxonomy [56]. Several robotic hands are demonstrated with in-hand object repositioning [1, 44, 152]. We will show that DELTAHANDS can perform versatile motions for grasping, in-hand object

repositioning, and manipulating household objects. Towards learning for dexterous manipulation, ROBEL [3] was presented as an open-sourced platform for reinforcement learning in the real world. We provide models of DELTAHANDS both in the real world and in simulation which can be leveraged in the future for efficient skill learning. Due to dexterous hands' high dimensionalities, the control difficulty became a primary blocker for autonomous manipulation. Postural hand synergies [165] were studied and used to reduce the dimensionality and thus simplify control [38]. Researchers also adapted the synergies to hardware design [26, 51]. DELTAHANDS can support various software and hardware synergies based on the parallelism and modularity of linear Delta robots, and we demonstrate it with teleoperation tasks.

6.3 Methodology

DELTAHANDS is a synergistic robotic hand framework to accommodate different manipulation applications. We use Delta robots as the DELTAHANDS' dexterous fingers which have three DoF each with simple kinematics. We provide the flexibility of dexterity by introducing actuation synergies to constrain the motions of fingers for easier control. We provide simulations of DELTAHANDS for exploring design parameters.

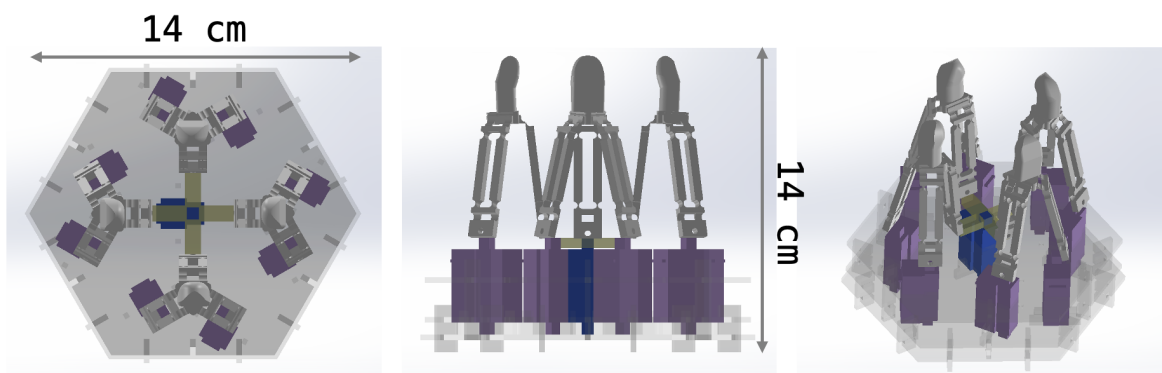
6.3.1 Hand design and manufacturing

The structure of the hand is modularized to fingertips, fingers, actuators, and frames from top to down as shown in Fig. 6.2. This greatly simplifies hand manufacturing and re-configurations by simply modifying or replacing certain components. We 3D print soft fingertips with thermoplastic polyurethane (TPU) (shore hardness 95A) on an Ultimaker 3D printer, and use a pre-bent human fingertip-like shape for better small-object grasping. We adapt the compliant design of soft Delta links from [120] as fingers and 3D print them with the same TPU material. We use linear actuators (Actuonix PQ12-63-6-P) with a stroke length of 20 mm to minimize the footprint of the Delta robots for compact hand design and actuation synergies.

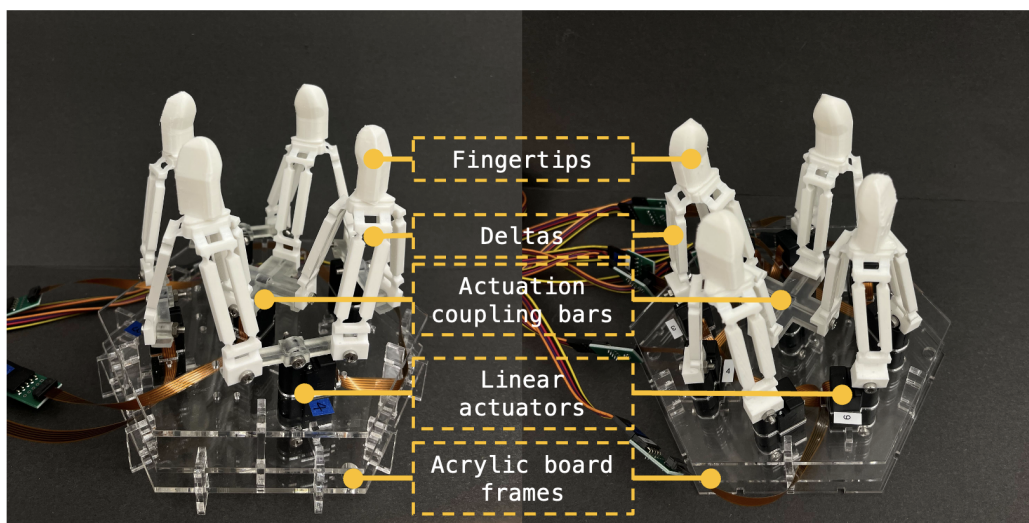
As a design choice, we use four fingers arranged in a square shape to maximize the workspace of the hand. But, we can easily reconfigure the hand to one to six fingers by modifying the layout of the acrylic frames. We laser-cut three layers of acrylic boards as frames to hold the actuators in certain configurations, and vertical fixtures to fix and reinforce the structure on the sides. Each individual actuator can be easily inserted and replaced without disassembling the hand structure. We show two hand prototypes in Fig. 6.2 (b). The size of a hand is 140 mm \times 140 mm \times 140 mm, which is similar to the human hand length (around 180 mm). They weigh in the range of 360 g to 430 g which is smaller and lighter compared to most dexterous robotic hands [113] given their high DoF.

6.3.2 Actuation coupling

A Delta robot has three DoF when it is actuated independently. However, in most manipulation tasks, motions are usually coupled through muscle synergies to create more coordinated movements between fingers [165]. For robotic hands, along with *software synergies* at the control level, *hardware synergies* can be realized at the mechanical level [26]. Our hand framework can



(a) Top, front, and 3D view of the CAD model for a 9-actuator Delta hand.



(b) 5-actuator Delta hand (c) 9-actuator Delta hand

Figure 6.2: (a) CAD model for a 9-actuator hand where the inner actuators of four Delta robots are coupled as one center actuator (blue). Prototypes of (b) 5-actuator and (c) 9-actuator hands.

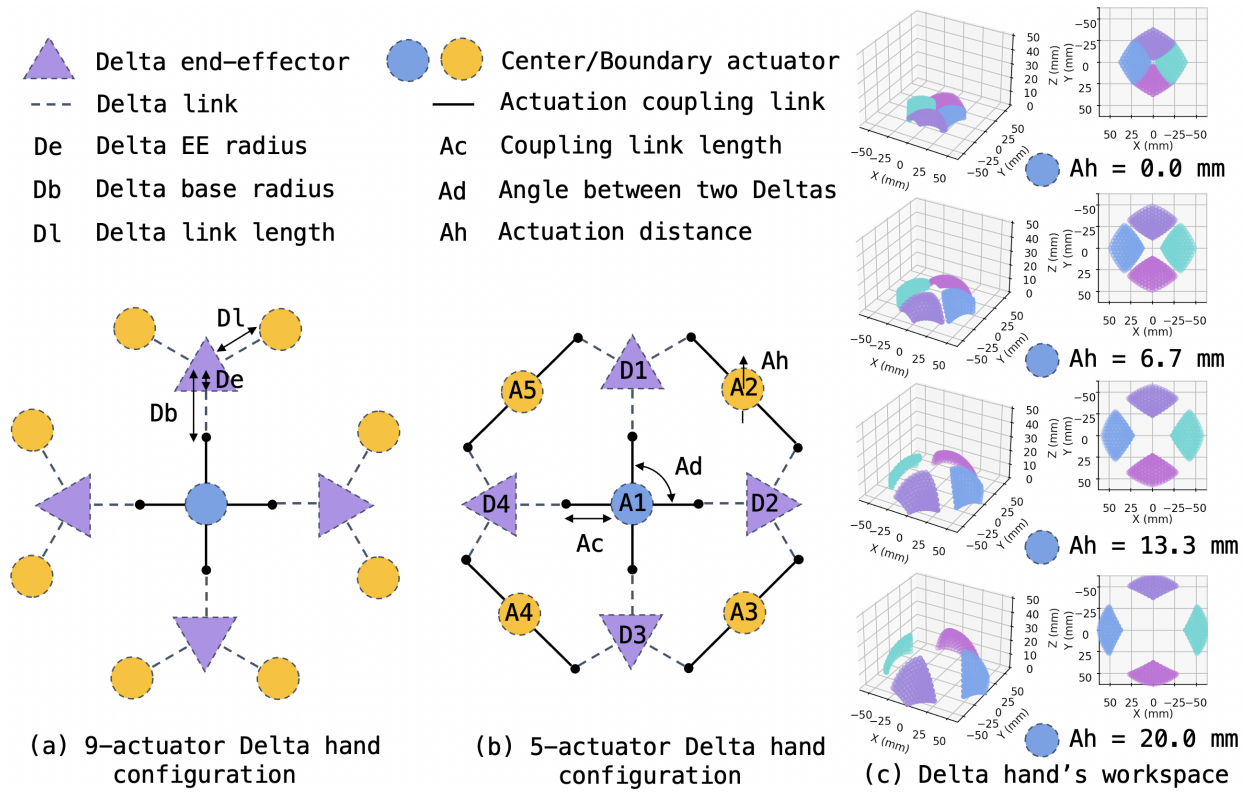


Figure 6.3: (a) and (b): Illustration of actuation coupling for 9-actuator and 5-actuator hands. (c) The hand's workspace varies by adjusting the center actuator's actuation distance.

be configured to different levels of *hardware synergies* by coupling the links of the Delta robots and sharing the actuators. This reduces energy consumption and control difficulty since fewer actuators are used. Here we show two potential coupling for a four-finger hand in Fig. 6.3 (a) and (b). Instead of using $4 \times 3 = 12$ actuators, we can reduce the number of actuators to 9 or even 5 by coupling links across fingers while still preserving certain key DoF. Since all actuators are co-linearly arranged, the coupling can be easily achieved by modifying the frames and actuator locations.

The study [165] reflects that the first two principal components of postural synergies mainly control open and close motions, therefore for the 9-actuator hand, we couple the inner links from four Delta robots to one center actuator which controls the open and close motion. All other eight boundary actuators can make fingers move independently in the XY plane for finer motions. For the 5-actuator hand, besides the center coupling, each Delta robot's boundary actuators are coupled with its neighbor's. Each finger can still perform lateral motion but their neighbors will move synchronously. We show how the center actuator's motion affects the workspace of the hand in Fig. 6.3 (c). When the center actuator is completely retracted, the fingers have a small overlapping workspace; the more the actuator extends, the more the fingers are separated.

6.3.3 Hand parametrization

DELTAHANDS can be parameterized physically and topologically. For physical parameters, at the hand level, we define the number of fingers N , coupling link length A_c , the angle between two fingertips A_d , and actuation distance A_h . At the finger level, we define the Delta robot's end-effector radius D_e , base radius D_b , and link length D_l as shown in Fig 6.3. For topological parameters, given the set of Delta robots as $\{D_1, D_2, \dots, D_n\}$ and the set of actuators $\{A_1, A_2, \dots, A_m\}$, we can associate them and represent actuation coupling. Such as for 5-actuator hand in Fig. 6.3, Delta robot $D_1 : \{A_1, A_5, A_2\}$, actuator $A_1 : \{D_3, D_4, D_1, D_2\}$.

6.3.4 Hand kinematics

Single Delta robot's kinematics have closed-form unique solutions that we can adapt to Delta hands. For each Delta robot D_i , we define its actuation space as $A_{D_i} = \{a_{i1}, a_{i2}, a_{i3}\} \in \mathbb{R}^+$, and end-effector space as $E_{D_i} = \{x_{i1}, y_{i2}, z_{i3}\} \in \mathbb{R}^+$, then we have forward and inverse kinematic models as $E_{D_i} = FK(A_{D_i})$, and $A_{D_i} = IK(E_{D_i})$. For a Delta hand without actuation coupling, we can simply use a single Delta robot's kinematics model for each finger. For hands with actuation coupling, all the *hardware synergies* are *rigid synergies* [26]. Thus we can control and model all DoF with kinematics models. Forward kinematics are the same by setting each Delta robot's actuation with its corresponding actuators' values. However, for the inverse kinematics, since the end-effector space is constrained by the reduced actuation dimensions, there might not be a precise solution for the desired end-effector position. Thus, we still first solve individual Delta robot's IK given the desired end-effector position, then project the solution to the feasible actuation space $\tilde{A} = PA$. Such as for the 9-actuator hand, the projection matrix P can be defined as:

$$P = \begin{bmatrix} \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & 0_{1 \times 8} \\ 0_{8 \times 1} & 0_{8 \times 1} & 0_{8 \times 1} & 0_{8 \times 1} & I_{8 \times 8} \end{bmatrix} \in \mathbb{R}^{9 \times 12} \quad (6.1)$$

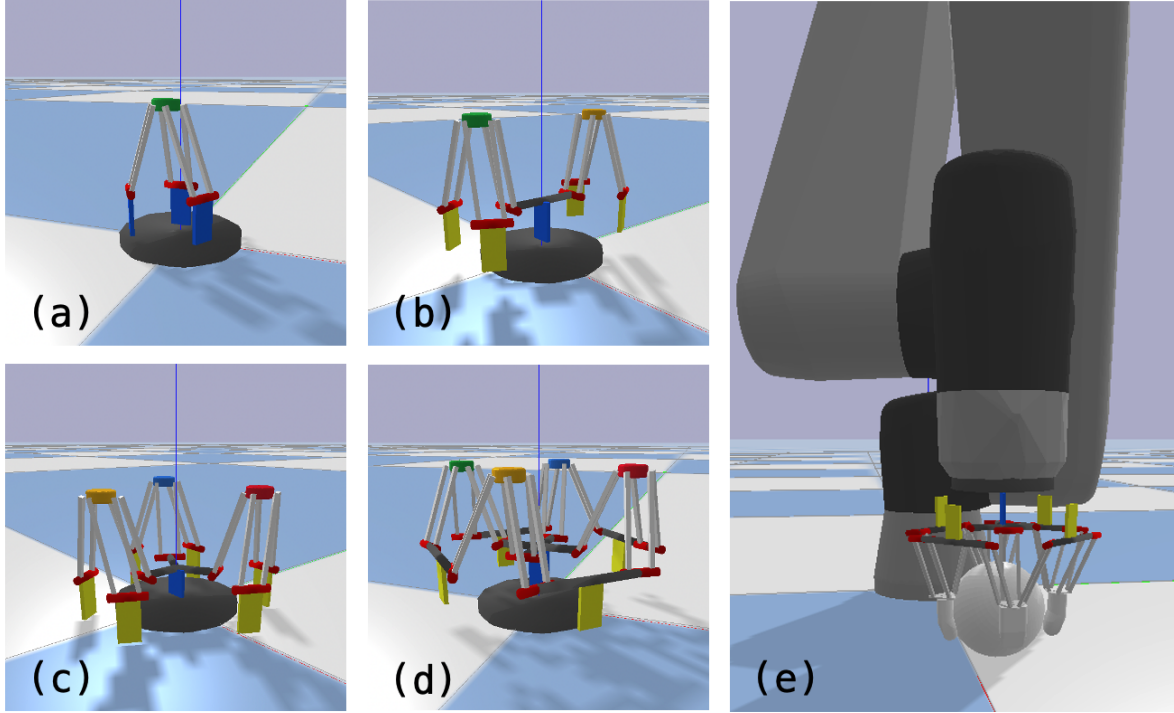


Figure 6.4: Simulation of DELTAHANDS including (a) single Delta, (b) twin Deltas, (c) triple Deltas and (d) quadruple Deltas as robotic hands. (e) Integration of a four-finger hand on a robot arm.

We approximate the hand’s IK solution by averaging the center actuation distance. The projection matrix reduces the actuation space dimension to reduce the control complexity which is similar to Eigengrasps [38]. Although the solution might not be precise, we demonstrate in Section 6.5 that we can still successfully complete dexterous grasping and manipulation tasks with the help of compliance in the finger links.

6.3.5 Hand design in simulation

Although we just presented two hand designs in the aforementioned section, we can build various hands based on different synergies. A practicable simulation enables us to explore unlimited design parameters, and iterate and validate the design in a fast and safe manner. We use PyBullet [45] and provide a high-level urdf generator given the hand parameters defined in Section 6.3.3. As shown in Fig. 6.4, we can build hands with anywhere from one to six fingers and simulate different hand synergies. We also integrate DELTAHANDS with a robot arm for grasping evaluations and can use it for potential design optimization and policy learning in simulation.

6.4 Delta Robot Characterization

Serving as the functional ”fingers” of the robotic hand, it is crucial to characterize and optimize the Delta robots’ capabilities. Our objective is to maximize the force profile and workspace of

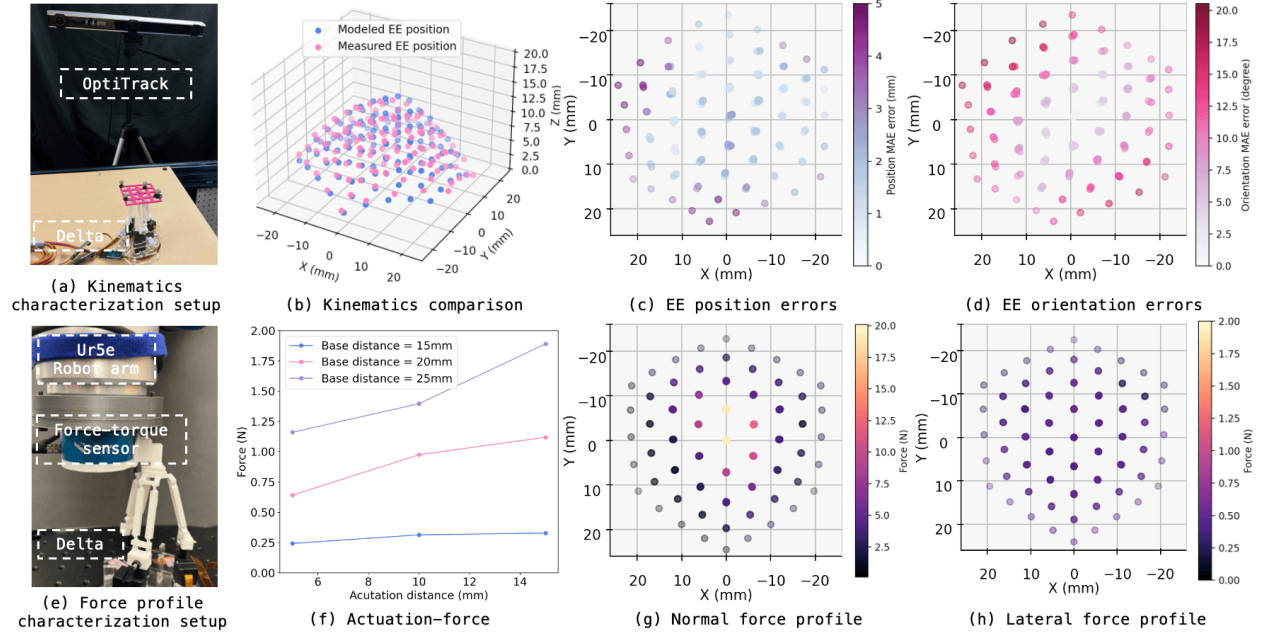


Figure 6.5: (a) Kinematics characterization setup with an OptiTrack system to track a Delta robot’s end-effector (EE) pose. (b) Comparing the modeled EE’s positions from forward kinematics and recorded positions from the OptiTrack system over a Delta robot’s workspace. (c) The EE’s position errors and (d) orientation errors. (e) Force profile characterization setup with a UR5e robot arm to execute Cartesian motions, a force-torque sensor to record contact forces, and a Delta robot fixed on the table to actuate towards the contact direction. (f) Correlation between actuation distances and contact forces. (g) (h) Characterization of a Delta robot’s force profile in normal and lateral directions.

the fingers based on functionality, and simultaneously, we strive to minimize the footprint and weight of the hand for efficient mounting on a robot arm, taking feasibility into account. Previous work [120] has characterized the soft materials of Delta robots but they didn’t explore optimization of the structure. In the following sections, we characterize the kinematics, workspace, and force profile of Delta robots and show how the design parameters affect the behaviors of Delta robots. This will assist with hand configurations for different applications.

6.4.1 Kinematics

We compare the kinematics accuracy of a Delta robot from the modeled and the observed end-effector’s poses. For both modeling and recording, we discretize the actuation space \mathbb{R}^3 for the three linear actuators into $5 \times 5 \times 5$ grids with 4 mm granularity given our 20 mm stroke actuators. We solve the forward kinematics and get modeled end-effector positions. Since Delta robots have only three translational DoF, the modeled orientations are all aligned with the principal axes. For observing poses, we use an OptiTrack [140] system to track the end-effector poses when actuating motors to the pre-defined distances (Fig. 6.5 (a)). Then we compare each pair of predicted vs recorded poses (Fig. 6.5 (b)). The average translational mean absolute errors (MAE) over the workspace are 0.73 mm, 0.77 mm, and 0.43 mm, and the average orientation MAE errors are 3.36° ,

2.28°, and 3.97° along the X, Y, and Z axes respectively. As shown in Fig. 6.5 (c) and (d), both translational and orientational accuracy decrease towards the boundary of the workspace where the soft Delta robot undergoes larger internal forces leading to more deformation as compared to the center of the workspace.

6.4.2 Force profile

We characterize a Delta robot's force profile in the lateral and normal directions using a pre-defined actuation distance 5 mm, and the correlations between the actuation distances and forces under three different configurations. As shown in Fig. 6.5 (e), the force profile characterization setup includes a Delta robot fixed on a tabletop optical board and a 6-Axis force-torque sensor mounted on a UR5e robot arm. We customize the end-effector (EE) of the Delta robot to be a cube and a flat surface on the force-torque sensor to constrain the contact face. We define the same $5 \times 5 \times 5$ grids in actuation space as the kinematics characterization. For each grid, the Delta robot first actuates to the desired location. The UR5e robot arm approaches the Delta robot from a safe distance and stops when a light contact (0.2 N) is detected by the force-torque sensor. Then the Delta robot actuates towards the touch direction 5 mm more and holds for 5 seconds. The force readings are recorded from the force-torque sensor during the experiments. We pre-define five touch directions including four lateral directions (+X, -X, +Y, -Y) and one normal direction (-Z).

Lateral and normal force profile over workspace

We plot the force profiles given 5 mm actuation in Fig. 6.5 (g) and (h). The peak of normal forces is at the center of the workspace and reaches up to 20 N. The forces decrease towards the boundary of the workspace. The average normal force is 5.32 N. In the lateral directions, similarly, the Delta robot becomes less stable towards the boundary, and the average of lateral forces are 0.61 N, 0.59 N, 0.76 N, and 0.67 N along +X, -X, +Y, -Y axes. This is because the parallel links become less supportive when the Delta robot's EE moves toward the boundary of the workspace. To measure the maximum lateral forces of a Delta robot, we choose four points along the finger closing direction within the hand and actuate the Delta robot until it buckles. The maximum forces from the outer to inner locations are 1.49 N, 2.34 N, 2.21 N, and 1.92 N which indicate the grasping forces of DELTAHANDS.

Force profile given different actuation distance

The actuation distance also affects the contact force since Delta robots are soft and compliant. We test the lateral direction forces by actuating the Delta robot 5 mm, 10 mm, and 15 mm starting from the home position. We also test three different configurations by adjusting the base radius (D_b) of the Delta robot to 15 mm, 20 mm, and 25 mm. As shown in Fig. 6.5 (f), the averaged lateral direction forces are linearly correlated to the actuation distances and the coefficient factor varies with different base distances.

6.4.3 Durability

To evaluate the lifespan, we actuate and reset a Delta robot 10,000 times (with the same $5 \times 5 \times 5$ grids in the workspace for 80 times) and then re-test the kinematics and force profile. We report

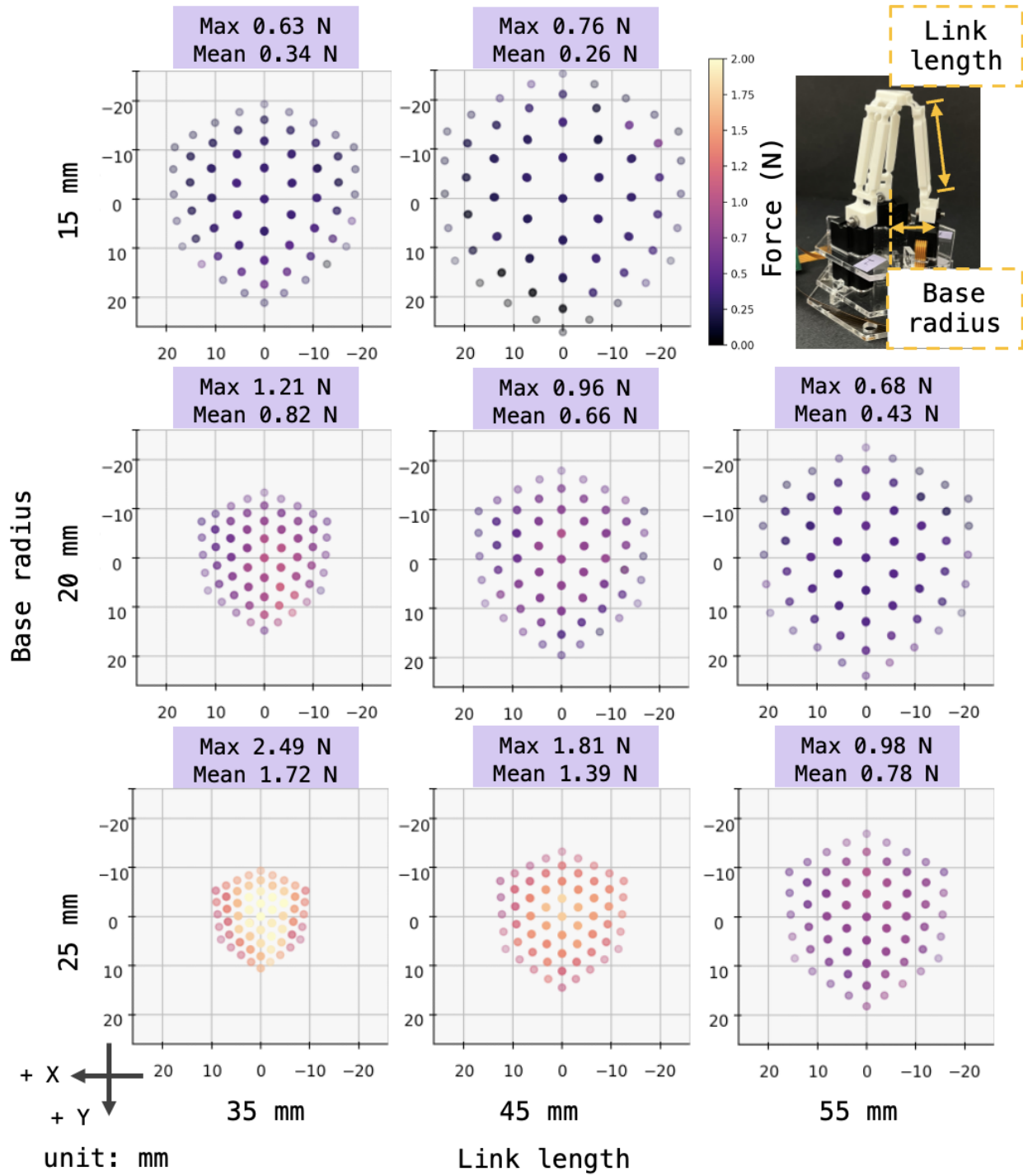


Figure 6.6: Characterization of force profile and workspace size by varying the Delta robot's base radius and link length. With a smaller base radius and longer link length, the workspace increases but the force profile decreases, and vice versa.

0.90 mm, 0.76 mm, 0.38 mm, as average translational MAE errors, 3.64° , 2.55° , 3.80° as average orientation MAE errors, and 0.73 N as the average lateral force profile. These are on the same level

as the previous characterization performance indicating no degradation of the Delta robot.

6.4.4 Workspace

To minimize the hand size while keeping a reasonable workspace and force profile for most manipulation tasks, design parameters including Delta robot’s link length (D_l), base radius (D_b), and EE radius (D_e) as indicated in Fig. 6.3 can be varied and configured. Here we characterize the workspace size and force profile by configuring the base radius as 15 mm, 20 mm, and 25 mm, and link length as 35 mm, 45 mm, and 55 mm. We keep the same EE radius as 6 mm to fit the fingertip. We follow the same procedure as force characterization. We plot the modeled workspace and recorded averaged lateral direction force profile data in Fig. 6.6. We observe the trade-off between the workspace size and stability of the Delta robot.

For household object manipulation in Section 6.5, we choose to use Delta robots with a 20 mm base radius (D_b) and a 45 mm link length (D_l) to accommodate the size and weight of the objects. We also pre-define the coupling link length (A_c) as 20 mm and the angle between fingers (A_d) as 90° to let the fingertips overlap when they are fully closed for firm small object grasping as shown in Fig. 6.3 (c). This gives us a maximum hand workspace of $124 \text{ mm} \times 124 \text{ mm} \times 25 \text{ mm}$.

6.5 Dexterous grasping and manipulation

6.5.1 Evaluation of grasping

We evaluate DELTAHANDS’ grasp capabilities with various objects qualitatively in the real world and quantitatively in simulation for two different hand configurations.

Grasp demonstration

Here we show grasping a set of 12 objects with 9-actuator and 5-actuator hands in the Fig. 6.7 (a)(c). We select objects (Fig. 6.7 (d)) with various sizes, weights, and geometries to qualitatively show the diversity of grasps. Along with static grasping, we can also reposition objects in hand (a cube and a rope) (Fig. 6.7 (b)).

Evaluation on YCB object dataset with grasp metrics

To compare 5-actuator and 9-actuator hands’ grasp capability, we evaluate them on the YCB object dataset [24] with grasp metrics [158] in simulation. First, we generate a pre-grasp shape distribution by only adjusting the open and close motions of the hand and the pose of the hand to get in contact with the object. Then, for different actuator coupling configurations, we fine-tune the grasp shape by sampling each actuator’s actuation individually based on the pre-grasp shape distribution. For each sampled grasp configuration, we evaluate whether the grasp is a force closure by considering the contact friction with a friction coefficient of 0.5. Then for force-closure grasps, we calculate the grasp quality scores by using the largest-minimum resisted wrench (Q_{LRW}) metric [158].

We choose 12 objects with various shapes and sizes for evaluation. We position the hand at 20 different heights and 4 different orientations (0° , 45° , 90° , 135° around the z-axis), and average













Object	Num force closure		Grasping score (Q_{LRW})	
	5-actuator	9-actuator	5-actuator	9-actuator
	720	586	130.216	132.105
	1110	872	95.642	95.312
	760	643	102.964	115.514
	1025	881	92.776	95.616
	291	218	131.036	128.247
	1098	926	126.796	122.690
	1034	942	63.981	64.350
	564	516	76.483	65.742
	765	661	76.442	74.753
	809	708	85.983	88.252
	804	674	75.161	52.418
	412	386	23.067	48.967

Table 6.1: Grasping evaluation of 5- and 9-actuator hands on 12 objects from the YCB dataset. We use the number of force closure grasps found out of 20,000 random searches and the largest-minimum resisted wrench (Q_{LRW}) grasping metric to evaluate the hand’s grasping capability. We observe that the 5-actuator hand has better sampling efficiency with lower actuation space dimension while having similar-quality grasping compared with the 9-actuator hand.



Figure 6.7: Grasping gallery with DELTAHANDS. We grasp (d) twelve daily objects with (a) a 9-actuator hand and (c) a 5-actuator hand. We also show (b) in-hand cube and rope repositioning.

the scores as the final evaluation scores. We show the number of force closures found from 20,000 random samples, and the grasping scores for both hands in Table. 6.1. We observe more force closures found for the 5-actuator hand on all objects indicating better sampling efficiency, and similar-level grasping scores for both hands which means that even with fewer DoF, it is still possible to grasp these objects.

6.5.2 Teleoperation for dexterous manipulation

In order to show DELTAHANDS’ dexterity, we teleoperate a 9-actuator hand and a Franka robot arm by using a Leap Motion [195] camera with its built-in human hand and finger tracking (Fig. 6.9). We map the operator’s left-hand palm translational motions to the Franka robot arm’s end-effector (EE) motions and two hands’ index and thumb fingers’ translational motions to the DELTAHANDS’ four-finger motions. We demonstrate three dexterous manipulation tasks: cloth folding, cap opening, and cable rearrangement (Fig. 6.8). Using real-time teleoperation with human corrections, we can achieve a closed human-in-the-loop control system.

To stabilize and simplify teleoperation, we leverage both *software synergies* in EE space and *hardware synergies* in actuation space. When mapping the operator fingers’ motions to DELTAHANDS’ EE motions, we pre-define task-specified synergy matrix $S \in \mathbb{R}^{12 \times 12}$ to coordinate finger motions similar to Eigengrasps [38]. Then, the DELTAHANDS’ EE poses are converted to a feasible actuation space by using the hand’s IK model with projection matrix P from Eq. 6.1. Thus, given the operator’s finger positions as $X = [p_{\text{left_thumb}}, p_{\text{left_index}}, p_{\text{right_thumb}}, p_{\text{right_index}}] \in \mathbb{R}^{12}$, where $p = [x, y, z]$, we map it to the DELTAHANDS’ actuation space as $A = P \times IK(SX) \in \mathbb{R}^9$.

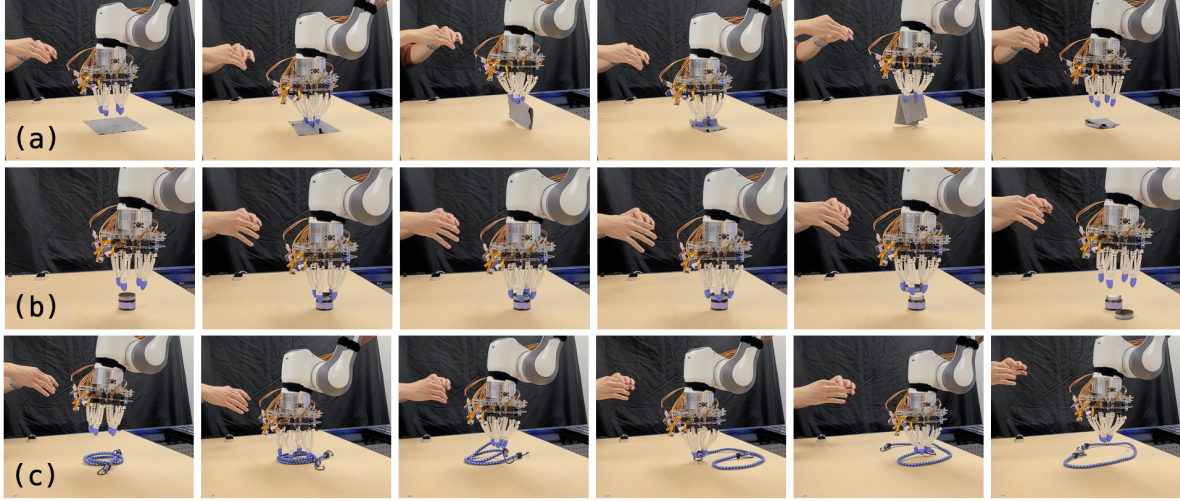


Figure 6.8: Teleoperation of (a) Cloth folding by folding the cloth twice in two perpendicular directions; (b) Cap opening by twisting the cap and then removing the cap from the bottle; (c) Cable arrangement by loosening the cable loop, and rearranging the cable.

Cloth folding

We fold a cloth in half twice along two perpendicular directions by pinching, lifting, placing, and spreading. The finger motions are aligned in X and Y directions therefore we define the synergy matrix $S_{folding}$ by averaging fingers' positions to a square shape constraint.

Cap opening

We open and remove a bottle cap by repeatedly grasping, twisting, and releasing. All fingers move symmetrically around the cap's central point, so we define the synergy matrix $S_{opening}$ by constraining fingers to a circular shape in a polar coordinate frame.

Cable arrangement

We straighten a loosely tangled cable by spreading, grasping, and sliding. This can also be realized by aligning the motions to X and Y directions, so we use the folding synergy matrix $S_{cable} = S_{folding}$.

These tasks demonstrate the hand's dexterous motions such as pinching, twisting, and spreading. Benefiting from hand synergies, we can easily and stably control the hand with human teleoperation. The soft materials of the hand allow it to safely and adaptively interact with the environment without damage and compensate for deviations in the kinematics.

6.6 Conclusions

We present DELTAHANDS, a synergistic dexterous hand framework that utilizes soft Delta robots. DELTAHANDS are low-cost, easy to build and control, and highly configurable regarding design parameters. We characterize the Delta robot's kinematics accuracy, force profile, and

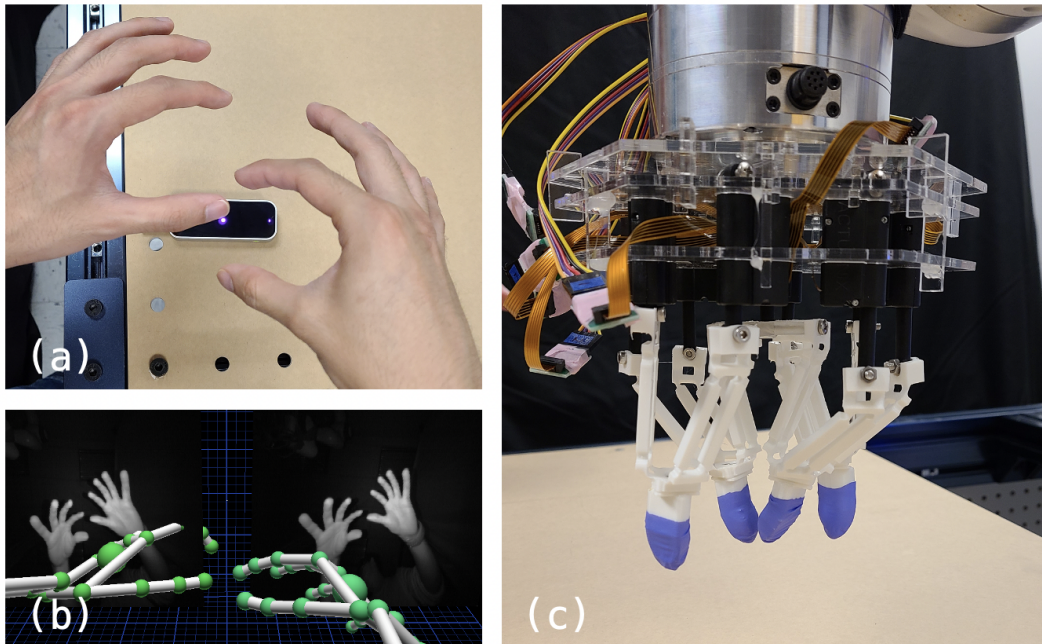


Figure 6.9: Teleoperation setup with a Leap Motion camera. An operator’s hand and finger poses are tracked by the camera and then mapped to the robot arm’s and a 9-actuator hand’s motions.

workspace range which can guide the design and configuration of our robotic hands. We present two different actuation synergies to reduce the control difficulty and energy consumption while keeping the hands’ dexterity. We also show the capability of using our hand framework to further explore various synergistic configurations. We demonstrate diverse object grasping with two hand configurations and quantitative evaluations in the simulator. Furthermore, we teleoperate a 9-actuator hand with a Franka robot arm to complete three dexterous manipulation tasks: cloth folding, cap opening, and cable arrangement.

To extend this work, we plan to explore task-oriented design optimizations as compared to manual tuning of hand parameters and actuation synergies. In addition, we aim to investigate adaptive control policies for different hand synergies. Finally, instead of closing the control loop through human teleoperation, we would like to add sensors on the hands and fingers to enable automated closed-loop control.

6.7 Acknowledgements

This work was funded by the NSF under project Grant No.CMMI-2024794. The authors sincerely thank Sarvesh Patil, Moonyoung (Mark) Lee, Sha Yi, Jennifer Yang, and Shashwat Singh for their help in discussions, experiments, and with manuscript revisions.

Tilde: Teleoperation for Dexterous In-Hand Manipulation Learning with a DeltaHand

This chapter is based on [Si, Zhang, Temel, and Kroemer, [178]].

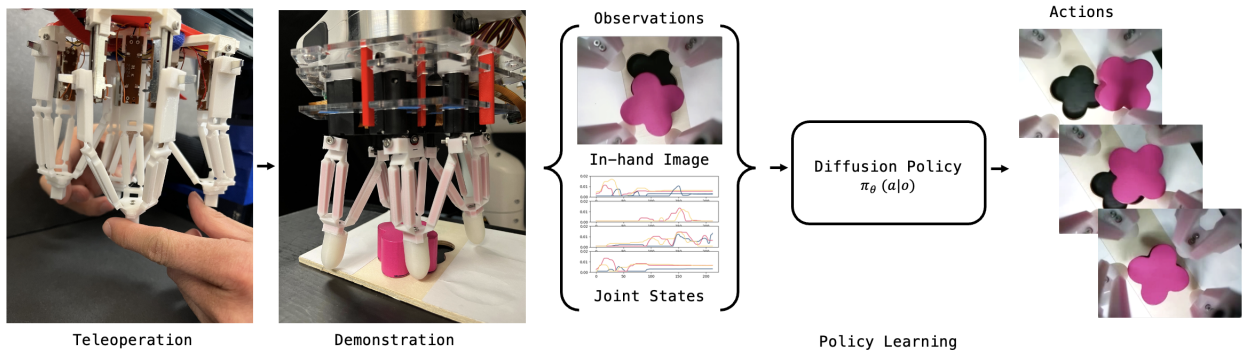


Figure 7.1: *Tilde*: *Teleoperation for Dexterous In-Hand Manipulation Learning with a DeltaHand*. We introduce an imitation learning-based in-hand manipulation system with a dexterous DeltaHand. We present a kinematic twin teleoperation interface, TeleHand, to collect demonstrations on seven dexterous manipulation tasks, such as shape insertion shown above. By using vision-conditioned diffusion policies, the DeltaHand can autonomously complete the tasks.

Abstract: Dexterous robotic manipulation remains a challenging domain due to its strict demands for precision and robustness on both hardware and software. While dexterous robotic hands have demonstrated remarkable capabilities in complex tasks, efficiently learning adaptive control policies for hands still presents a significant hurdle given the high dimensionalities of hands and tasks. To bridge this gap, we propose *Tilde*, an imitation learning-based in-hand manipulation system on a dexterous DeltaHand. It leverages 1) a low-cost, configurable, simple-to-control, soft dexterous robotic hand, DeltaHand, 2) a user-friendly, precise, real-time teleoperation interface, TeleHand, and 3) an efficient and generalizable imitation learning approach with diffusion policies.

Our proposed TeleHand has a kinematic twin design to the DeltaHand that enables precise one-to-one joint control of the DeltaHand during teleoperation. This facilitates efficient high-quality data collection of human demonstrations in the real world. To evaluate the effectiveness of our system, we demonstrate the fully autonomous closed-loop deployment of diffusion policies learned from demonstrations across seven dexterous manipulation tasks with an average 90% success rate.

7.1 Introduction

Dexterous manipulation is essential for a wide range of real-world tasks such as inserting small components precisely for manufacturing, administering medicine in hospitals, and handling delicate ingredients while cooking. However, a significant skill gap exists between human and robotic proficiency due to the demands for precision, robustness, and rapid adaptation to unstructured environments on both the hardware and software. Take the in-hand shape insertion task (Fig. 7.1) as an example. The robotic hand has to adjust its control policy based on real-time sensory feedback, such as visual observations of the object, and seamlessly switch between skills like translation, rotation, and finger gaiting to align and insert the block into the template. Completing such high-dimensional and long-horizon tasks requires precise and dexterous manipulators as well as adaptable and robust policies that can handle diverse scenarios. Thus, integrated systems are necessary to address the challenges of dexterous manipulation and advance the field.

Recent advances in imitation learning have shown great advantages in utilizing diffusion models [33, 157, 198] for efficient manipulation policy learning, as compared to deep reinforcement learning [3, 9] which is computationally expensive and data-hungry, or motion planning [29, 106, 135] which relies on accurate modeling. However, imitation learning methods require high-quality demonstrations, which are challenging to collect quickly and reliably for dexterous manipulations. To leverage imitation learning, we need highly precise and easy-to-use teleoperation interfaces for dexterous robotic hands that will allow us to collect diverse demonstrations.

Although anthropomorphic hands [42, 152, 159, 174] have already shown their ability to perform various manipulation tasks through teleoperation, these hands are designed to be general-purpose replacements for human hands which may be unnecessarily complex for certain domains. By contrast, non-anthropomorphic hands [111, 127, 177], with their lower control complexity and higher design flexibility, can be better tailored to tasks such as precise peg insertion or in-hand manipulation. However, these designs present additional challenges for imitation learning given the human-to-robot hand correspondence problem. DELTAHANDS [177], as shown in Fig. 7.1, are soft, compact, easy to customize, and possess high degrees-of-freedom (3-DoF per finger) that are simple to control, which makes them a great fit for dexterous in-hand manipulation. However, we need an intuitive and precise teleoperation interface to enable efficient imitation learning for DELTAHANDS.

In this work, we present *Tilde*, an imitation learning-based dexterous in-hand manipulation system built upon DELTAHANDS [177] (Fig. 7.1). We first introduce a customized DeltaHand with an integrated in-hand camera for visual feedback and a novel 3D-printed finger design with hybrid soft and rigid materials that is capable of exerting on average 40% more force than the original design with pure soft material. We then present a kinematic twin teleoperation interface named TeleHand, which has the same kinematics as the DeltaHand to enable a one-to-one mapping between joint states from the TeleHand to the DeltaHand. This direct mapping allows for real-time,

precise control of the DeltaHand. The TeleHand’s user-friendly design simplifies human operation, making it an efficient tool for collecting high-quality human demonstration data. Finally, by leveraging diffusion policies [33], we demonstrate the fully autonomous closed-loop deployment of our system on seven dexterous manipulation tasks including grasping, in-hand object rotations and translations using finger gaiting, precise shape insertion, and syringe pushing, all with an average success rate of 90%. We believe that our low-cost and user-friendly integrated system can serve as a useful research platform for learning data-efficient dexterous in-hand manipulation policies.

7.2 Related Work

7.2.1 Robotic Hands for Robot Learning

Robot learning with robotic hands has been broadly studied for robotic manipulation [97]. Anthropomorphic hands such as the commercialized Shadow Hand [42] and Allegro Hand [159] have been widely used for manipulation learning [13, 70, 154], but they are intended for more general purpose usage and being close-sourced makes it difficult to adapt them to specific requirements. Open-sourced research hands such as the Leap Hand [174] provide research opportunities on robot learning with low-cost hardware. Alternatively, non-anthropomorphic hands have advantages in design and control simplicity. The Yale OpenHand project [111] includes a series of hand designs that allow for fast prototyping and easy modifications, but they were optimized for simpler control with lower dexterity. ROBEL [3] was introduced as a platform for reinforcement learning benchmarks. DELTAHANDS [177] strike a good balance between cost, high dexterity, flexible design, and accurate yet simple kinematics which we adapt and customize in our integrated system.

7.2.2 Teleoperation Systems for Dexterous Manipulation

Teleoperation for robots enables efficient data collection of human demonstrations for robot learning. Kinesthetic teaching [4, 136] has been widely used to directly manipulate robots that have backdrivable motors and record the robot’s joint positions. However, the human may occlude portions of the view when they move the robot, which is unsuitable for training control policies using visual feedback. In addition, kinesthetic teaching on soft robots may introduce motions from undesired and unrepeatable bending or buckling due to their compliant structure, instead of reflecting changes in the actuator’s joint positions. Visual tracking has predominantly been used to collect low-cost demonstrations directly from human hand motions [13, 70, 154]. However, unstable, noisy visual tracking may require additional data smoothing processing or lead to more difficult policy training. This approach is also largely limited to anthropomorphic hands. Touch screen [192], and virtual reality (VR) setups [123, 218] were proposed in recent years, but they suffer from human-to-robot hand correspondence issues. Custom teleoperation interfaces such as tactile gloves [8] for anthropomorphic hands and twin robot hardware [201, 221] for robot arms with parallel-jaw grippers were presented as precise, intuitive, yet low-cost hardware. Similarly, we develop TeleHand, a kinematic twin teleoperation interface for a dexterous DeltaHand that is easy and intuitive to use.

7.2.3 Learning for Dexterous Manipulation

Reinforcement-learning in both simulation [9, 27, 101, 153] and the real-world [3, 89] have shown promising results in dexterous manipulation tasks such as solving in-hand Rubik’s cubes and in-hand reorientation of novel objects. However, the huge exploration space results in heavy computation and data-generation costs which may be unsuitable for scaling up dexterous manipulation. Imitation learning, alternatively, has shown better sample efficiency while still possessing high performance [59, 63, 84], but it requires high-quality expert demonstrations. We utilize imitation learning to train policies from real-world human demonstrations. Specifically, we employ Diffusion Policy [33], which leverages diffusion models to handle the high dimensional task spaces and multi-modal action sequences of dexterous manipulation tasks. While most imitation learning suffers from domain shifts in data distribution, DAgger [163] was proposed to overcome this issue by using additional on-policy interactions and expert corrections. We incorporate this approach into our system to improve performance.

7.3 Methodology

We present a dexterous manipulation learning system from three aspects: a dexterous robotic hand adapted from DELTAHANDS, a kinematic twin teleoperation interface, and imitation learning with diffusion policies. We demonstrate key features of our system including 1) the high dexterity and precision of the DeltaHand, 2) the low latency, ease-of-use, and precision of the teleoperation interface, TeleHand, and 3) the efficiency and generalizability of the policy learning for dexterous manipulation.

7.3.1 Dexterous Hand Design

Finger Design DELTAHANDS [177] is a configurable, highly dexterous, low-cost robotic hand framework based on Delta robots. The original DeltaHand’s fingers were configured with 3D-printed soft TPU links (Fig. 7.2(d)) for compliant and safe interactions. However, to benefit from the DeltaHand’s compliance and extend its capabilities to manipulation tasks that require more forces such as pushing the plunger of a syringe, we modify the design of the Delta finger’s links and joints as shown in Fig. 7.2(c). To improve the force profile of the Delta finger, we 3D-print hybrid TPU and PLA links where we embed rigid PLA material (red) inside a thin outer shell of TPU (white). This strengthens the whole finger structure and enables fingers to apply more force while still preserving enough compliance to safely handle collisions. To improve kinematic precision, we use edged joints with smaller joint lengths instead of the original curved joints to reduce undesired buckling and deformations during finger motions. We conduct similar kinematics and force profile characterizations as [177] and observe that with the new design, the Delta finger’s averaged kinematics error is reduced from 0.65 mm to 0.53 mm in position and 3.33 degrees to 2.50 degrees in orientation. In addition, the averaged lateral force profile is increased from 3.16 N to 4.51 N with 10 mm actuation. These modifications can be easily incorporated by just swapping out the Delta fingers from the linear actuators, showcasing the flexibility of the DELTAHANDS framework.

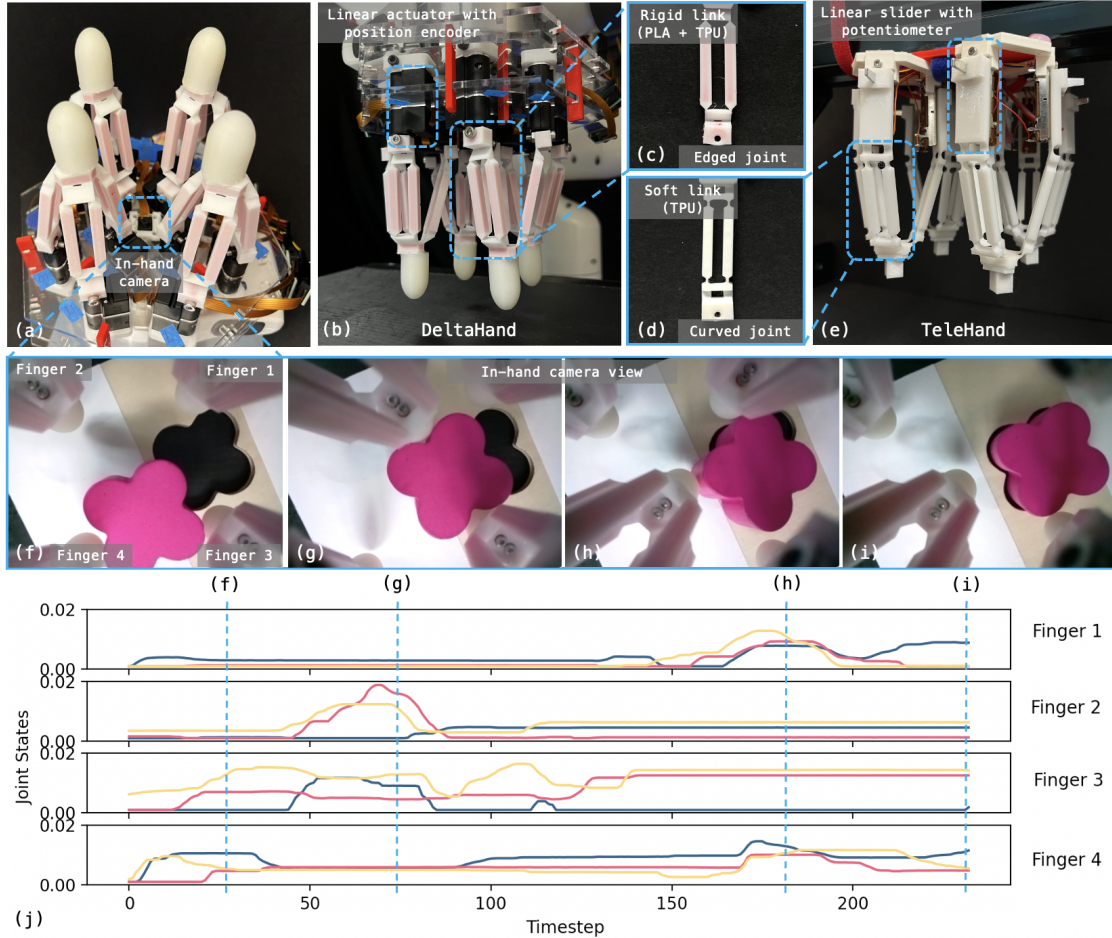


Figure 7.2: (a) A DeltaHand with an in-hand RGB camera. A kinematic twin teleoperation interface including (b) a DeltaHand and (e) a TeleHand. The TeleHand uses linear sliders with potentiometers to record the joint states of each finger. The DeltaHand will reproduce the motions of a TeleHand by using the Telehand’s potentiometer readings as desired joint positions for its linear actuators. (c) The DeltaHand’s fingers have 3D-printed rigid-core embedded links and edged joints, which increase the stiffness of each finger and enable them to exert more force. (d) The TeleHand’s fingers have 3D-printed soft links and curved joints, which induce more compliance in each finger. Therefore less force is required for users to teleoperate the robot, which makes teleoperation easier. (f)-(i) In-hand camera images that capture the object and the DeltaHand’s fingers. (j) The TeleHand’s joint states indicate the movement of each finger during a demonstration.

In-hand Camera Sensors are key components for closed-loop control by providing real-time feedback. In particular, local visual sensing is crucial for dexterous manipulation by capturing detailed geometric features [184]. Therefore, we integrate a mini Arducam camera module¹ into the hand for in-hand visual sensing as shown in Fig. 7.2(a). The camera can stream 640×480 resolution RGB images at 30 fps over Wi-Fi using a Raspberry Pi 4. The camera is located at the center of the hand and on the same level as the Delta finger bases without taking extra space. It has symmetric observations to provide useful inductive bias [78]. We manually tune and fix the camera’s focal length to focus on the area around the fingertips. The DeltaHand’s kinematics permit a mostly unobstructed view of the fingertips which benefits visual servoing.

Fingertip Design To increase the contact friction and enable soft contact for more secure grasps, we first 3D print the “bone” of the fingertip with TPU material, and then cast an additional layer of silicon rubber using Ecoflex 00-20 FAST².

Hand Configurations An overview picture of the DeltaHand can be seen in Fig. 7.2(a). We arrange four 3-DoF Delta fingers in a circular layout with a 40 mm radius from the hand center to each Delta finger center. Each finger has a 40 mm link length and 20 mm base radius, and is individually actuated by three linear motors with 20 mm stroke length. This gives a total of 12 DoF and a 110 mm \times 110 mm \times 30 mm workspace for the DeltaHand.

7.3.2 Teleoperation Interface

Previous work on DELTAHANDS [177] utilizes a Leap Motion camera for teleoperation. When we conducted preliminary experiments with it, we found the visual tracking to be noisy and unstable, especially when fingertips are close together due to potential occlusions. In addition, since the Leap Motion was placed on the table surface and teleoperators held their hands in the air, the operators’ hands would unavoidably drift over time because of fatigue which would inject additional noise into demonstrations. Given these reasons, we develop a kinematic twin teleoperation system for the DeltaHand to get precise and high-quality demonstrations. The system includes a TeleHand (Fig. 7.2(e)) manipulated by a human teleoperator, and a DeltaHand (Fig. 7.2(b)) to reproduce the TeleHand’s finger motions in real-time.

The TeleHand has the same configurations including the hand size, finger arrangement, and finger size, as the DeltaHand to enable direct one-to-one joint position mapping. The DeltaHand’s fingers are actuated by linear motors with 20 mm stroke length and each finger’s link bases move prismatically. Similarly, the TeleHand consists of linear sliders with a 20 mm motion range to create the same linear mobility for each finger as the DeltaHand except they move passively. Teleoperators can easily drag and move the finger end-effectors of the TeleHand which will lead to joint position changes in the sliders. The joint positions of TeleHand’s fingers will be recorded by each sliders’ potentiometers and then directly mapped to the DeltaHand as the linear motors’ desired positions. For the TeleHand, we use the original Delta finger design (Fig. 7.2(d)) which is more compliant and easier for humans to manipulate.

¹<https://www.arducam.com/product/arducam-raspberry-pi-5mp-spy-camera-b0066/>

²<https://www.smooth-on.com/products/ecoflex-00-20-fast/>

To enable real-time interfacing, we use Robot Operating System (ROS) for communication. Both the TeleHand and the DeltaHand use Arduino microcontrollers to directly publish and receive ROS topics via a control PC. This also allows our teleoperation system to be easily integrated with other robotic arms. Our teleoperation system including the DeltaHand and the TeleHand can be manufactured in a day with off-the-shelf materials, 3D printing, and laser cutting, and costs around \$1000. From Fig. 7.2(j), we show that the joint states read from the TeleHand are continuous and smooth which improves the training stability and efficiency.

7.3.3 Learning with Diffusion Policies

Diffusion models have shown their advantages while being used for policy learning from demonstrations [33, 157, 198] compared to behavior cloning [59, 150]. They can greatly improve performance by capturing multimodal action distributions, and high-dimensional action spaces, which are key challenges for dexterous manipulation tasks. Therefore, we adapt the CNN-based Diffusion Policy [33] to our system for dexterous in-hand manipulation policy learning with a DeltaHand. We condition the diffusion policies on visual observations from the in-hand camera and joint states of the DeltaHand (Fig. 7.2(f)-(j)) and predict action sequences. Both the joint states and actions are represented as the 12-dimensional absolute actuator joint positions. We trained policies using either actuator joint positions or end-effector positions derived from forward kinematics as inputs, and we experimentally found that using joint positions resulted in better performance.

Imitation learning methods for long-horizon tasks are known to suffer from covariate shift [163]. To avoid this issue, we use DAgger [163] to add on-policy interactions if the policy fails during the inference, and refine the policy with these DAgger demonstrations. In addition, data augmentation has been broadly used to improve the generalization and robustness of learning, especially for dexterous manipulation tasks with high dimensional state and action spaces. Through experimentation, we found that using various data augmentation techniques on observations greatly improved task performance. We leveraged 1) random image cropping and rotation to improve the rotational and translational invariance of fingers' visual servoing to the objects, and 2) Gaussian noise to joint state observations to guide the policy in learning funneling behaviors that can make the policy more robust when encountering unseen joint states. Specifically, we randomly cropped the images from their original size of (240, 320) to (216, 288) and rotated the images within 30 degrees. We added Gaussian noise with a standard deviation of 3.16 mm to each joint state. However, we found that random resized cropping does not improve performance because the camera is fixed w.r.t. the hand. In addition, some tasks are directional and the fingers' movements are reflected in the in-hand view, so directional invariance created by image flipping is therefore detrimental.

7.4 Experimental Setup

We evaluate our proposed system on seven dexterous manipulation tasks as shown in Fig. 7.4: **Grasp**, **Block Slide**, **Block Lift**, **Ball Roll**, **Cap Twist**, **Syringe Push**, and **Shape Insert**. **Grasp** is a fundamental skill for most manipulation tasks. The second to fifth tasks focus on different in-hand object repositioning skills: **Block Slide** corresponds to horizontal XY translations, **Block Lift** corresponds to vertical Z translations, **Ball Roll** corresponds to rotations around the X and Y

axes, and **Cap Twist** corresponds to rotations around the Z axis. The above tasks mostly require repeated motions, while the last two tasks consist of multi-modal action sequences and longer time-horizons. **Syringe Push** requires fingers to precisely align the syringe before pushing the plunger, while **Shape Insert** requires the fingers to translate, rotate, and transport the object to the final goal pose. Through our experiments, we show the capability of our proposed system to handle these dexterous manipulation tasks.

7.4.1 Data Collection

We mount a DeltaHand on a Franka robot arm as shown in Fig. 7.3. For most tasks, we keep the robot arm static while the DeltaHand uses its in-hand capabilities to manipulate the objects. An external RGB camera³ is placed in front of the experiment workspace. For each task, we manually preset the height and the location of the arm to approximately align the DeltaHand’s workspace with the object. To collect demonstrations, we first define the goal for each task which can be verified from the visual observations. If we reach the goal, we end the demonstration, or we run until we reach 5000 time steps which roughly equates to 250 seconds (data collection runs at 20fps speed). To collect DAgger demonstrations during inference, we first deploy the learned policy and let the DeltaHand manipulate an object autonomously while a human monitors. When the human decides that the policy has failed or is unlikely to finish the task, such as when the fingers stop or oscillate between similar states repeatedly, the human will pause the policy deployment. When the policy is paused, the human teleoperator first manually moves the fingers of TeleHand to match the current DeltaHand’s finger positions to reproduce the failure configuration and then takes over the control of the DeltaHand with the Telehand to finish the task with teleoperation. We use these teleoperated DAgger demonstrations from the failure point to fine-tune the policy.

7.4.2 Training Details

We train CNN-based Diffusion Policies [33] for all tasks. For visual features, we use ResNet18 as the visual encoder for both in-hand and external observations and then concatenate the 512-dimensional visual features with the 12-dimensional joint state vector. Actions are represented as the next timestep’s joint state. We fix the observation horizon, action prediction horizon, and action execution horizon to 2, 16, and 8 for all tasks, respectively. We train each policy model for 100 epochs with the AdamW optimizer using learning rate= $1e-4$ and weight decay= $1e-6$.

To improve the training stability, we normalize all the joint states and actions to the range $[-1, 1]$, and images to $[0, 1]$. In-hand images, external images, and joint states from demonstrations are synchronized to 20 fps. However, we found that down-sampling the data by a factor of 3 reduces the effect of the idle actions from demonstrations.

7.4.3 Tasks

An overview of all tasks is shown in Fig. 7.4. We aim to use these tasks to evaluate the policies’ capabilities, efficiency, and generalizability. For tasks (a) - (d), we evaluate the policies with additional unseen test objects as displayed on the right side of the object sets. For tasks (f) and (g), we randomly initialize the objects within the experiment workspace.

³<https://www.amazon.com/gp/product/B0C289GYVZ/>

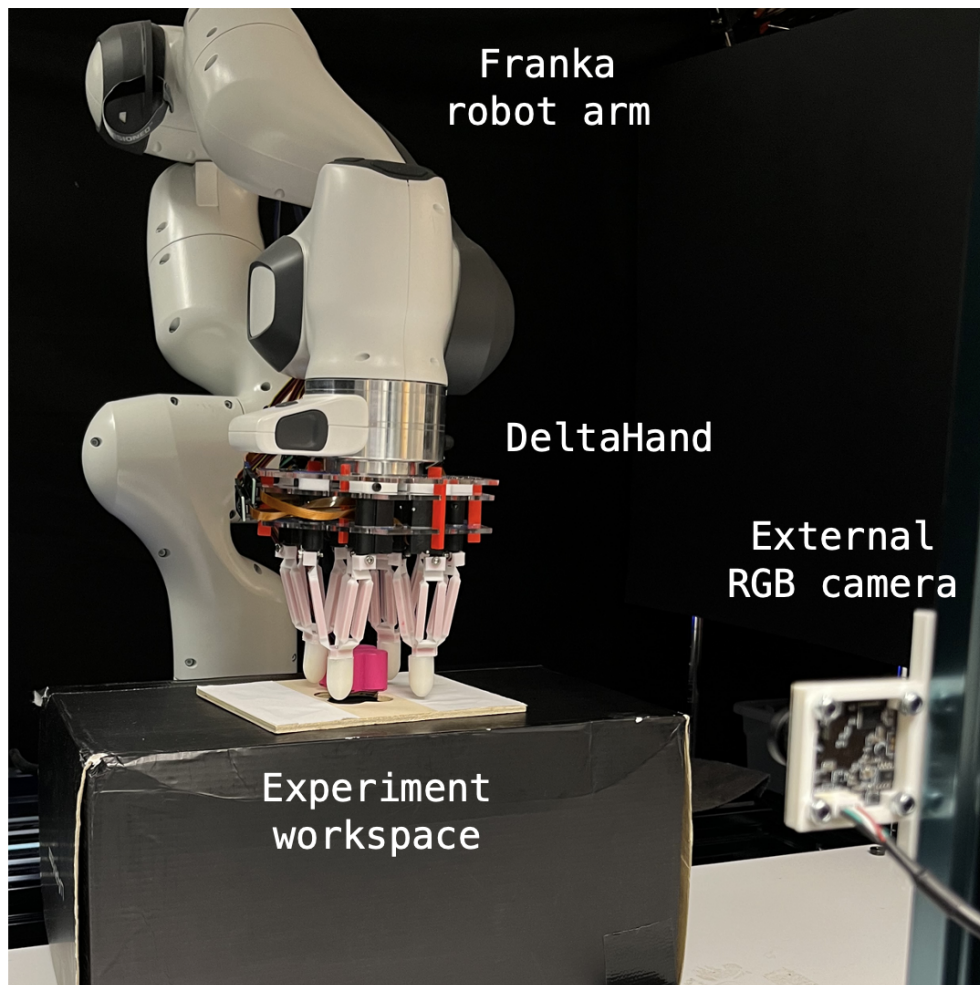


Figure 7.3: Experimental setup. We mount a DeltaHand on a Franka robot arm. We pre-set the height and location of the Franka arm on top of the experiment workspace. An external RGB camera is mounted in front of the experiment workspace.

7.4. Experimental Setup

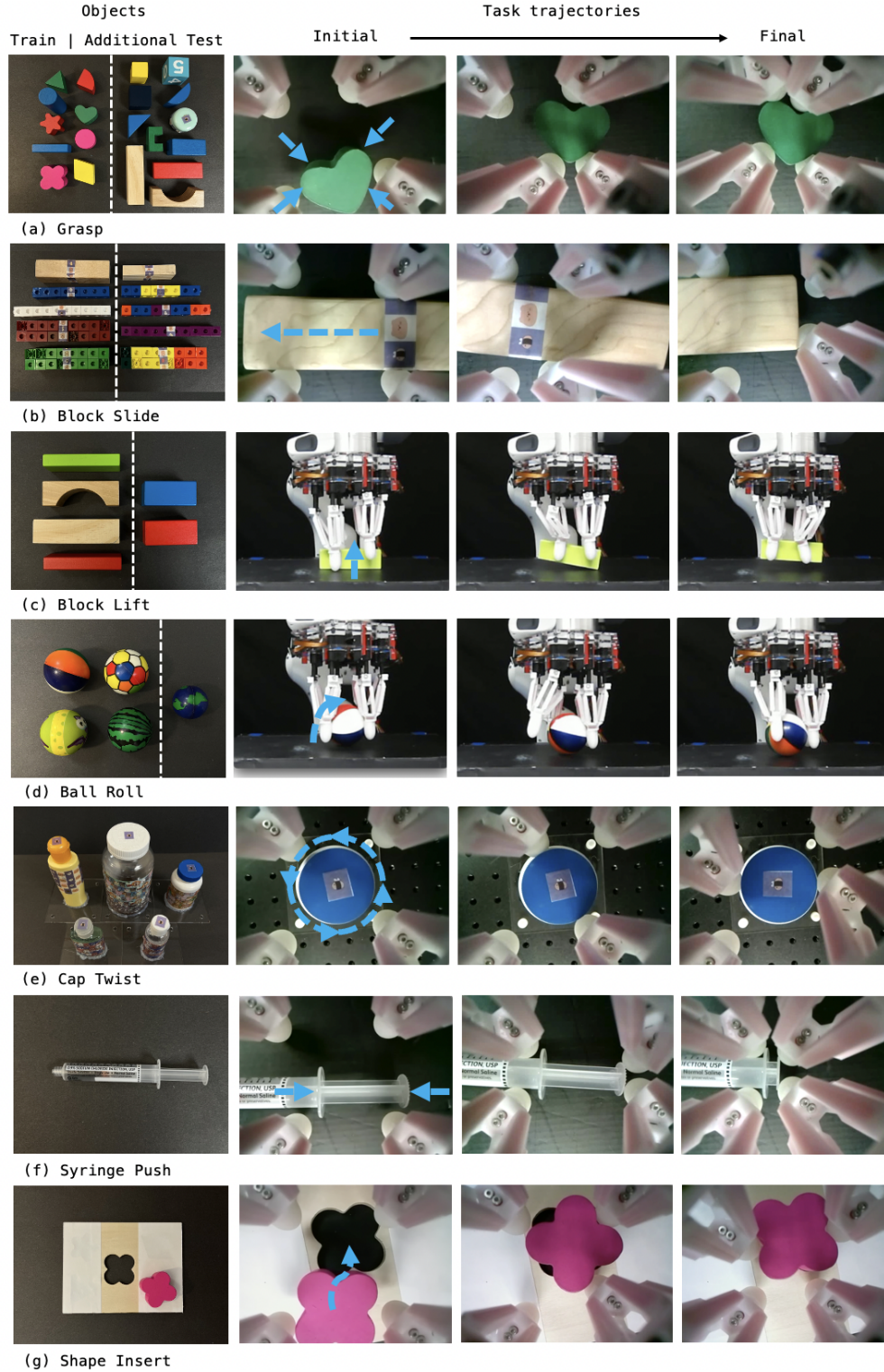


Figure 7.4: Task gallery. We evaluate our system on seven dexterous manipulation tasks: (a) **Grasp** (b) **Block Slide** (c) **Block Lift** (d) **Ball Roll** (e) **Cap Twist** (f) **Syringe Push** (g) **Shape Insert**. The goals of tasks are indicated by blue arrows in the initial images of task trajectories. For tasks (a)- (d), we separate the training and additional unseen testing objects with white dashed lines.

Task	Grasp	Block Slide	Block Lift	Cap Twist	Ball Roll	Syringe Push
# demos	45	40	20	30	20	50
# DAgger demos	10	0	10	10	5	20
Before DAgger	17/20	10/10	7/10	8/10	7/10	6/10
After DAgger	20/20	10/10	8/10	10/10	10/10	8/10

Table 7.1: Experimental results on six tasks. We show that with less than 50 demos, we can achieve success rates over 60% on all tasks before DAgger. With additional DAgger demonstrations, all tasks have improved results and achieved success rates over 80%.

Grasp We grasp a set of objects with various shapes, colors, and weights. We first teleoperate the DeltaHand to center the object and then grasp the object using all fingers. Afterwards, we lift the robot arm 10 cm to check whether the grasp is stable.

Block Slide We slide rectangular blocks of different sizes and colors horizontally from right to left. We tape the center of the blocks to provide distinctive features and allow for consistent human resets. During the demonstrations, we teleoperate the right set of fingers to move the object while we position the left set of fingers to form a funnel to stabilize the block. We end the demonstration once the right end of the block is located in the center of the in-hand view.

Block Lift We lift blocks of differing sizes, shapes, and colors up from the table by alternating grasps on the block with the right and left set of fingers.

Ball Roll We roll the ball between the fingers. We use two opposing fingers to loosely hold the ball while the other two fingers roll the ball around the axis defined by the virtual connecting line of the two holding fingers.

Cap Twist We rotate the cap of bottles 360 degrees. The bottom parts of the bottles are fixed to the table and the cap is twisted by two fingers. A full rotation is indicated by the bee sticker on the cap.

Syringe Push We push the plunger to close the syringe. We place the left set of fingers on the syringe barrel while we position the right set of fingers at the end of the plunger. The right and left sets of fingers have to align with the syringe and then push together. Any misalignment during the push will block and jam the pushing which requires repositioning fingers. During initialization, the syringe is randomly placed between the fingers on the table and the plunger is opened to a random length. We removed the rubber tip of the plunger which usually seals in liquid to reduce the friction and forces for this set of experiments. We further evaluated the forceful syringe push with the rubber tip on the plunger in Section. 7.5.7 as a challenging task.

Shape Insert We move and match a flower-shaped block to its corresponding template hole on a board. Both the board and the object are initialized randomly and the DeltaHand needs to translate, rotate, and transport the object to match the template hole.

7.5 Experimental Evaluations

We experimentally evaluate the performance of our proposed system on the aforementioned tasks, and we aim to answer the following questions:

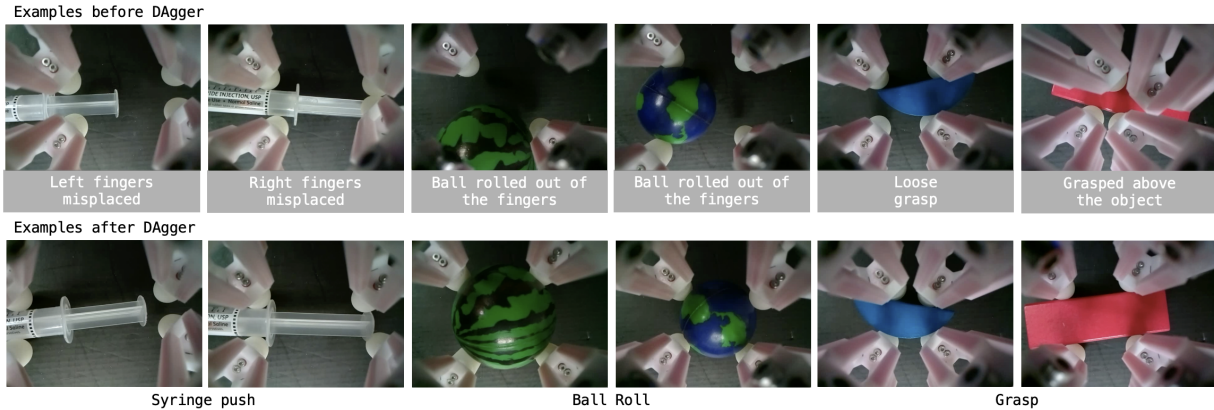


Figure 7.5: Qualitative comparisons between task executions from policies trained before and after DAgger demonstrations. By refining the policies with corrective demonstrations from failure cases, the policies can handle these challenging scenarios.

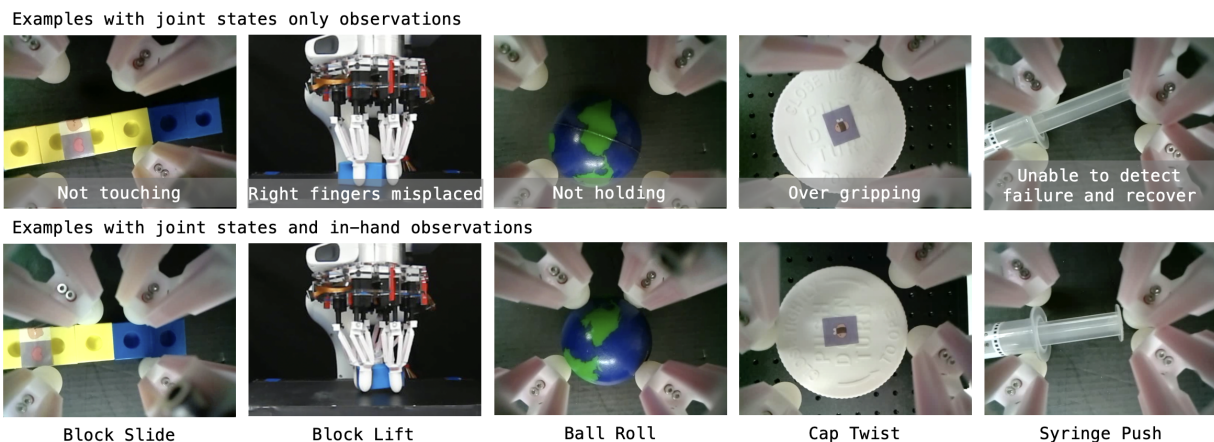


Figure 7.6: Qualitative comparisons between task policies conditioned on observations with joint states only versus joint states with in-hand images. We observe that visual observations improve the generalizability of the policies by adapting the contact points of the fingers to the objects' shapes, sizes, and locations.

1. How practical is our system at performing dexterous manipulation tasks?
2. How effective are the in-hand observations on the task performance?
3. How efficiently can our system learn robust policies with a limited number of demonstrations?

7.5.1 Experimental results

The number of demonstration data for the first six tasks can be seen in the first and second rows of Table 7.1. They vary depending on the task difficulty and the number of objects we use. After the initial policy training, we run 10 test trials on both the train and additional test objects. If the policy fails during the evaluation, we collect new demonstrations starting from

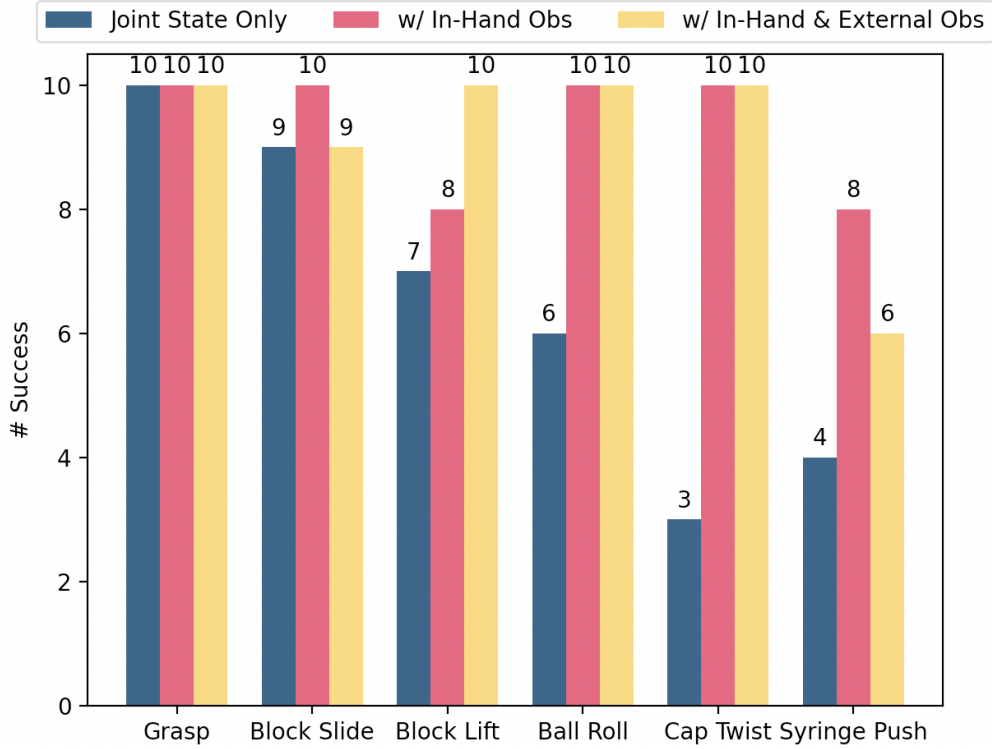


Figure 7.7: Effect of using visual observations. We evaluate the policies trained with observations as 1) joint states only, 2) joint states with in-hand images, and 3) joint states with in-hand and external images. The results show performance improvement with visual observations compared to only using joint states as observations.

the failure configuration and teleoperate the DeltaHand until success is achieved. These DAgger demonstrations are then used to refine the policy. The success rates before and after DAgger are shown in the third and fourth rows of Table 7.1. We use joint states and in-hand images as observations for all tasks. We observe that for most tasks, with less than 50 demonstrations, we can achieve a success rate of over 60%, and even a 100% success rate on the **Block Slide** task. For other tasks, DAgger demonstrations can improve the success rates to over 80% and recover from previously encountered failure cases.

7.5.2 Failure Analysis

Fig. 7.5 shows some common failure cases during the initial tests and how DAgger improves the performance on these. In the initial **Syringe Push** evaluations, the left two fingers occasionally incorrectly grasped the plunger when the syringe was placed more to the left. This showed that the policy did not know to position the left two fingers onto the barrel of the syringe. With additional DAgger demonstrations, the policy was able to move the left two fingers further to the left when necessary to hold onto the barrel of the syringe while the right two fingers pushed the plunger. Another common failure case was that the right two fingers occasionally grasped onto the plunger instead of positioning themselves behind the end of the plunger because the syringe was initialized too far to the right. With DAgger, the policy was able to either make the two left fingers pull the syringe more to the left before the right two fingers started pushing on the plunger, or make

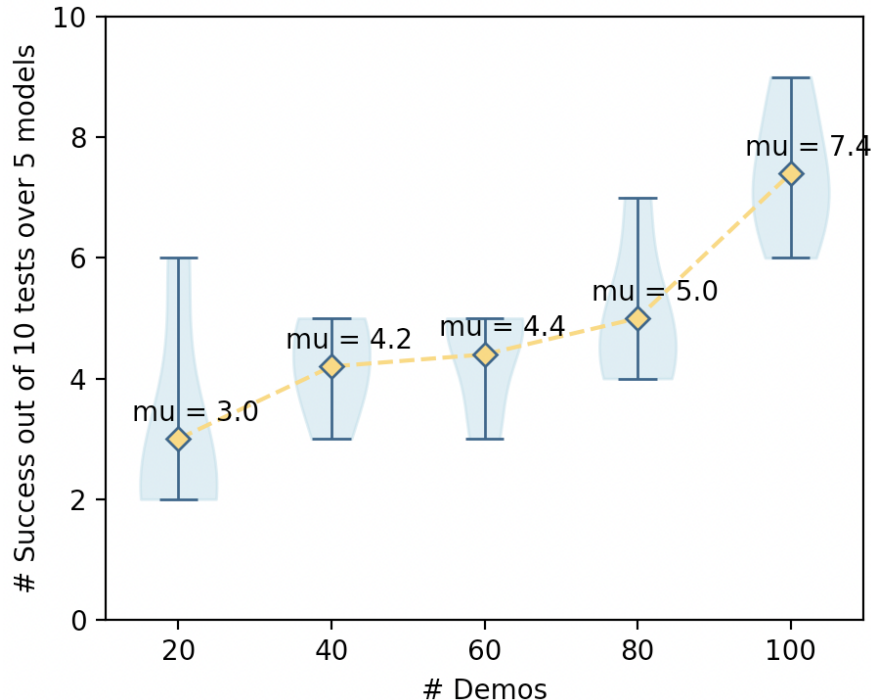


Figure 7.8: Data efficiency evaluation. We evaluate policies trained with 20, 40, 60, 80, and 100 demonstrations of the **Shape Insert** task and average the performances over five models using different random seeds. The success rates increased with more training data.

the right two fingers first push the plunger as much as they could and then reposition themselves behind the end of the plunger and push the rest of the way in. These additional demos from the failure cases enabled the diffusion policies to learn more robust recovery behaviors.

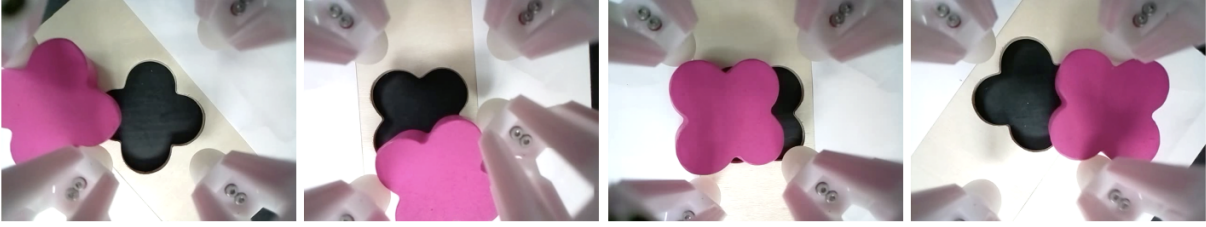
In the **Ball Roll** task, the balls occasionally rolled out from the grasp of the two opposing fingers when pushed. By using additional DAgger examples, the DeltaHand was able to adapt to different sizes of balls and also stop rotating the ball when the policy determined the ball was not positioned correctly and reposition the ball before restarting to rotate it.

Finally, in the **Grasp** examples, the DeltaHand was unable to grasp some unseen objects that were of differing heights and shapes such as flat blocks versus tall blocks and semicircle blocks. With additional DAgger examples on these objects, the policy was able to successfully adapt to the new objects. We conclude that with more diverse data, the domain shifts from the training data distributions were reduced and the policies became more robust to different scenarios.

7.5.3 Effect of using visual observations

We have two types of observations: DeltaHand’s joint states from motor encoders as proprioception, and in-hand images or external images as exteroception. We evaluate how sensitive the tasks are to different kinds of observations. As shown in Fig. 7.7, we train the policies with observations as 1) joint states only, 2) joint states and in-hand images, and 3) joint states, in-hand images, and external images. We show that most tasks achieve better performance with in-hand images compared to using joint states only. However, with additional external images, the performance does not differ much from solely using in-hand images with joint states. We noticed that in

Failure examples with policies trained on 20 demonstrations



Failure examples with policies trained on 60 demonstrations



Failure examples with policies trained on 100 demonstrations

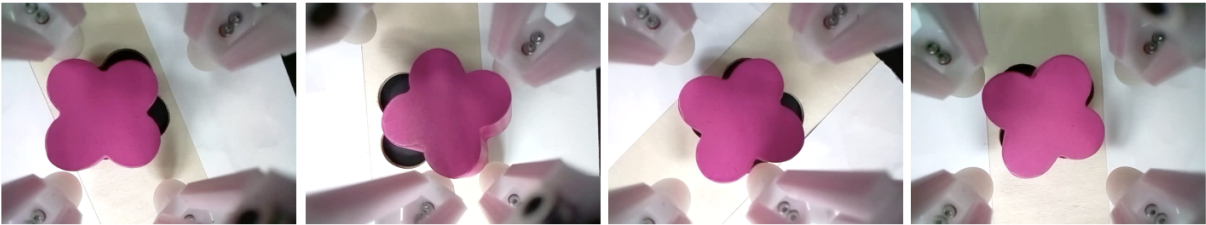


Figure 7.9: Examples for comparison between policies trained with 20, 60, and 100 demonstrations. **Shape Insert** requires the hand to transport and reposition objects to align with the template hole. With less training data, the task mostly failed at the early transportation stage. When the training data size increased, the policies often failed during the final fine orientation alignment stage where most cases are visually ambiguous.

most failure cases, the objects are hard to see from the external camera due to their small size and occlusions from the fingers, thus the external images are not as informative as the in-hand images. But for the **Block Lift** task, the policy with external images gets better results because the external camera can capture the object’s state better such as the height of lifting, which is more informative. For the **Grasp** task, all three policies achieved a 100% success rate and we give significant credit to the compliance of the DeltaHand which can passively adapt to most objects even without in-hand visual observations.

Some examples for comparison can be seen in Fig. 7.6. By using joint states only, common failure cases were that fingers would not contact the object and were unable to move or secure the object, they over-gripped the object which restricts movement, or were unable to detect failure and recover. By leveraging in-hand visual observations, the fingers can visual servo to adaptively make contact with objects based on their shapes, sizes, and locations.

Task	Observations	BC [118]	IBC [59]	Diffusion Policy [33]
Cap	State	0/10	0/10	3/10
Twist	State + In-Hand	4/10	0/10	10/10
Shape	State	0/10	3/10	2/10
Insert	State + In-Hand	2/10	3/10	7/10

Table 7.2: We evaluate the performance of three Imitation Learning methods: Behavior Cloning (BC) (Robotmimic [118]), Implicit Behavior Cloning (IBC) [59], and Diffusion Policy [33] on two tasks, **Cap Twist** and **Shape Insert**, either with joint states only or with joint states and in-hand visual observations. We show Diffusion Policy achieves the best performance.

7.5.4 Data Efficiency Evaluation

From Table. 7.1, we see that most tasks can achieve 100% success rates with less than 50 demonstrations. These tasks such as **Block Slide**, **Block Lift**, **Cap Twist**, and **Ball Roll**, use repeated motion patterns. For long-horizon tasks, multi-modal actions are required to complete the tasks. For instance, in **Shape Insert**, the fingers need to rotate, translate, or transfer objects between different sets of fingers as shown in Fig. 7.2 (f)-(i). To evaluate data efficiency on the **Shape Insert** task, we randomly sample five sets of 20, 40, 60, and 80 demonstrations from the dataset of 100 demonstrations and train policies with these data subsets respectively. The evaluation results in Fig. 7.8 show an increase in success rate when policies are trained on more data, indicating that the policy has better generalizability given more data.

We visualize common failure cases from policies trained with 20, 60, and 100 demonstrations in Fig. 7.9. We observe that with less training data, the task failed mainly at the early transportation stage where the fingers are unable to move the object closer to the template hole. With more training data, the failures occurred closer to the final orientation alignment stage. We conclude that diffusion policies can learn more complex and finer manipulation skills with more data.

7.5.5 Comparison of Different Imitation Learning Methods

To support our design choice of using diffusion policies, we comparably evaluate three Imitation Learning methods: Behavior Cloning (BC) (Robotmimic [118]), Implicit Behavior Cloning (IBC) [59], and Diffusion Policy [33] either with joint states only or with joint states and in-hand visual observations on two tasks: **Cap Twist** and **Shape Insert**. We report the performance of each method in Table. 7.2. We show that the Diffusion Policy achieved the best performance. We observe that 1) BC can predict smooth trajectories due to the continuity of the MLP network and has a fast inference speed (60 FPS) with its small model scale. However, it fails to reason about the multi-modal actions necessary to complete more complex tasks such as **Shape Insert**. For the **Shape Insert** task, it often fails at the final fine orientation alignment stage. On the other hand, actions predicted from 2) IBC contain a lot of noise due to the sampling process, which is unsuitable for dexterous manipulation tasks that require high precision. The successful trials on the **Shape Insert** task resulted from random interactions with the block that did not reflect clear intentional trajectory motions. Finally, 3) Diffusion Policy can handle multi-modal actions while also being able to predict relatively smooth trajectories at a real-time inference speed (20 FPS). We include more implementation details of these three methods in Section 7.6.4.



Figure 7.10: We evaluate the generalization of policies with additional unseen irregular-shaped objects used on the **Block Slide** task.

7.5.6 Evaluation on Robustness and Generalization of Policies

The robustness and generalizability of policies to unstructured environments is crucial for real-world deployment. Along with the evaluation on unseen objects, we further evaluate the robustness of the learned policies to randomly initialized hand poses on the **Block Slide** task. For each trial, we first generate a random relative transformation consisting of a z-axis translation, and rotations around the x and y axes based on the original pre-defined fixed hand pose. The sampling range of the z-axis translation is between $(-5, 5)$ mm and $(-10, 10)$ degrees in both the x-axis and y-axis orientations. We then move the DeltaHand to this randomly initialized pose and begin policy inference. The policy is still able to succeed in 17 of 20 trials. In addition, we include several unseen irregularly-shaped objects in these 20 tests, as shown in Fig. 7.10, and the learned policy can still succeed without any additional training. We observe that the compliance of the fingers can help with misalignments between the hand and the object and compensate for policy errors. Most failure cases occurred when the object shifted outside the hand’s workspace due to the random initialization, and the fingers were no longer able to reach the object. This issue can be potentially addressed by integrating robot arm motions to adjust the hand pose in future work.

7.5.7 Performance on Challenging Tasks

Diamond-shaped Block Insert The flower-shaped block is radially symmetric and can be aligned with less than a 45-degree rotation. We further tested our system with a diamond-shaped block that requires up to 180 degrees of orientation alignment. The demonstration trajectory length of the diamond-shaped block insertion is on average 23.65 seconds compared to 18.08 seconds for the flower-shaped block insertion. We show that with 80 demonstrations, the learned policy can still handle this long-horizon task with 5 successful runs out of 10 trials. A test example can be seen in Fig 7.11. We show that the policy learns to coordinate movements among fingers to re-orient

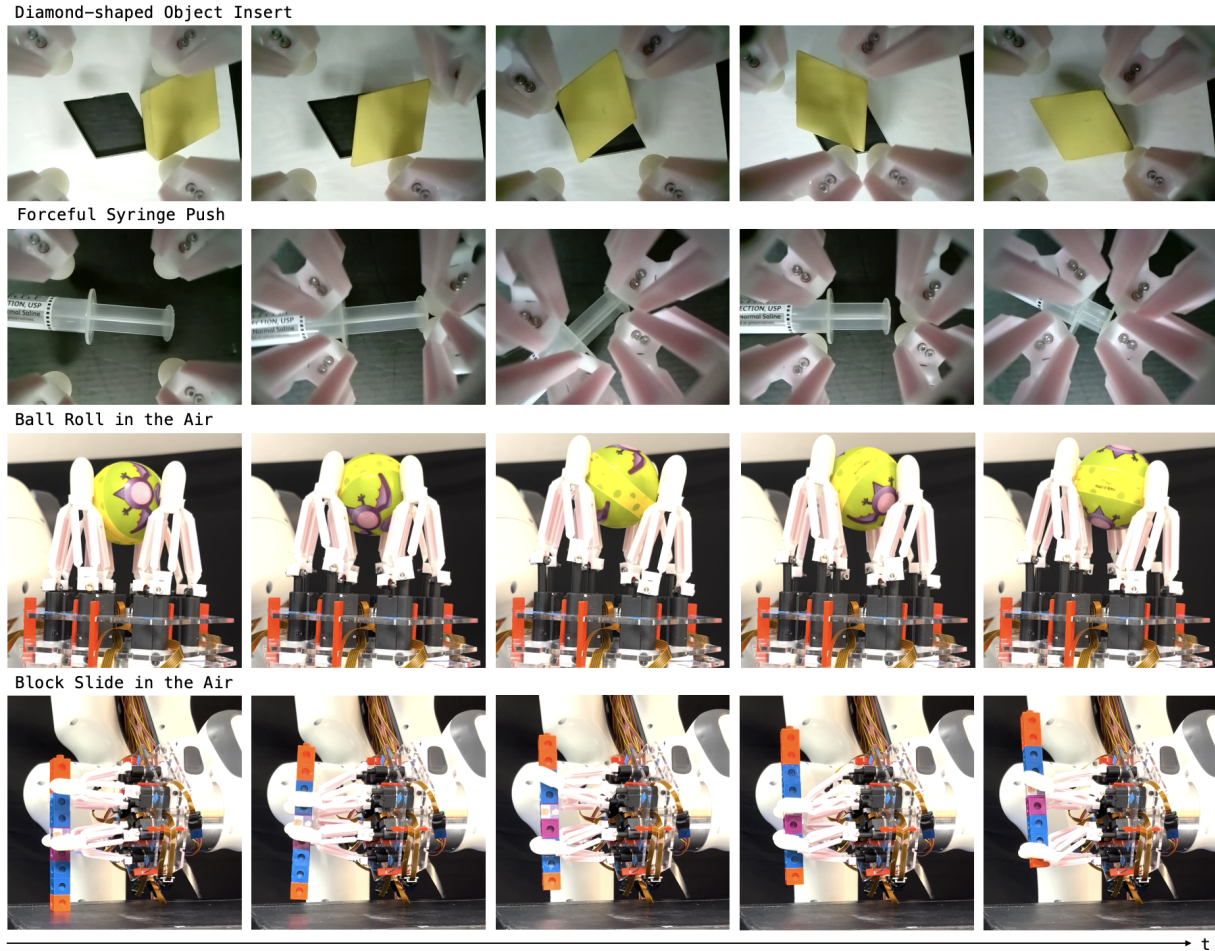


Figure 7.11: Test examples of four challenging tasks: diamond-shaped block insertion, forceful syringe push, ball roll in the air, and block slide in the air over time. Diamond-shaped block insertion requires orientation alignment of more than 90 degrees. Forceful syringe push requires higher alignment precision to succeed. Both ball roll and block slide in the air have non-recoverable failure risks.

the object over 90 degrees. Most failures are due to unseen object poses, which we believe can be resolved with more data.

Forceful Syringe Push We further evaluate a forceful syringe push task by adding the rubber seal to the tip of the plunger where the friction between the plunger and the syringe barrel demands around 4.0 N to push the syringe. With 40 demonstrations, we can get 8 successful runs out of 10 tests. A test example can be seen in Fig 7.11. We observe that the policy can detect the misalignment caused by the pushing force in the third plot of Fig 7.11 and realign the fingers with the syringe to complete the task.

Finger Gaitting in the Air In all the aforementioned tasks, objects are mainly supported on the table and can be recovered if the policies perform incorrect actions. We further evaluate two challenging finger-gaiting-in-the-air tasks including **Ball Roll** and **Block Slide** which have non-

recoverable failure risks. Examples can be seen in Fig. 7.11 where the fingers hold and rotate a ball upright for **Ball Roll** and pinch and slide a block upward for **Block Slide**. We quantitatively evaluated these two tasks and obtained 9 and 6 successful runs out of 10 tests respectively. The singular failure case for **Ball Roll** occurred when the ball fell down between fingers. This resulted in a large visual occlusion from the in-hand camera view which lead the policy to fail. Most failure cases for **Block Slide** resulted when one pair of fingers pinched too tightly and the other pair could not move the block any further. This demonstrates the importance of incorporating tactile sensors in the fingers which we plan to explore in future work.

7.5.8 Limitations and Discussions

From the experimental results, we demonstrate our system’s capability on a varied set of grasping and in-hand translation and rotation manipulation tasks. Through our extensive evaluations, we observe the limitations of our system as follows:

Lack of Tactile Sensing

We show that in-hand visual observations have greatly improved the performance of the learned policy from our ablation study. However, tactile sensing is crucial in providing direct force feedback for delicate object manipulation which have non-recoverable failure modes. Although we can still achieve many tasks by solely using visual feedback, the policy needs to learn more robust behaviors and relies on repeated attempts to achieve the target goal. Tactile feedback could lead to more intentional and controlled interactions with objects in highly-dexterous tasks.

Generalization to Unstructured Environments

We show that the learned policies can adapt to unseen objects in a top-down tabletop environment. However, we observe failure cases with environments out of the training distribution such as different backgrounds or large disturbances in the hand poses. To improve the robustness of in-hand tasks, we plan to explore object-centric learning approaches such as conditioning the diffusion policy on object segmentation masks to better handle unseen environments and allow for explicit reasoning about objects in clutter near or in the hand.

Lack of Robot Arm Motion

We emphasize the in-hand manipulation capabilities of our proposed system in this work, and our evaluations focus on finger-based manipulations. However, integrating robot arm motions can improve robustness and extend the capabilities of our system beyond in-hand manipulation. For example, when randomly disturbing the hand pose, we observe failure cases of objects being outside of the fingers’ workspace. Leveraging the robot arm’s movement could address such limitations by properly initializing and aligning the hand’s pose with the target object. This can also enable extrinsic dexterous manipulation.

7.6 Conclusions

We present *Tilde*, an imitation learning-based in-hand manipulation system with a dexterous DeltaHand. We introduce a kinematic twin teleoperation interface for low-cost data collection of high-quality human demonstrations and efficient end-to-end real-world policy learning by using diffusion policies. We show that with our system, we can perform a variety of dexterous manipulations and achieve an average success rate of 90% across our evaluation tasks. These tasks include grasping, in-hand object re-positioning and re-orientation, and finger gaiting. Our experiments show the capability of the system to learn robust vision-based dexterous manipulation policies from demonstrations that were acquired with our easy-to-use and precise teleoperation interface.

In the future, we would like to improve the generalizability of our system for broader dexterous manipulation tasks in unstructured environments. Therefore, we plan to augment sensing modalities with tactile sensing by incorporating fingertip tactile sensors for more delicate tasks, integrate arm motions into our teleoperation system to achieve intrinsic and extrinsic dexterity [65, 224], and explore object-centric approaches to improve the policies’ robustness to unseen scenarios.

Acknowledgments

The authors would like to thank Sha Yi, Shashwat Singh, Jennifer Yang, Moonyoung (Mark) Lee, Mohit Sharma, Saumya Saxena, and Haomin Shi for their help in discussions, experiments, and with manuscript revisions. This work was supported by the National Science Foundation under grant No. CMMI-2024794 and Google.

Appendix

7.6.1 DeltaHand Comparison

We compare between the adapted hand design for this work and the original design from DELTAHANDS [177] as shown in Fig. 7.12. In this work, we decoupled the center actuator and changed 1) the fingertip design, 2) the finger links and joints design, and 3) added an in-hand camera. The new fingertip has an omnidirectional design and an additional soft elastomer coating layer to increase the contact area and contact friction. This has greatly improved grasp stability during experiments. The finger links have embedded rigid PLA cores and the finger joints have smaller gaps to improve the kinematics accuracy and force profile. We observe that the hand can exert stronger forces to manipulate daily objects such as pushing to close a syringe plunger while the original design from DELTAHANDS [177] can only manipulate lighter objects such as clothes or cables. The decoupled actuators give independent 3 DoF control for each finger to increase dexterity. The inclusion of an in-hand camera provides clear in-hand visual observations for closed-loop policy learning.

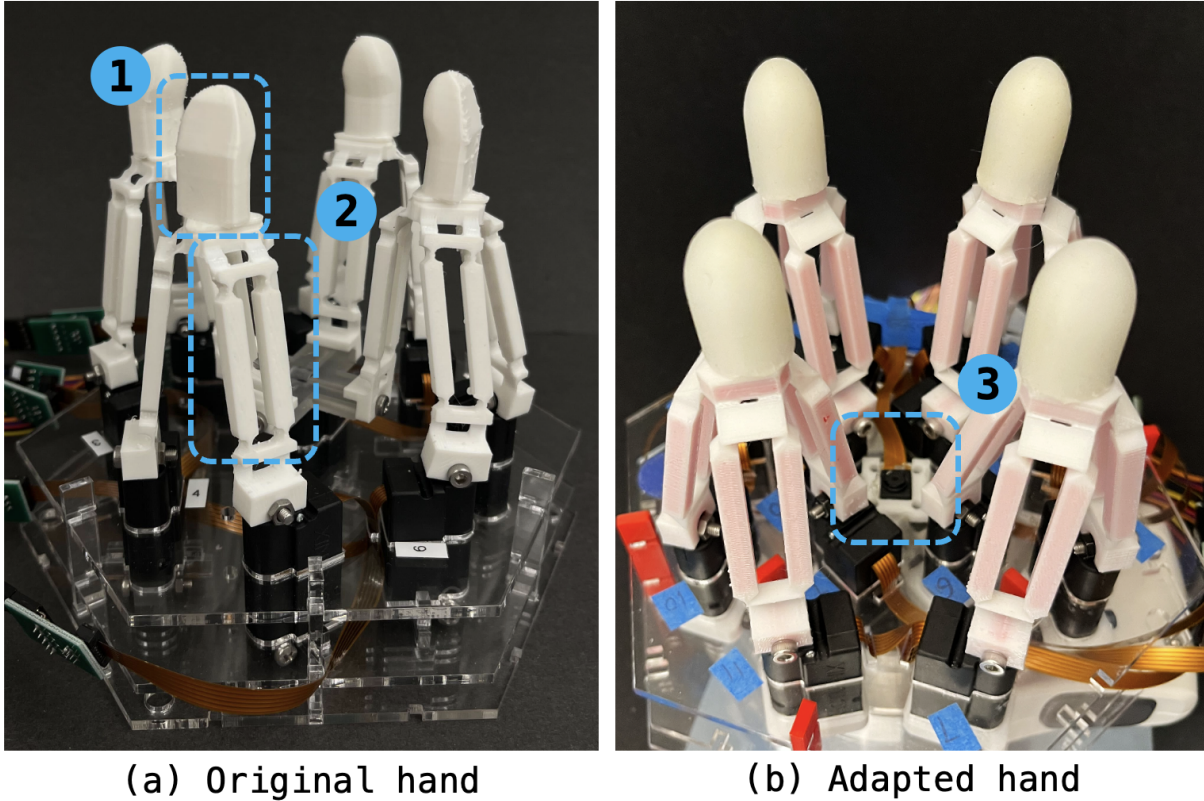


Figure 7.12: Comparisons between (a) the hand from DELTAHANDS [177] and (b) our adapted hand. We customize 1) the fingertips to an omnidirectional design for larger contact area with a soft coating elastomer layer for increased friction, 2) the finger links to a rigid core-embedded multi-material design and the finger joints to a better defined shape for reducing kinematics error and increasing force profile, 3) an in-hand camera to provide clear visual observations for closed-loop policy learning.

7.6.2 Teleoperation Interface Details

Communication and Control from TeleHand to DeltaHand The overview of the teleoperation system can be seen in Fig. 7.13. The whole system can be modularized into the Control PC, DeltaHand, TeleHand and external sensors. The Control PC sends and receives all control and sensor signals from the other modules as well as serves as the main computing source for policy inference. The DeltaHand module includes the hand and the in-hand RGB Camera. A microcontroller (Adafruit Feather M0) is used to send and receive the current and desired joint states of the actuators and run a PID control loop. The in-hand camera is connected to a Raspberry Pi which sends images to the Control PC. The TeleHand uses a microcontroller (Arduino Mega) to read and send the sliders' positions when it is teleoperated by a human. Other sensors such as the external RGB camera can be connected directly to the Control PC. All sensor and control signals are published as ROS topics and their frequencies are listed in Table. 7.3. As a data pre-processing step for policy learning, we synchronize all the streaming data to the joint states' frequency (20 FPS) by using their ROS timestamps.

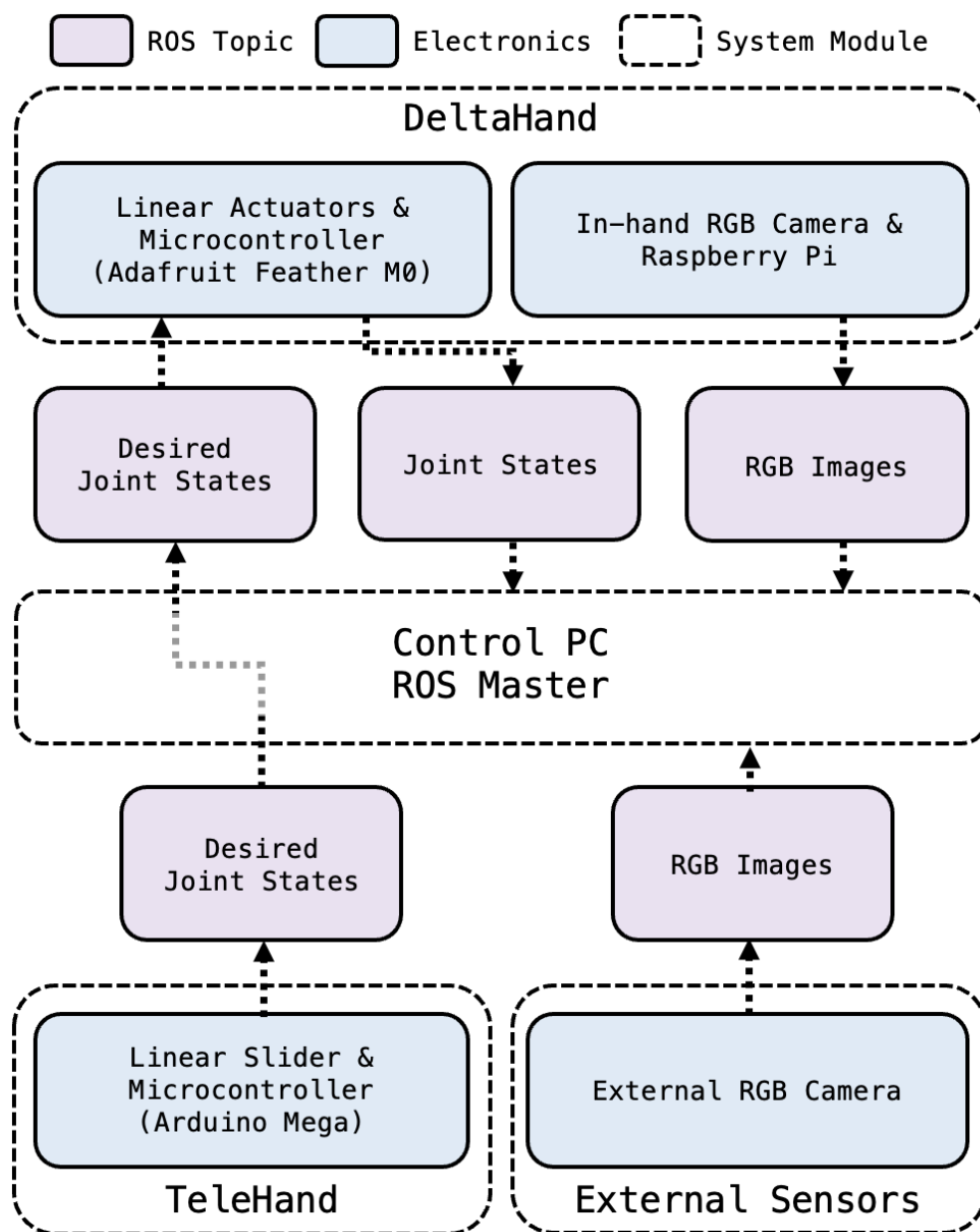


Figure 7.13: Overview of the teleoperation system. The system can be modularized to the control PC, DeltaHand, TeleHand, and external sensors. The Control PC receives and sends all sensor and control signals as ROS topics. It also serves as the main computing source for policy execution. The DeltaHand has its own microcontrollers for the actuators and sensors. The TeleHand is used to interface with a human to get control signals from the slider potentiometers. All other sensors belong to the external sensors module.

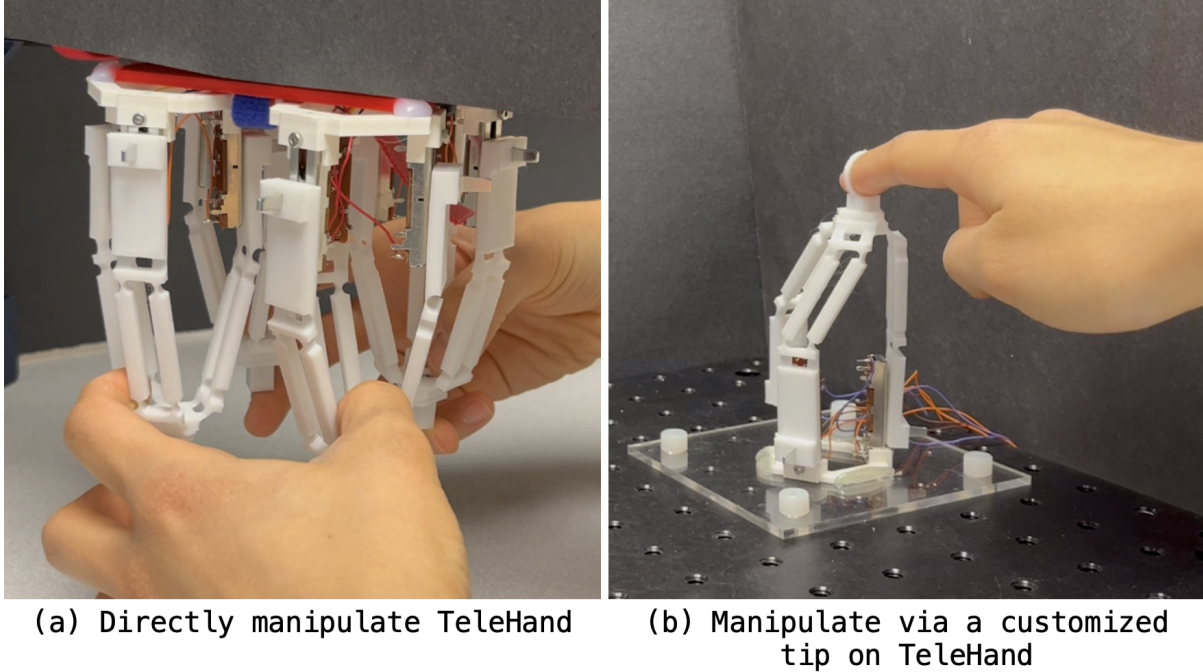


Figure 7.14: The TeleHand can be teleoperated differently depending on the human’s preference. The human can a) directly manipulate the TeleHand by grabbing the links or tips or b) attach and use a customized end-effector such as a ring to constrain their finger to the TeleHand’s end-effectors.

	Joint State	Delta Control	Linear Slider	In-hand Camera	External Camera
Frequency (Hz)	20	33	133	30	10

Table 7.3: Frequency of sensor and control signals on ROS.

TeleHand Customization As the DeltaHand’s kinematic twin, TeleHand is also modularized and its fingertips can be customized in different ways as shown in Fig. 7.14. The human user can directly manipulate the TeleHand by grabbing its links or tips. Alternatively, the human’s fingers can be more firmly attached to the TeleHand’s end-effectors by adding customized tips such as rings or finger gloves. With direct manipulation, it is easy to attach and detach the human fingers from the TeleHand while keeping the end-effectors of the TeleHand in the same position due to friction from the sliders. In this manner, human teleoperators can easily reposition their hands on the TeleHand for different motions, or even regrasp the TeleHand to only manipulate a subset of fingers while the others remain stationary. With additional customized fingertips, there’s a clear human finger to TeleHand finger to DeltaHand finger motion mapping. Through our experiments, we found that directly manipulating TeleHand is the easiest and most stable way for collecting demonstrations because humans are good at using hand synergies such as squeezing or spreading the four fingers along a desired axis using two hands.

Teleoperation Interface Comparison Visual tracking [13, 70, 174] has broadly been used for teleoperating anthropomorphic robotic hands. We characterize and compare two teleoperation interfaces: the TeleHand and visual tracking by using a Leap Motion camera as presented in DELTA-

Teleoperation	Communication	Demonstration Collecting Time (seconds)↓			Mapping Error (mm)↓		
Interfaces	Frequency (Hz)↑	Block Slide	Cap Twist	Shape Insert	Block Slide	Cap Twist	Shape Insert
Visual Tracking	10	54.05	81.51	14.47	0.60	0.34	0.31
Ours	133	16.99	31.11	11.81	0.23	0.23	0.20

Table 7.4: Teleoperation interface characterization and comparison with visual tracking by using a Leap Motion camera. Our proposed TeleHand has better performance on all metrics when evaluating communication efficiency, demonstration collection time, and mapping errors.

HANDS [177] on three metrics: communication efficiency, human demonstration collection time, and mapping errors from the teleoperation inputs to the robotic hand motions. We evaluate these metrics using three tasks: **Block Slide**, **Cap Twist**, and **Shape Insert**. For each task, we collect three demonstrations until task completion. We calculate the average performance over these three runs and report the results in the Table 7.4.

From the results, we observe that our proposed teleoperation interface, TeleHand, has better performance on all metrics. In addition, we visualize desired joint states and joint states from a demonstration of **Block Slide** in Fig. 7.15. We observe that by using the TeleHand, the signals are less noisy and the motions are simpler which helps policy learning. We also observe saturated joint states during teleoperation using the Leap Motion camera. This is because the human teleoperator may move their fingers outside the workspace of the DeltaHand which leads to an invalid inverse kinematics solution and a noisy trajectory as the human needs time to move their finger back within the workspace. Our TeleHand has one-to-one joint mapping and does not experience this issue.

7.6.3 Comparison with Existing Work

We compare our proposed system with state-of-the-art work from two aspects: a teleoperation system based on hardware development (Table. 7.5) and a learning system based on policy learning for dexterous robotic hands (Table. 7.6). For teleoperation system comparison, we show that our system has a relatively low cost while preserving high dexterity. The compliance of the soft links and fingertips of the DeltaHand also provides higher tolerance for policy deviations as compared to rigid hands. We plan to open-source our system to provide accessibility to potential users. For policy learning comparison, we show that our system has demonstrated capabilities on various tasks including grasping, translational, and rotational tasks. By leveraging end-to-end real-world imitation learning, policy learning is more data efficient and has less data distribution shifts. Our system also provides proprioceptive and visual feedback for closed-loop control.

7.6.4 Learning Policy Implementation Details

For an ablation study on using different imitation learning methods, we adapted the implementation of Behavior Cloning (BC), Implicit Behavior Cloning (IBC), and Diffusion Policy from [33, 214]. For fair comparison, we use the same ResNet18 image encoder adapted from Diffusion Policy [33] for all three methods. For BC and IBC, we use the same 5-layer MLP network architecture consisting of $\{N_{input}, 1024, 1024, 512, 256, N_{output}\}$ perceptrons, where N_{input} and N_{output} are the number of input and output channels. For BC, $N_{input} = (512 + 12) * 2$, where 512 is the dimension of the image feature, 12 is dimension of the joint state, and 2 is the observation horizon; $N_{output} = 12 * 8$, where 12 is the action dimension, and 8 is the execution horizon. For

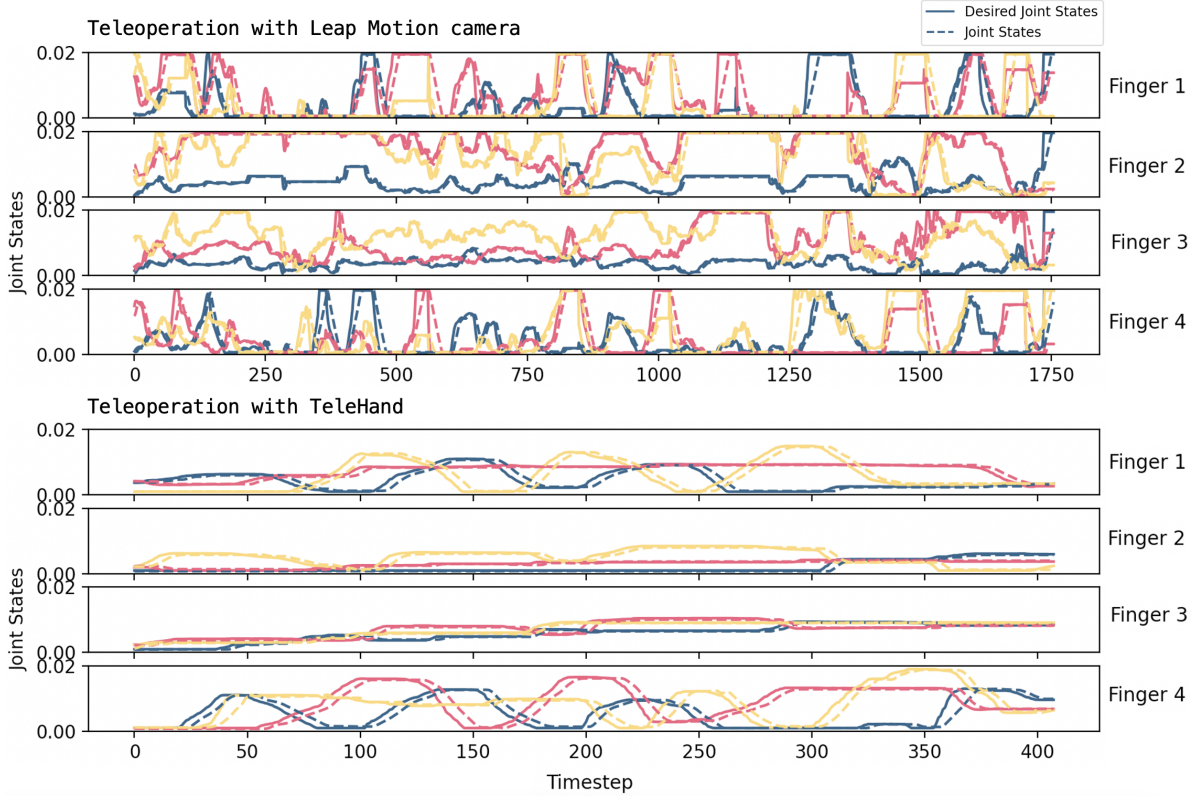


Figure 7.15: Visualization of the desired joint states and joint states of the DeltaHand when being teleoperated with a Leap Motion camera or TeleHand. We observe less noise and simpler motions from teleoperation with the TeleHand. With direct one-to-one joint mapping, the TeleHand does not suffer from kinematics constraints, which are shown as saturated joint states when teleoperating using the Leap Motion.

IBC, $N_{input} = (512 + 12) * 2 + 12 * 8$, where 512 and 12 are again the dimensions of the image feature and joint state respectively, 2 is the observation horizon, and 8 is the execution horizon; $N_{output} = 1$, is the energy of the observation-action pair. We use the same dataset, data normalization, data augmentation, training configurations including optimizer configuration, and number of training epochs for BC and IBC that we used to train the diffusion policy. For IBC training, we sample 256 random actions for each observation input; and during inference, we iteratively sample 1024 random actions 3 times with a noise scale starting from 0.33 and then reducing it by a factor of 0.5 in each iteration.

Teleoperation System	Robotic Hand	Cost	# DoF	Hand Type	Hand Material	Teleoperation Interface	Availability
DexPilot [70]	Allegro	\$15,000	16	Anthropomorphic	Rigid	Vision (RealSense)	Off-the-shelf
DIME [13]	Allegro	\$15,000	16	Anthropomorphic	Rigid	Vision (RealSense)	Off-the-shelf
LEAP Hand [174]	LEAP Hand	\$2,000	16	Anthropomorphic	Rigid	Vision (RGB Camera) Manus Meta Glove	Open-sourced
Stewart Hand [127]	Stewart Hand	\$600	6	Underactuated	Rigid	Space Mouse	Open-sourced
[123]	DASH Hand	\$1500	16	Anthropomorphic	Soft	Manus Meta Glove	Open-sourced
[190]	RBO Hand 3	\$2350*	16	Anthropomorphic	Soft	Vision (Webcam)	Open-sourced
Ours	DeltaHand	\$1,000	12	Exactly Constrained	Soft	Kinematic Twin	Open-sourced

Table 7.5: Comparison with other state-of-the-art teleoperation systems for dexterous robotic hands. We show that our system has a relatively lower cost while still preserving high dexterity. The DeltaHand has soft fingertips and a compliant finger structure which assists with adapting to different objects and environments and tolerating deviations from learned policies. *The estimated cost of the RBO Hand 3 is \$250 for manufacturing the hand, \$480 for 16 Freescale MPX4250 pressure sensors, and \$1600 for 16 pneumatic Matrix Series 320 valve controllers based on [50, 152].

Learning System	Robotic Hand	Task Types			Policy Learning		Feedback	
		Grasping	Translational	Rotational	Method	Environment	Proprioception	Vision
DIME [13]	Allegro			✓	IL	Sim & Real	✓	✓
LEAP Hand [174]	LEAP Hand	✓		✓	RL	Sim-to-Real	✓	
Visual Dexterity [27]	D’Claw [3]			✓	RL	Sim-to-Real	✓	✓
DEFT [89]	DASH Hand [123]	✓		✓	RL	Real		✓
[143]	RBO Hand 3 [152]		✓	✓	Motion Planning	Real		
[5]	Shadow Hand [42]			✓	RL	Sim-to-Real		✓
[135]	Yale Model Q [114]	✓	✓	✓	Motion Planning	Real		✓
Ours	DeltaHand [177]	✓	✓	✓	IL	Real	✓	✓

Table 7.6: Comparison with other state-of-the-art policy learning systems for dexterous robotic hands. We show that our system is evaluated on a variety of tasks including grasping, in-hand translation and rotation tasks. Leveraging end-to-end real-world imitation learning has the benefits of data efficiency and less data distribution shift.

Mixed Dexterity: Enabling Fine-Grained Manipulation with a Dexterous Third Finger

This chapter is based on [Zhang, Zhang, Si, Veloso, Temel, Gupta and Kroemer] that is currently under submission.

Abstract: Recent works in imitation learning have resulted in advances in dexterous manipulation with parallel jaw grippers, anthropomorphic hands, and dual arm versions of the above. However, oftentimes a small misalignment of the fingers to accomplish a fine-grained task may require moving the entire robot arm to make the adjustment, which may be challenging when two robot arms are working together in close proximity. In this work, we augment the existing strengths of parallel jaw grippers with a third additional dexterous finger that can assist in completing tasks that a parallel jaw gripper would not be able to accomplish by itself. We also integrate additional sensing modalities inside the third finger to enable force and vibration sensing when performing three delicate tasks: picking up a soft tortilla from a stack, removing a stick of gum from a pack, and opening the lid of a vitamin bottle. We demonstrate that with the additional sensing modalities, our trained multi-modal diffusion transformer models improve task performance both quantitatively and qualitatively over image-only diffusion policies.

8.1 Introduction

For robots to be useful in daily household chores such as cooking and cleaning, they need to interact with a wide variety of both rigid and deformable objects. We have seen impressive work with robots that utilize only parallel jaw grippers. However, parallel jaw grippers are still limited by their single degree of freedom and are typically unable to perform tasks beyond simple pick and place tasks or tasks such as opening a door, cabinet, or drawer [64, 206]. For more challenging dexterous tasks, we have seen work using two arms such as ALOHA [221]; however, it introduces additional complexity in collision avoidance and collaborative arm planning when performing tasks in close proximity.

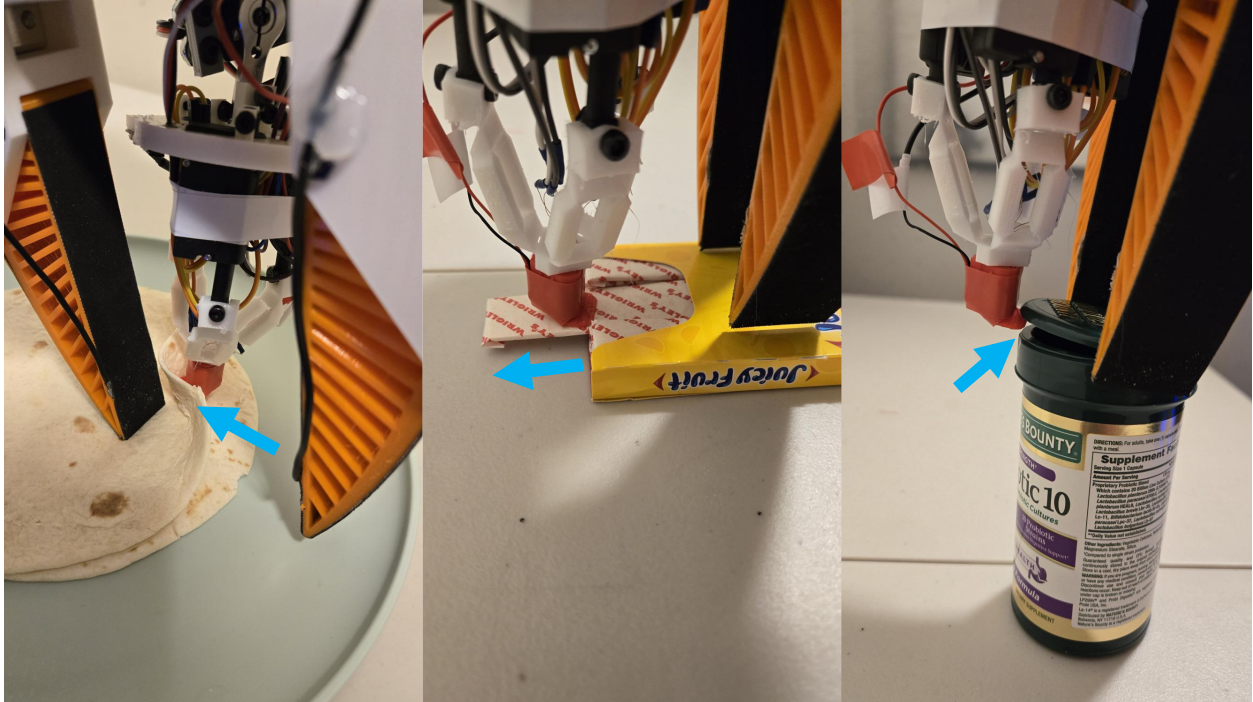


Figure 8.1: Illustration of the three tasks that the dexterous third finger can perform. In the left hand image, the third finger lifts up one flour tortilla so that the parallel jaw fingers can grasp it. In the center image, the third finger pushes down and out to extract a stick of gum from a pack. In the right hand image, the third finger lifts up the cap of a vitamin bottle.

Meanwhile, anthropomorphic hands are much more dexterous, but are harder to provide good demonstrations because of their higher degrees of freedom. With tendon-driven anthropomorphic robotic hands, their performance typically degrades over time and each joint may be linked to others, which increases the complexity of fine motor control for each joint. On the other hand, hands that are made up of actuators at each joint like the Allegro hand and the LEAP Hand [174] have higher precision control at each joint, but tend to be bulkier than tendon-driven hands which may make it difficult for them to manipulate thin or small objects dexterously.

In addition, when performing dexterous tasks, both of the above types of hands may require moving the robot arm to make fine adjustments. This arm movement could cause safety concerns with the environment because to reach a desired end-effector orientation, the entire arm may produce a large movement depending on the arm’s kinematics and workspace. This can lead to collisions when using dual arms that perform a task in close proximity, or it could hit something in a dynamic environment.

In this work, we present a dexterous third finger design that allows us to leverage the simplicity of a strong parallel jaw gripper, while also tapping into some of the potential of anthropomorphic hands with a dexterous third finger that mimics an index finger. Our system requires only one robot arm and involves first learning a simple visual servoing policy to position the robot arm close to the desired position. Afterwards, a second fine-grained policy uses the dexterous third finger to perform the task while the parallel jaw gripper either holds the object in place, or closes to pick up the object. We demonstrate the efficacy of our approach on three challenging tasks as shown in Fig.

8.1: picking up a soft tortilla from a plate, removing a stick of gum from a pack, and flipping open the cap of a vitamin bottle. In addition, we incorporate vibrotactile and force sensing to improve the performance of learned diffusion transformer policies both qualitatively and quantitatively over image-only diffusion policies.

8.2 Related Work

8.2.1 Three Finger Grippers

While there have been many three-finger grippers over the years, they tend to be either symmetric in that each finger has the same actuators such as the the ROBEL D’Claw hand [2] and the TriFinger [204] or a similar curling motion such as the soft gripper [199]. Another type of three-finger grippers have two fingers that can rotate while one finger just closes such as the Robotiq 3 finger adaptive hand, the Barrett Hand, and the Yale Open Hand Model O [139]. The Gelsight Svelte stands out from recent work in having integrated Gelsight sensors, as well as being able to perform a pinch grip, a lateral grip, and an opposition grip [220]. Finally, there have been other papers that actuate the fingertips to manipulate and rotate objects in hand such as the belt gripper [23], the roller gripper [213], and the Stewart Hand [128]. In contrast to all of the above works, we leverage the powerful capabilities of a typical Panda Hand parallel jaw gripper together with finray fingers and augment it with a highly dexterous third finger that can also be stowed away if it is not needed for a task.

8.2.2 Teleoperation for Imitation Learning

Behavior cloning (BC), or training models to mimic expert behavior, has been a popular strategy to train robots over the last many decades [150, 151, 167] and has seen a recent resurgence in interest [85, 226]. A wide variety of teleoperation systems have been used to collect expert behavior on robots such as using virtual reality and mixed reality headsets: OPEN TEACH [82], HoloDex [11], Open Television [30]; RGB Cameras: Dexpilot [69], DIME [12], Robotic telekinesis [181], AnyTeleop [155]; and kinematic twins: GELLO [200], Tilde [178], ALOHA [221], ALOHA 2 [6], and Mobile ALOHA [61]. This has led to a diverse set of large-scale robotics/imitation learning datasets in both the real world as well as simulation such as: DROID [92], BC-Z [85], BridgeData V2 [196], RT-1 [22], RT-X [138], RoboTurk [116], and Robomimic [117]. Others have developed portable teleoperation setups in order to collect data in the wild to transfer to the robot such as Grasping in the wild [182], VIME [211], DexCap [197], and UMI [35]. In our case, we use a 3D space mouse to learn a visual servoing policy using inspiration from [87] to approach the target object. Then once the robot is in approximately the correct place, we use a kinematic twin of the third finger to manipulate the object while the parallel jaw grippers hold the object in place.

8.2.3 Learning for Dexterous Manipulation

Several recent works use only vision to learn dexterous manipulation tasks on dexterous end-effectors such as OpenAI’s in-hand manipulation [10], Visual Dexterity [28], VideoDex [175], FISH [68], and Tilde [178]. Others use various sensors such as tactile: TAVI [66] and T-Dex [67].

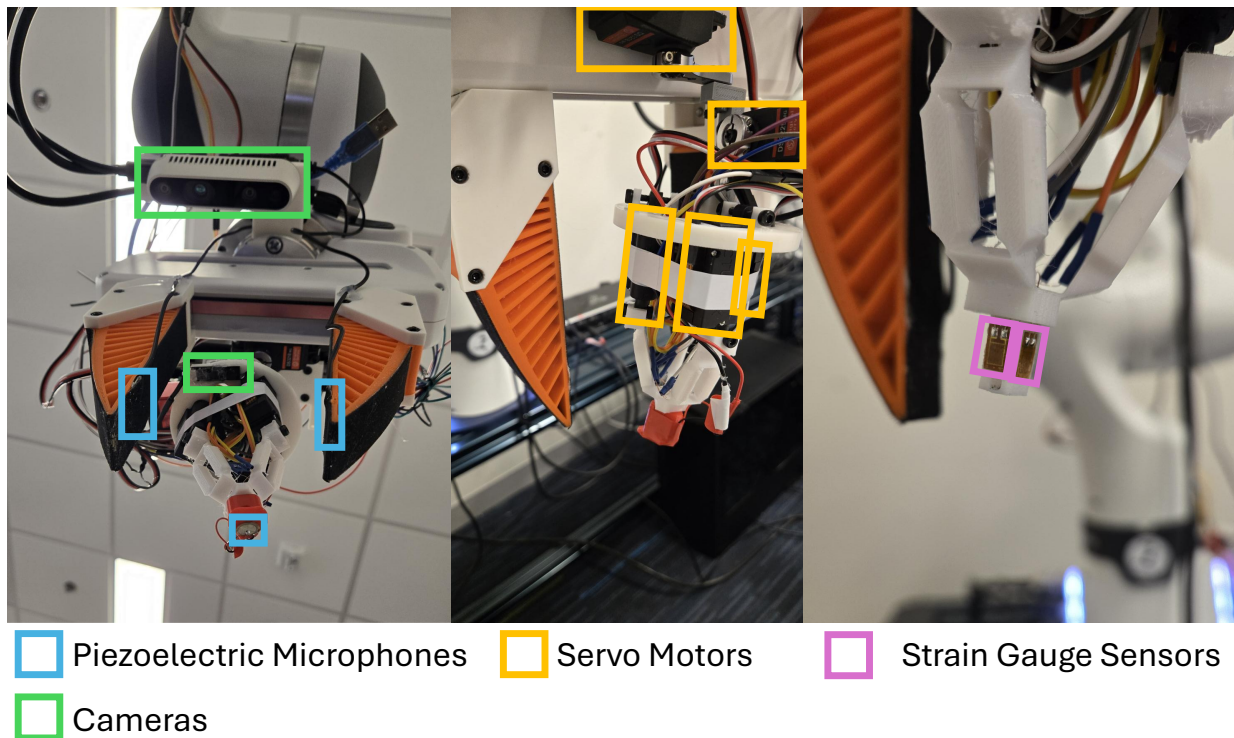


Figure 8.2: **Robotic Platform for Online Testing:** We conduct our experiments on a Franka Panda robot. In the depicted configuration, we have an Intel RealSense D435 as well as an in-hand Adafruit 640x480 USB spy camera. We have attached 3 piezoelectric microphones, 1 on each of the three fingers. The third finger consists of 5 servo motors, 2 rotational and 3 linear actuators that make up the Delta finger. Finally, under the red finger, we have 4 strain gauge sensors that are connected to an Adafruit HX711 connected through 2 pairs of Wheatstone bridges to sense bending and contact with the fingertip.

We leverage Diffusion Policies [32] to learn a joint-based policy for the dexterous third-finger and modify the diffusion policy to take in additional inputs from force and vibrotactile sensors.

8.3 Robot Setup

8.3.1 Third Finger

Our robot setup consists of a Franka Emika Panda robot with the default Panda Hand but with substituted finray fingers from UMI[35] as shown in Fig. 8.2 to conform to soft objects. In addition to the Panda Hand, we have a 6 degree of freedom third finger that consists of a 100mm linear rail prismatic joint, two 270° DSServo rotational servos from Amazon, and a delta end-effector consisting of three 6V Actuator PQ-12R 20mm travel micro linear servos that are connected together with a 3D printed hybrid TPU/PLA linkage from [178]. The first rotation servo mimics the ability of the index finger to rotate between the thumb and middle fingers to find edges when going underneath items. The second rotation servo mimics the curling motion of the fingers. The delta fingertip has three translational degrees of freedom that allows precise insertion between

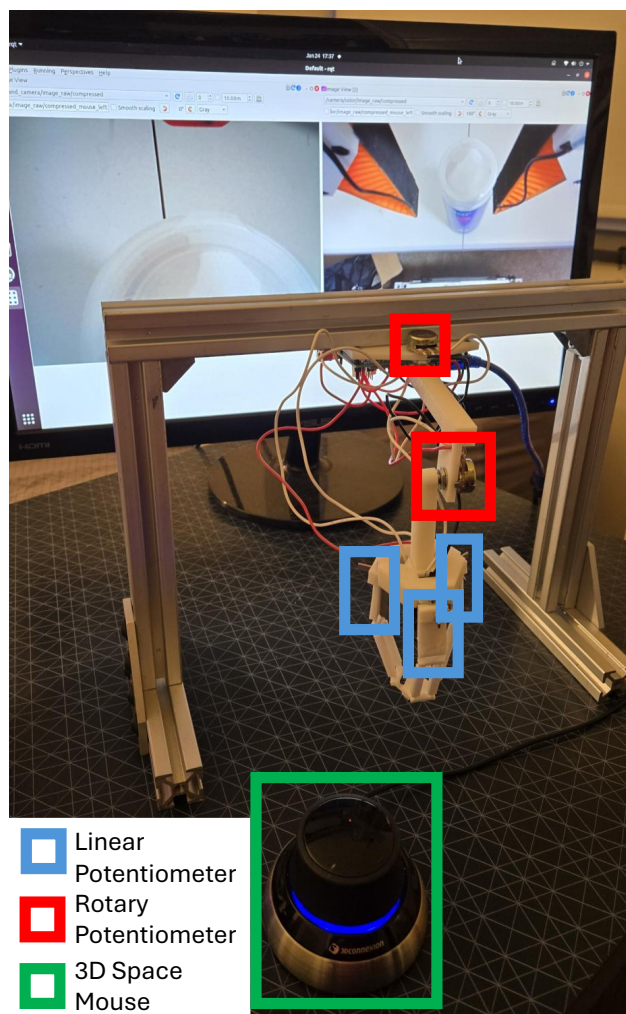


Figure 8.3: **Teleoperation Setup:** We create a kinematic twin to the third finger on the robot and utilize rotary and linear potentiometers to sense the desired joint state of each actuator and command the third finger directly to mimic the positions. We use the 3D Space Mouse in order to move the entire robot to learn the servoing policy.

thin objects within its workspace. For each of our experiments, we do not actively utilize the first prismatic joint, but it can be used to stow away the third finger when it is not needed.

Next, we have two cameras on our robotic setup. The first camera is an Intel Realsense D435, which is mounted near the wrist and provides a view of the two fin ray grippers for servoing to an object. The other camera is a 640x480 Adafruit USB spy camera mounted on the delta platform that holds the 3 linear actuators and provides a close up view of the third finger’s fingernail when it is interacting with thin objects. In addition, we have equipped each finger of the robot with a piezo-electric contact microphone hidden underneath tape. These three contact microphones are connected to a Behringer UMC 404HD USB Audio Interface. Finally, we have four BF350 350 ohm strain gauge sensors on each side of a pillar underneath the cover of the third finger. They provide force feedback in two directions when the third finger makes contact with objects. We read from the strain gauge sensors using an Adafruit HX711 connected to 2 pairs of Wheatstone

bridges. We use an Arduino nano to control all of the motors and read in the sensor values from the Adafruit HX711.

8.3.2 Teleoperation

Our Teleoperation setup is shown in Fig. 8.3. We have one monitor that is streaming the camera views from both the wrist camera as well as the finger camera. In addition, we have a 3D Connexion SpaceMouse Compact that provides human input to move the robot arm in 6 DoF end-effector space. Finally we have a kinematic twin to the third finger that allows us to control each servo of the third finger by moving the links. The kinematic twin consists of two 10k Ohm rotary potentiometers that match the two rotational servos and three 20mm 10k Ohm sliding potentiometers that match the 20mm linear actuators on the Delta finger. All of the potentiometers are connected to an Arduino nano that is connected to a computer that communicates the joint states using ROS and pyserial.

8.4 Method

MixDex contains a coarse-grained servoing policy and a fine-grained dexterous third-finger policy. Following the observation that the approaching segment of a skill is mostly free-space motion, the servoing policy π^{servo} (Section 8.4.1) is a straight-forward regression-based network that guides the robot arm to a location where the third-finger can perform the fine-grained segment. The third-finger policy π^{tf} is a transformer diffusion model that takes in multi-modal sensory input (*i.e.* vision, force, and vibro-tactile) and predicts future joint actions to move the third-finger (Section 8.4.2). In order to collect high-quality demonstrations for training π^{tf} , we employ 2 techniques: firstly, we condition the data collection process on locations that π^{servo} ends up at, and secondly, we build a kinematic twin (Figure 8.3) of the third-finger so that the data collection process is intuitive for the demonstrator.

8.4.1 Servoing Policy

First of all, we need the robot to position itself appropriately relative to the target object in order to provide demonstrations for the third finger’s task. To do so, we leverage the process from the paper titled Course-to-Fine Imitation Learning [87]. We first have a human use the 3D SpaceMouse to move the robot in 6 DoF such that the robot is properly positioned for the follow up task. Next, we program the robot to lift up a specified amount depending on the accuracy of the positioning required. The lower we lift up, the more accurate the servoing position will be. For example, for the soft tortilla task, we lift the robot up 5cm, while in the vitamin bottle task, we lift up the robot 2.5cm. After the robot raises itself up, we record the robot’s end-effector position as well as the wrist camera image at that location.

Afterwards, we record around a 2 minute rosbag where the human operator uses the 3D SpaceMouse to servo around the object in the X, Y and Z axes in order to collect a dataset of robot end-effector positions and wrist images. After the human operator is finished, we terminate recording the rosbag and post-process each image with a label that is a scaled relative transformation to the original image above the desired placement location. In addition, we label the image as done if the

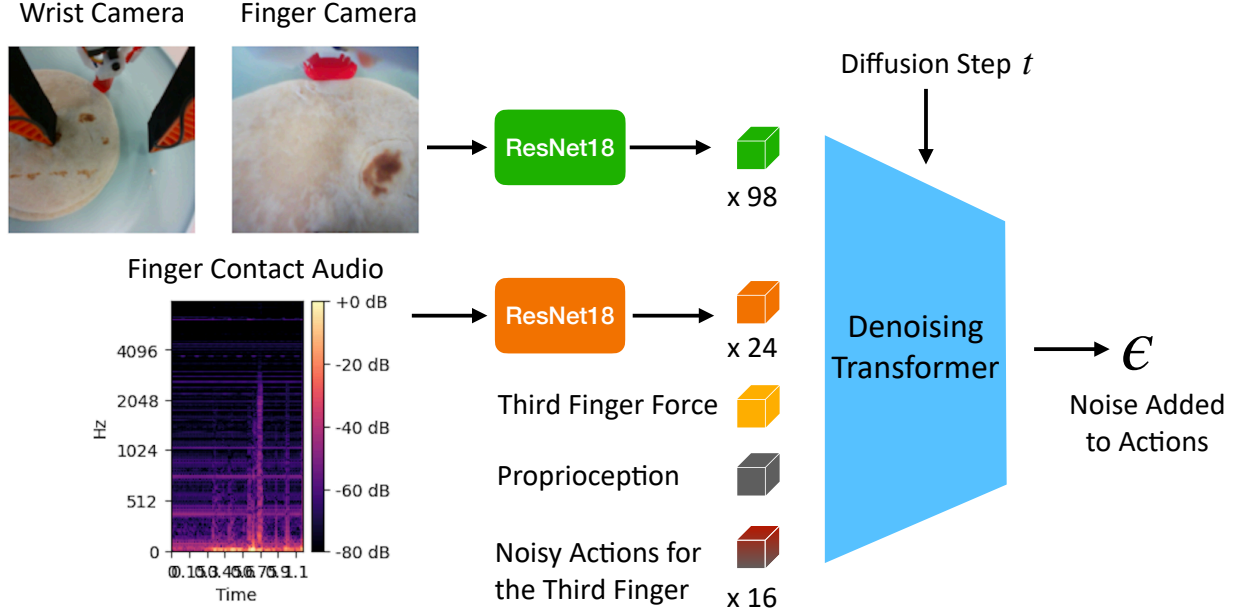


Figure 8.4: Our Multi-modal Diffusion Transformer Network Architecture which takes in camera images from both the Wrist Camera and the Finger Camera as well as the mel spectrogram from the third finger contact microphone. In addition, tokens representing the third finger’s joint states and force are fed into the Denoising transformer to predict the noise that was added to the action.

relative translation transformation is within 1cm of the original pose. We repeat the data collection 15 times for each task and then train a simple visual servoing policy with ResNet18 to output the transformations and servoing progress. In total, it takes around 45 minutes to collect the servoing data for each task.

8.4.2 Third Finger Policy

Data Collection

To collect data for the third finger policy, we condition the data collection process on the servoing policy π^{servo} . Specifically, we first train a robot servoing policy and then deploy it so that the end-effector is positioned at a strategic location that allows the third-finger to perform the fine-grained aspect of the tasks. We utilize the servoing policy to position the robot end-effector at the beginning of each demonstration. The benefit of doing so is that the demonstrator can provide necessary corrections on the third-finger to compensate for any errors π^{servo} might produce. The third-finger diffusion policy π^{tf} can thus learn to correct itself, thereby maximizing the chance of online task success.

After the robot end-effector is at an appropriate location, we start the third finger and use the kinematic twin, as shown in Figure 8.3, to teach the fine-grained manipulation task and record the third-finger’s joint trajectories. We only look at the two camera views to perform the task to make sure that the demonstrations are learnable from the sensors on the third finger.

Policy Architecture

As shown in Figure 8.4, the third finger policy takes as input the two camera views, audio and force data from the fingertip sensors, and finger proprioception. We first use librosa [130] to transform 1.114s of audio history from the fingertip microphone to a mel spectrogram like the one shown in Figure 8.4. Then we use separate ResNet18 models to encode the camera and mel spectrogram audio observations; both networks are trained from scratch. The RGB-input ResNet outputs $7 \times 7 = 49$ featurized tokens for each of the wrist and finger view, resulting in a total of 98 RGB tokens. The audio-ResNet outputs $8 \times 3 = 24$ tokens for the contact microphone input. Since π^{tf} outputs 16 future joint actions, there are 16 action tokens, one for each noisy action. Lastly, the force and proprioception input are processed using MLPs.

A decoder-style transformer model consisting of cross- and self-attention layers is employed as the diffusion head to predict noise added to the finger joint actions. The diffusion step t is input into the network via FiLM conditioning [147]. We add sinusoidal position embedding to the input observation and action tokens before feeding into the transformer.

8.5 Experiments

8.5.1 Soft Tortilla

The soft tortilla task involves placing two soft Mission branded Fajita style Flour Tortillas on a mint green plastic plate from Target. We place the two tortillas with different amounts of overlap and varying positions on the table and plate in 20 trials for each of the four trained networks with results shown in Table 8.1. Our visual servoing policy servos the robot arm to approximately the middle of the right side of the stacked tortillas and then tilts -9 degrees around the x axis to lift up the right fin ray finger so that when the parallel jaw gripper closes, it won't accidentally pinch the bottom tortilla.

What we observed was that the vision-only diffusion transformer performed the worse qualitatively because it did not have access to the additional sensor modalities. Thus, occasionally, it would utilize too much force when lifting up the side of the tortilla and rip it as shown in Fig. 8.5. Meanwhile, most of the failures resulting from the third finger policy that had access to all the inputs was that it would accidentally pick up both tortillas at once instead of just the top tortilla. In addition, it would sometimes fold the tortilla a little bit when closing the gripper as shown in Fig. 8.6.

On the other hand, the vision and audio policy and the vision and force policy would often close the gripper when the finger had not gone under the edge of the top tortilla to lift it up. These two policies would also sometimes poke small holes in the tortilla as shown in Fig. 8.7. It was surprising that these two policies performed worse than the vision-only policy, but the policy that contained all the inputs performed the best.

8.5.2 Gum

For the Gum task, we opened a pack of Juicy Fruit gum and removed 1 stick of gum from the center column. Afterwards, we trained the visual servoing policy to position the robot's gripper to be on top of the Juicy Fruit logo. Then we demonstrated removing a stick of gum from the center

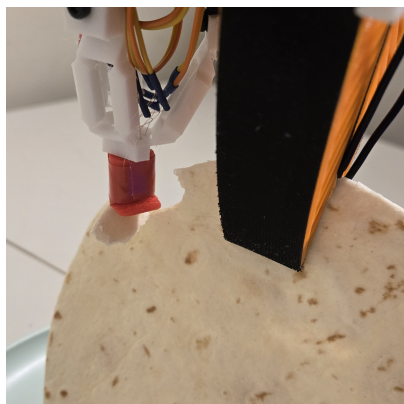


Figure 8.5: Ripped Tortilla



Figure 8.6: Folded Grasp



Figure 8.7: Small Hole

Task	Vision	Vision + Audio	Vision + Force	Vision + Force + Audio
Soft Tortilla	60%	40%	50%	75%
Gum	70%	80%	60%	90%
Vitamin Bottle	20%	10%	20%	30%

Table 8.1: Ablation study over the inputs on each of the three tasks.

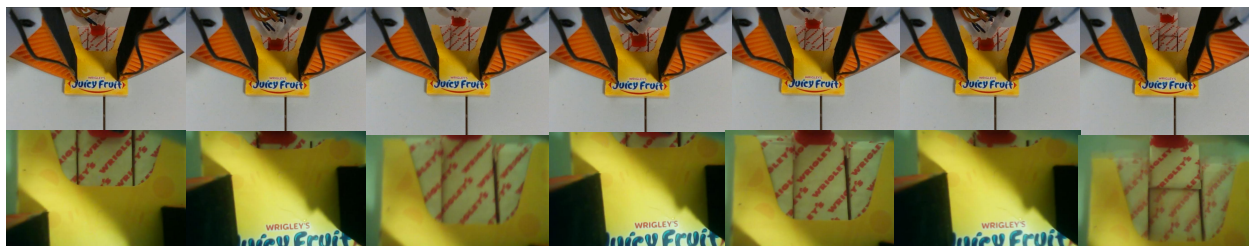


Figure 8.8: Synchronized images from the wrist and finger cameras from a successful gum inference trial



Figure 8.9: Third finger stuck behind the gum sticks

column by using the third finger to push down on the stick and then push out in order to slowly move the gum out of the pack. Once the gum is sticking out sufficiently, we consider the task to be complete. Fig. 8.8 shows one successful inference trial of the gum task.

During the gum inference trials, we observed that the vision-only approach tended to exert a lot of force on the stick of gum, which sometimes resulted in breaking the stick in half. On the other hand, our method using the force sensor and contact audio took more steps on average to complete, but was more gentle with the gum. The success rate between the vision only and our method was around the same over the course of 10 trials. Some of the failure modes we observed was when the finger nail got stuck under the top stick of gum and would push it back into the pack as shown in Fig 8.9. Other times the vision only policy would attempt to slide out the gum, but it would not push down enough and then the fingernail would slip and no movement in the gum would be observed.

8.5.3 Vitamin Bottle

The vitamin bottle was the most difficult task, which resulted in the lowest success rate across the tasks and methods. This is because the vitamin bottle cap requires a decent amount of force and leverage in the right direction to open it. We first learned a servo policy to position the gripper around the center of the cap and then we move the robot 2.5cm down and 2cm back in order for the fingers to stay out of the way of the cap opening as well as to give space for the third finger to pull up on the cap. Oftentimes though, the third finger would slip from the small ledge that is present in the bottle while pulling up because the delta links are compliant. Some experiment runs for the vitamin experiment are shown in the attached supplementary video.

8.6 Limitations

While our mixed dexterity setup can open up a range of new tasks for a parallel jaw gripper, it still falls short in design for a variety of tasks that may be desirable. For example, the Delta finger is made up of hybrid PLA and TPU links which aren't able to exert a lot of force as noted with the vitamin bottle experiment. Thus, it may be unable to perform tasks such as pressing down on the sprayer of a whiteboard cleaning bottle or removing a book from a bookshelf. In addition, when the parallel jaw grippers are attempting to close on a very small object, the platform that holds the linear actuators of the delta finger and the in hand camera will bump into the two fin ray grippers and will not be able to push objects into the two grippers such as a peanut m&m on a plate as shown in Fig 8.10. Finally, the third delta finger is still a little big, so it is difficult to maneuver in tight spaces such as inside a container.

Additionally, the third finger's teleoperation system is a bit bulky and may require two hands to operate correctly. This makes collecting data in the wild without a mobile platform extremely difficult unlike UMI [35]. We also did not test the generalizability of this system to different environments.



Figure 8.10: Contact between the Delta Finger platform and the two Fin Ray fingers when the Fin Ray fingers are fairly close prevent the third finger from being able to go between the fingers and help with a M&M singulation task.

8.7 Conclusions and Future Work

In conclusion, we have presented a new design for a mixed dexterity setup that combines the advantages of a parallel jaw gripper with a highly dexterous third finger. The third finger with the included force and vibrotactile sensors is able to manipulate soft delicate objects more carefully, resulting in potentially better satisfaction for end users. The third finger system also enables tasks that were previously impossible for a single parallel jaw gripper without another arm such as picking up a single tortilla, taking out a stick of gum, and opening a vitamin bottle cap. All the tasks were trained with just 15 servoing demonstrations and less than 50 expert demonstrations. In the future, we plan to simplify the design so that it is stronger and more dexterous on the axes that are relevant to certain tasks.

Conclusions and Future Work

9.1 Conclusion

In this thesis, we covered three main sections of multimodal sensing, dexterous hand designs, and finally learning adaptive policies utilizing the multimodal inputs to perform fine-grained tasks autonomously on the dexterous hands. In the multimodal sensing section, we first explored how incorporating multimodal sensors can improve the capabilities of parallel jaw grippers to provide self-supervisory information to learn to distinguish material properties between different food items. Afterwards, we characterized the performance of contact microphones in a manufacturing environment and exposed them to a variety of external disturbances to verify the efficacy of using vibrotactile sensing to monitor insertions for flexible manufacturing. Additionally, with a novel magnetic force and localization sensing system, we enabled an off-the-shelf parallel jaw robot to perform delicate tasks such as pipetting and inserting a key into a lock. In the second section we focused on designing a compliant soft dexterous end-effector that could safely interact with delicate objects such as pills or grapes. We additionally demonstrate that this design can be miniaturized, modularized, and customized to suit tasks that require fewer degrees of freedom. Finally, we combine the above two sections into a unified framework of using multimodal sensing and dexterous Delta fingers to autonomously perform fine-grained manipulation tasks through diffusion policies. The sensors, end-effector designs, and system pipelines developed in this thesis can help robots perform more challenging fine-grained manipulation tasks, and will help progress the field of robotics towards more robust and dexterous robotic assistants.

9.2 Key Takeaways and Insights

In this section, I would like to go over some key takeaways and insights that I have developed over my time at CMU working on a wide variety of tasks with different robot platforms. I have broken them up similarly to the sections in the Thesis above, but also added some extra thoughts about system design for tasks as some projects I worked on did not make it into this Thesis.

9.2.1 Sensor Design

First of all, I think sensor design is very important for the future of robotics. There are a lot of things that can be done with just vision; however, I believe that multi-modal sensing is necessary to do tasks robustly especially in environments or tasks with heavy occlusion. Thus, sensor design is extremely important for future domestic robots. So, first of all, what I realized is that having a robust sensor design is one of the most important things to focus on. For example, the Finger Vision and Gelsight sensors have amazing capabilities by leveraging vision sensors; however, they contain gel-like skins that are currently fragile and can be easily damaged, which would require frequent replacements and recalibration that could make previous data useless. Instead, the contact microphones and the magnetometer that was under the fingertip were much more robust and did not need exchanging. However the magnetic stickers did need to be replaced, but as long as they were manufactured similarly, we had to change no code to perform the same remote localization task reliably.

Next, what is important is consistency with sensors. I previously used a thermal camera to track eggs cooking over time, but the values were never stable and would often change over time or within the same recording. Having a consistent calibration technique or making sure that the robot is able to tare its values consistently helps out with robot learning with noisy sensors. In addition, injecting Gaussian random noise into the collected data during the training process improves the robustness of the learned neural network similar to how image augmentation techniques are used for training computer vision tasks. Also a very important part of training anything with offline data is making sure that synchronization between the different sensor modalities is correct. Thus, developing a GUI to quickly visualize and check to make sure that the data is correctly synchronized before training is always one of the most worthwhile things to do.

9.2.2 Robot Hand Design

In terms of robot hand design, I have extensively used the compliant Delta fingers, which are very dexterous in a small translational workspace, but there are still some drawbacks to the design with their limited grasping force. I also built my own anthropomorphic hand with tendons. What I realized is that compliance is very important in robotic hands because it can help deal with misalignments from the learned robot policies without causing catastrophic damage such as accidentally breaking an egg. However, compliance in the joints does require strong "bones" in the links to provide more rigidity and power for lifting objects. When I first designed my anthropomorphic hand, I didn't realize how important the wrist design would be. There are many issues with not having a strong wrist as the robot hand often flops back and forth. However, I think a wrist still needs to have enough compliance such that if it accidentally strikes a table or object, it will not break but instead have a light impact. Thus, I would invest a decent amount of time in a wrist design that is compliant, but also mimics a ball joint with a hole in the middle to route all of the tendons. I also should have shielded all of my tendons in sleeves around sharp edges to prevent degradation.

I don't believe that direct driving each finger with a motor on the hand is enough to perform dexterous tasks. I believe that some hybrid design of tendon driven fingers mixed with direct driving would be the best solution such as the robot nano hand, which uses some micro servos to provide certain rotational degrees of freedom on some fingers. I should have used a similar design

because the index finger rotation on my anthropomorphic hand was not as strong as I would have liked. I also think more tendons need to be focused in the thumb, index, and middle fingers. What is needed is the ability to perform precise forward kinematics on each joint of the index finger so that it can do a variety of local fine-grained movements similar to my last Mixed Dexterity project. The pinky and ring fingers are not as useful in my experience. However, having a soft palm in the hand to rest the objects is very useful as often times in the Mixed Dexterity project, the object may rotate in hand between the parallel jaw fingers when the third delta finger is trying to do manipulation.

In terms of integrated sensors in a robotic hand, I believe that incorporating a camera within the hand is definitely useful in visual servoing when the hand may be occluding the object from cameras in the robot head. I also believe that having fingertip force sensors and contact microphones are very useful for performing many tasks. However, only putting them on the thumb, index, and middle fingers would be sufficient for many tasks. I do not believe that having a high fidelity force sensor in the palm is necessary for many tasks.

One interesting design change to have fingernails like humans was a great asset for thin objects and I believe more robot designs should incorporate them. Although I would potentially change the design of the finger from my Mixed Dexterity to have the finger nails be more easily swapped out or made out of a harder material such as metal so they don't get chipped as often like our human fingernails.

9.2.3 System Design for Tasks

In terms of system design for tasks, first it is important to understand which degrees of freedom will a robot be using for the task and whether the robot you select will be able to perform the task. For example, some planar pick and place tasks are just better on the industrial robots such as the UR5e and the Yaskawa GP4 because of their rigidity, precision, and their kinematics. However, if you want to open a door, they may not be able to perform the task without a mobile base.

Next, it is important to properly determine the sensors that you will need for the task. Oftentimes only a wrist camera and force sensor on the robot wrist is sufficient to perform the task. However, if you want to capture more information to protect delicate objects, you might want to add some additional sensors such as contact microphones or tactile sensors on the finger tips. In addition, if the wrist camera view is heavily occluded by a handheld object or it takes too long to move the robot up to scan the environment, an external view or two is very useful.

Next, we have the data collection pipeline for training. I always used ROS and rosbags to save and synchronize the various sensors and robot states, but it is also possible to do it with other methods. I would determine what kind of ground truth object tracking you need and whether to use segment anything [94] and label a dataset for a success classifier for a task or whether aruco markers would be sufficient as a minimum viable product. Once you have a reliable data labeler, I would write a self supervised data collection script to have the robot interact with the object of the task to collect sensor values from various interactions. If you are utilizing a learning from demonstration framework, I would instead determine what kind of high quality robot teleoperation setup is necessary and how many degrees of freedom to control in each phase of a demonstration.

After the data collection process is complete for around 10 test trials, I would process the data and make sure it is all synchronized across the modalities. Sometimes if different sensors aren't connected to the same computer, you will need to make sure the two clocks on the computers are

synchronized each time you collect data because the time drift for high frequency data such as audio can be affected by small time differences. Once you have synchronized data, you will need to determine what kinds of data augmentation techniques you want to use for each input and also how you want to represent the data as raw or as tokens. Finally I would advise training simple models first just to test that something can be learned from the data before moving onto more complex models.

The last part is evaluating the robotic system. Since I used ROS, I would have a real-time node that loads in the weights of trained neural network and then transforms the inputs the same way as in training to finally output either the next actions or the classification result. It is also important that the data processing pipeline between training and inference are exactly the same or else many issues will show up that may not be seen during training.

9.2.4 Lifelong Learning

Finally, with respect to lifelong learning, robots will need to understand the quality of data themselves and determine when to add new data to the training set while deleting older outdated data. For example, sometimes vibrotactile data collected after some amount of time sounds different and the trained networks do not perform as well as before. At this point, a new training dataset should be created as a mix of the two datasets with quality data from each while removing the less informative from the two datasets. This is because the more data we have, the longer it takes to train a model, but giving the model more training data may not necessarily always improve the end model. Instead, having high quality data in diverse environments can help the models generalize better.

In addition, I think the current robot policies trained using diffusion on human demonstrations don't have a great representation of safety. Failed demonstrations are typically taken out of the training data set because they can mess with the learning process, but these failures are important to help robots be safer in the future. Thus, incorporating the failed demonstrations into these imitation learning frameworks with additional reinforcement learning should be done because it is time consuming to collect data in the real world and the data may be useful.

9.3 Future Research Directions

While I have demonstrated a lot of progress on a variety of tasks using a single robot arm, I believe that bimanual robots are still necessary for a variety of applications in manufacturing, restaurant, and domestic environments. Although there have been great advances in bimanual dexterity with parallel jaw robots such as ALOHA2 [6] and Mobile Aloha [61], I believe that anthropomorphic hands will continue to enable further advancement in automation in a variety of tasks that are strenuous for humans, such as packing heavy dog food bags into cardboard boxes. Thus, I will go over several research directions for manipulation that I still think are unsolved, but are necessary in order for robots to be more widely adopted in the real world.

9.3.1 Enhanced Dexterity and Precision

While our Mixed Dexterity approach achieved improved success rates on tasks such as picking up a soft tortilla with a third finger, it will be a challenge to improve the dexterity and control of anthropomorphic robotics hands to imitate the local control that the Delta finger contains. This could involve exploring:

- Higher degrees of freedom in the index finger: Many of the compact anthropomorphic hand designs have only one curling actuator for the index through pinky fingers. However, I believe that the index finger is a very versatile finger that is important for many dexterous grasping tasks on small or thin objects. Thus, I believe that more additional degrees of freedom should be implemented instead of only having additional degrees of freedom in the thumb.
- Additional sensors on fingertips: Many anthropomorphic hands only have force tactile sensing with taxels, however, incorporating vibrotactile sensing into fingernails on anthropomorphic fingers could provide additional vibrotactile feedback for a variety of fine-grained tasks. In addition, I believe that in-hand cameras contributed massively to the success of my last two works, so adding an in-hand palm camera to help the fingers servo to the correct grasping positions could help with learning more robust dexterous hand policies.
- Clutch system for holding heaving objects: I also think that some sort of locking mechanism is necessary in each finger to grasp heavy objects to avoid excessive wear to the finger motors or degrading the tendons. An electric clutch to stop the finger from curling back when holding a bag of groceries could greatly improve the energy efficiency of the hand under heavy loads.

9.3.2 Generalization and Robustness

Another key challenge is to improve the generalization and robustness of the learned manipulation skills. While I showed that the learned policies could adapt to different instances of the same food items, there needs to be a large, high quality dataset that involves different background environments to train better more general policies for similar tasks. Future research should investigate:

- Real-to-sim-to-real transfer: Bridging the gap between simulated and real environments is crucial for scaling data collection on out-of-distribution environments without costly data collection in the real world. There have been many advances in training robots to walk in a variety of environments using reinforcement learning in simulation, but simulators still need to be improved in order to properly model the diversity of object interactions in the real world.
- Learning from diverse datasets: We have seen that leveraging pre-trained vision language models can improve robot policy performance [17, 149], but there still aren't many high quality robotic datasets, so we as a community have to somehow create these datasets to capture more of the diversity of environments in the real world. Then training on these larger and more diverse datasets can improve the generalization capabilities of the learned skills on a wider range of objects, tasks, and environments.

- **Robustness to disturbances:** Developing controllers that are robust to external disturbances while still ensuring safety is paramount for deployment in collaborative human-robot environments. For example, adapting to unexpected contacts or changes in the object position is essential for reliable performance in unstructured environments.

9.3.3 Application to Real-World Tasks

Ultimately, the goal is to increase the adoption of robots in manufacturing and household settings. Thus, they need to be proven in the real world environments with easy to use interfaces for non-experts. Future work should focus on:

- **Scalability of deployment with a wide range of robotic platforms:** Integrating the learned manipulation skills with various robotic platforms, including different combinations of embodiments and perception systems, will allow for more generalization to complex tasks and environments.
- **Simple human-robot interactions and task specification:** Developing intuitive interfaces for users to specify desired manipulation tasks and provide feedback is crucial for practical applications. This could involve natural language interaction or visual programming interfaces.
- **Evaluation in realistic scenarios:** Extensive testing and evaluation in real-world environments, such as assembly lines or home kitchens, are necessary to validate the effectiveness and robustness of the proposed approaches.

By addressing these challenges, future research can pave the way for dexterous robots that can perform a wide range of fine-grained manipulation tasks, revolutionizing both manufacturing processes and everyday life.

Bibliography

- [1] Sylvain Abondance, Clark B Teeple, and Robert J Wood. “A dexterous soft robotic hand for delicate in-hand manipulation”. In: *IEEE Robotics and Automation Letters* 5.4 (2020), pages 5502–5509.
- [2] Michael Ahn, Henry Zhu, Kristian Hartikainen, Hugo Ponte, Abhishek Gupta, Sergey Levine, and Vikash Kumar. “ROBEL: RObotics BEnchmarks for Learning with low-cost robots”. In: *Conference on Robot Learning (CoRL)*. 2019.
- [3] Michael Ahn, Henry Zhu, Kristian Hartikainen, Hugo Ponte, Abhishek Gupta, Sergey Levine, and Vikash Kumar. “Robel: Robotics benchmarks for learning with low-cost robots”. In: *Conference on robot learning*. PMLR. 2020, pages 1300–1313.
- [4] Baris Akgun, Maya Cakmak, Karl Jiang, and Andrea L Thomaz. “Keyframe-based learning from demonstration: Method and evaluation”. In: *International Journal of Social Robotics* 4 (2012), pages 343–355.
- [5] Ilge Akkaya et al. “Solving rubik’s cube with a robot hand”. In: *arXiv preprint arXiv:1910.07113* (2019).
- [6] Jorge Aldaco et al. “ALOHA 2: An Enhanced Low-Cost Hardware for Bimanual Teleoperation”. In: *arXiv preprint arXiv:2405.02292* (2024).
- [7] John R Amend, Eric Brown, Nicholas Rodenberg, Heinrich M Jaeger, and Hod Lipson. “A positive pressure universal gripper based on the jamming of granular material”. In: *IEEE transactions on robotics* 28.2 (2012), pages 341–350.
- [8] Heni Ben Amor, Oliver Kroemer, Ulrich Hillenbrand, Gerhard Neumann, and Jan Peters. “Generalization of human grasping for multi-fingered robot hands”. In: *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. 2012, pages 2043–2050.
- [9] OpenAI: Marcin Andrychowicz et al. “Learning dexterous in-hand manipulation”. In: *The International Journal of Robotics Research* 39.1 (2020), pages 3–20.
- [10] OpenAI: Marcin Andrychowicz et al. “Learning dexterous in-hand manipulation”. In: *The International Journal of Robotics Research* 39.1 (2020), pages 3–20.

- [11] Sridhar Pandian Arunachalam, Irmak Güzey, Soumith Chintala, and Lerrel Pinto. “Holo-Dex: Teaching Dexterity with Immersive Mixed Reality”. In: *2023 IEEE International Conference on Robotics and Automation (ICRA)*. 2023, pages 5962–5969.
- [12] Sridhar Pandian Arunachalam, Sneha Silwal, Ben Evans, and Lerrel Pinto. “Dexterous Imitation Made Easy: A Learning-Based Framework for Efficient Dexterous Manipulation”. In: *2023 IEEE International Conference on Robotics and Automation (ICRA)* (2022), pages 5954–5961.
- [13] Sridhar Pandian Arunachalam, Sneha Silwal, Ben Evans, and Lerrel Pinto. “Dexterous imitation made easy: A learning-based framework for efficient dexterous manipulation”. In: *2023 IEEE international conference on robotics and automation (icra)*. IEEE. 2023, pages 5954–5961.
- [14] Dominik Bauer, Cornelia Bauer, Arjun Lakshmipathy, Roberto Shu, and Nancy S Pollard. “Towards very low-cost iterative prototyping for fully printable dexterous soft robotic hands”. In: *2022 IEEE 5th Int. Conference on Soft Robotics (RoboSoft)*. IEEE. 2022, pages 490–497.
- [15] Cristian C Beltran-Hernandez, Damien Petit, Ixchel G Ramirez-Alpizar, and Kensuke Harada. “Variable compliance control for robotic peg-in-hole assembly: A deep-reinforcement-learning approach”. In: *Applied Sciences* 10.19 (2020), page 6923.
- [16] Yoshua Bengio, Aaron Courville, and Pascal Vincent. “Representation learning: A review and new perspectives”. In: *transactions on pattern analysis and machine intelligence* 35.8 (2013), pages 1798–1828.
- [17] Kevin Black et al. “
pi.0 : A Vision – Language – Action Flow Model for General Robot Control”. In: *arXiv preprint arXiv:2410.24164* (2024).
- [18] Jeannette Bohg, Karol Hausman, Bharath Sankaran, Oliver Brock, Danica Kragic, Stefan Schaal, and Gaurav S Sukhatme. “Interactive perception: Leveraging action in perception and perception in action”. In: *IEEE Transactions on Robotics* 33.6 (2017), pages 1273–1291.
- [19] Pasu Boonvisut and M Cenk Çavuşoğlu. “Estimation of soft tissue mechanical parameters from robotic manipulation data”. In: *IEEE/ASME Transactions on Mechatronics* 18.5 (2012), pages 1602–1611.
- [20] Mohamed Bouri and Reymond Clavel. “The linear delta: Developments and applications”. In: *ISR 2010 (41st International Symposium on Robotics) and ROBOTIK 2010 (6th German Conference on Robotics)*. VDE. 2010, pages 1–8.
- [21] Valery Bourny, Thierry Capitaine, Ludovic Barrandon, Claude Pégard, and Aurélien Lorthois. “A localization system based on buried magnets and dead reckoning for mobile robots”. In: *2010 IEEE International Symposium on Industrial Electronics*. IEEE. 2010, pages 373–378.
- [22] Anthony Brohan et al. *RT-1: Robotics Transformer for Real-World Control at Scale*. 2023. arXiv: 2212.06817 [cs.LG].

- [23] Yilin Cai and Shenli Yuan. “In-hand manipulation in power grasp: Design of an adaptive robot hand with active surfaces”. In: *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2023, pages 10296–10302.
- [24] Berk Calli, Arjun Singh, James Bruce, Aaron Walsman, Kurt Konolige, Siddhartha Srinivasa, Pieter Abbeel, and Aaron M Dollar. “Yale-CMU-Berkeley dataset for robotic manipulation research”. In: *The International Journal of Robotics Research* 36.3 (2017), pages 261–268.
- [25] Juan Manuel Camacho and Victor Sosa. “Alternative method to calculate the magnetic field of permanent magnets with azimuthal symmetry”. In: *Revista mexicana de física E* 59.1 (2013), pages 8–17.
- [26] Manuel G Catalano, Giorgio Grioli, Alessandro Serio, Edoardo Farnioli, Cristina Piazza, and Antonio Bicchi. “Adaptive synergies for a humanoid robot hand”. In: *2012 12th IEEE-RAS International Conference on Humanoid Robots (Humanoids 2012)*. IEEE. 2012, pages 7–14.
- [27] Tao Chen, Megha Tippur, Siyang Wu, Vikash Kumar, Edward Adelson, and Pulkit Agrawal. “Visual dexterity: In-hand reorientation of novel and complex object shapes”. In: *Science Robotics* 8.84 (2023), eadc9244. DOI: 10.1126/scirobotics.adc9244. eprint: <https://www.science.org/doi/pdf/10.1126/scirobotics.adc9244>. URL: <https://www.science.org/doi/abs/10.1126/scirobotics.adc9244>.
- [28] Tao Chen, Megha Tippur, Siyang Wu, Vikash Kumar, Edward Adelson, and Pulkit Agrawal. “Visual dexterity: In-hand reorientation of novel and complex object shapes”. In: *Science Robotics* 8.84 (2023), eadc9244. DOI: 10.1126/scirobotics.adc9244.
- [29] Xianyi Cheng, Sarvesh Patil, Zeynep Temel, Oliver Kroemer, and Matthew T Mason. “Enhancing dexterity in robotic manipulation via hierarchical contact exploration”. In: *IEEE Robotics and Automation Letters* 9.1 (2023), pages 390–397.
- [30] Xuxin Cheng, Jialong Li, Shiqi Yang, Ge Yang, and Xiaolong Wang. “Open-television: Teleoperation with immersive active visual feedback”. In: *arXiv preprint arXiv:2407.01512* (2024).
- [31] Nuttapon Chentanez, Matthias Müller, and Miles Macklin. “Real-time simulation of large elasto-plastic deformation with shape matching”. In: *Proceedings of the ACM SIGGRAPH / Eurographics Symposium on Computer Animation*. 2016, pages 159–167.
- [32] Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. “Diffusion Policy: Visuomotor Policy Learning via Action Diffusion”. In: *RSS*. 2023.
- [33] Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. “Diffusion policy: Visuomotor policy learning via action diffusion”. In: *arXiv preprint arXiv:2303.04137* (2023).
- [34] Cheng Chi, Xuguang Sun, Ning Xue, Tong Li, and Chang Liu. “Recent progress in technologies for tactile sensors”. In: *Sensors* 18.4 (2018), page 948.

- [35] Cheng Chi, Zhenjia Xu, Chuer Pan, Eric Cousineau, Benjamin Burchfiel, Siyuan Feng, Russ Tedrake, and Shuran Song. “Universal Manipulation Interface: In-The-Wild Robot Teaching Without In-The-Wild Robots”. In: *arXiv preprint arXiv:2402.10329* (2024). arXiv: 2402.10329 [cs.LG].
- [36] Byung June Choi, Bumsoo Kim, Joo Young Chun, Hyungpil Moon, Ja Choon Koo, Hyouk Ryeol Choi, and Wankyun Chung. “Magnetic patterns based global localization for a mobile robot using hall sensors”. In: *2009 International Conference on Mechatronics and Automation*. IEEE. 2009, pages 192–197.
- [37] Vivian Chu et al. “Using robotic exploratory procedures to learn the meaning of haptic adjectives”. In: *2013 IEEE International Conference on Robotics and Automation*. IEEE. 2013, pages 3048–3055.
- [38] Matei Ciocarlie, Corey Goldfeder, and Peter Allen. “Dexterous grasping via eigengrasps: A low-dimensional approach to a high-complexity problem”. In: *Robotics: Science and systems manipulation workshop-sensing and adapting to the real world*. 2007.
- [39] James J Clark. “A magnetic field based compliance matching sensor for high resolution, high compliance tactile sensing”. In: *Proceedings. 1988 IEEE International Conference on Robotics and Automation*. IEEE. 1988, pages 772–777.
- [40] Samuel Clarke, Travers Rhodes, Christopher G Atkeson, and Oliver Kroemer. “Learning audio feedback for estimating amount and flow of granular material”. In: *Proceedings of Machine Learning Research* 87 (2018).
- [41] Reymond Clavel. “Conception d’un robot parallèle rapide à 4 degrés de liberté”. In: 1991.
- [42] Shadow Robot Company. *Shadow Dexterous Hand*. <https://www.shadowrobot.com/dexterous-hand-series/>. 2023.
- [43] Jorge E Correa, Joseph Toombs, Nicholas Toombs, and Placid M Ferreira. “Laminated micro-machine: Design and fabrication of a flexure-based Delta robot”. In: *Journal of Manufacturing Processes* 24 (2016), pages 370–375.
- [44] Ryan Coulson, Chao Li, Carmel Majidi, and Nancy S Pollard. “The elliott and connolly benchmark: A test for evaluating the in-hand dexterity of robot hands”. In: *2020 IEEE-RAS 20th International Conference on Humanoid Robots (Humanoids)*. IEEE. 2021, pages 238–245.
- [45] Erwin Coumans and Yunfei Bai. “Pybullet, a python module for physics simulation for games, robotics and machine learning”. In: (2016).
- [46] Tamás Czimmermann, Gastone Ciuti, Mario Milazzo, Marcello Chiurazzi, Stefano Roccella, Calogero Maria Oddo, and Paolo Dario. “Visual-Based Defect Detection and Classification Approaches for Industrial Applications—A SURVEY”. In: *Sensors* 20.5 (2020), page 1459.
- [47] Dima Damen et al. “The EPIC-KITCHENS dataset: collection, challenges and baselines”. In: *IEEE Computer Architecture Letters* 01 (2020), pages 1–1.
- [48] Raphael Deimel and Oliver Brock. “A compliant hand based on a novel pneumatic actuator”. In: *2013 IEEE International Conference on Robotics and Automation*. IEEE. 2013, pages 2047–2053.

- [49] Raphael Deimel and Oliver Brock. “A novel type of compliant and underactuated robotic hand for dexterous grasping”. In: *The International Journal of Robotics Research* 35.1-3 (2016), pages 161–185.
- [50] Raphael Deimel, Marcel Radke, and Oliver Brock. “Mass control of pneumatic soft continuum actuators with commodity components”. In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2016, pages 774–779.
- [51] Cosimo Della Santina, Giorgio Grioli, Manuel Catalano, Alberto Brando, and Antonio Bicchi. “Dexterity augmentation on a synergistic hand: The Pisa/IIT SoftHand+”. In: *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*. IEEE. 2015, pages 497–503.
- [52] *Delta Robot One*. <https://projecthub.arduino.cc/deltarobotone/74ee1a02-7c8c-4999-92b5-0371fc37b04f>.
- [53] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. “Imagenet: A large-scale hierarchical image database”. In: *CVPR*. Ieee. 2009, pages 248–255.
- [54] Zackory Erickson, Eliot Xing, Bharat Srirangam, Sonia Chernova, and Charles C Kemp. “Multimodal Material Classification for Robots using Spectroscopy and High Resolution Texture Imaging”. In: *arXiv:2004.01160* (2020).
- [55] Bin Fang, Fuchun Sun, Linyuan Wu, Fukang Liu, Xiangxiang Wang, Haiming Huang, Wenbing Huang, Huaping Liu, and Li Wen. “Multimode grasping soft gripper achieved by layer jamming structure and tendon-driven mechanism”. In: *Soft Robotics* 9.2 (2022), pages 233–249.
- [56] Thomas Feix, Javier Romero, Heinz-Bodo Schmiedmayer, Aaron M. Dollar, and Danica Kragic. “The GRASP Taxonomy of Human Grasp Types”. In: *IEEE Trans. on Human-Machine Systems* 46.1 (2016), pages 66–77. DOI: 10.1109/THMS.2015.2470657.
- [57] Ryan Feng, Youngsun Kim, Gilwoo Lee, Ethan K Gordon, Matt Schmittle, Shivaum Kumar, Tapomayukh Bhattacharjee, and Siddhartha S Srinivasa. “Robot-assisted feeding: Generalizing skewering strategies across food items on a realistic plate”. In: *arXiv preprint arXiv:1906.02350* (2019).
- [58] Ludger Figura and Arthur A Teixeira. *Food physics: physical properties-measurement and applications*. Springer Science & Business Media, 2007.
- [59] Pete Florence et al. “Implicit behavioral cloning”. In: *Conference on Robot Learning*. PMLR. 2022, pages 158–168.
- [60] Letian Fu, Huang Huang, Lars Berscheid, Hui Li, Ken Goldberg, and Sachin Chitta. “Safe self-supervised learning in real of visuo-tactile feedback policies for industrial insertion”. In: *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2023, pages 10380–10386.
- [61] Zipeng Fu, Tony Z Zhao, and Chelsea Finn. “Mobile aloha: Learning bimanual mobile manipulation with low-cost whole-body teleoperation”. In: *arXiv preprint arXiv:2401.02117* (2024).
- [62] Edward P Furlani. *Permanent magnet and electromechanical devices: materials, analysis, and applications*. Academic press, 2001.

- [63] Abhishek Gupta, Clemens Eppner, Sergey Levine, and Pieter Abbeel. “Learning dexterous manipulation for a soft robotic hand from human demonstrations”. In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2016, pages 3786–3793.
- [64] Arjun Gupta, Michelle Zhang, Rishik Sathua, and Saurabh Gupta. “Opening Cabinets and Drawers in the Real World using a Commodity Mobile Manipulator”. In: *arXiv preprint arXiv:2402.17767* (2024).
- [65] Arnav Gupta, Yuemin Mao, Ankit Bhatia, Xianyi Cheng, Jonathan King, Yifan Hou, and Matthew T Mason. “Extrinsic Dexterous Manipulation with a Direct-drive Hand: A Case Study”. In: *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2022, pages 4660–4667.
- [66] Irmak Guzey, Yinlong Dai, Ben Evans, Soumith Chintala, and Lerrel Pinto. “See to touch: Learning tactile dexterity through visual incentives”. In: *arXiv preprint arXiv:2309.12300* (2023).
- [67] Irmak Guzey, Ben Evans, Soumith Chintala, and Lerrel Pinto. “Dexterity from Touch: Self-Supervised Pre-Training of Tactile Representations with Robotic Play”. In: *7th Annual Conference on Robot Learning*. 2023.
- [68] Siddhant Haldar, Jyothish Pari, Anant Rai, and Lerrel Pinto. “Teach a Robot to FISH: Versatile Imitation from One Minute of Demonstrations”. In: *arXiv preprint arXiv:2303.01497* (2023).
- [69] Ankur Handa, Karl Van Wyk, Wei Yang, Jacky Liang, Yu-Wei Chao, Qian Wan, Stan Birchfield, Nathan Ratliff, and Dieter Fox. “DexPilot: Vision-Based Teleoperation of Dexterous Robotic Hand-Arm System”. In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. 2020.
- [70] Ankur Handa, Karl Van Wyk, Wei Yang, Jacky Liang, Yu-Wei Chao, Qian Wan, Stan Birchfield, Nathan Ratliff, and Dieter Fox. “Dexpilot: Vision-based teleoperation of dexterous robotic hand-arm system”. In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2020, pages 9164–9170.
- [71] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. “Mask r-cnn”. In: *Proceedings of the IEEE international conference on computer vision*. 2017, pages 2961–2969.
- [72] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. “Deep residual learning for image recognition”. In: *CVPR*. 2016, pages 770–778.
- [73] Tess Hellebrekers, Nadine Chang, Keene Chin, Michael J Ford, Oliver Kroemer, and Carmel Majidi. “Soft Magnetic Tactile Skin for Continuous Force and Location Estimation Using Neural Networks”. In: *IEEE Robotics and Automation Letters* 5.3 (2020), pages 3892–3898.
- [74] Tess Hellebrekers, Oliver Kroemer, and Carmel Majidi. “Soft Magnetic Skin for Continuous Deformation Sensing”. In: *Advanced Intelligent Systems* 1.4 (2019), page 1900025.

- [75] Tess Hellebrekers, Kevin Zhang, Manuela Veloso, Oliver Kroemer, and Carmel Majidi. “Localization and force-feedback with soft magnetic stickers for precise robot manipulation”. In: *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2020, pages 8867–8874.
- [76] Bianca S Homberg, Robert K Katzschmann, Mehmet R Dogar, and Daniela Rus. “Haptic identification of objects using a modular soft robotic gripper”. In: (2015), pages 1698–1705.
- [77] Kaijen Hsiao, Paul Nangeroni, Manfred Huber, Ashutosh Saxena, and Andrew Y Ng. “Reactive grasping using optical proximity sensors”. In: *2009 IEEE International Conference on Robotics and Automation*. IEEE. 2009, pages 2098–2105.
- [78] Kyle Hsu, Moo Jin Kim, Rafael Rafailov, Jiajun Wu, and Chelsea Finn. “Vision-based manipulators need to also see from their hands”. In: *arXiv preprint arXiv:2203.12677* (2022).
- [79] Josie Hughes, Utku Culha, Fabio Giardina, Fabian Guenther, Andre Rosendo, and Fumiya Iida. “Soft manipulators and grippers: a review”. In: *Frontiers in Robotics and AI* 3 (2016), page 69.
- [80] *IRB 360 FlexPicker*. <https://new.abb.com/products/robotics/robots/delta-robots/irb-360>.
- [81] Phillip Isola, Joseph J. Lim, and Edward H. Adelson. “Discovering States and Transformations in Image Collections”. In: *CVPR*. 2015.
- [82] Aadithya Iyer, Zhuoran Peng, Yinlong Dai, Irmak Guzey, Siddhant Haldar, Soumith Chintala, and Lerrel Pinto. *OPEN TEACH: A Versatile Teleoperation System for Robotic Manipulation*. 2024. arXiv: 2403.07870 [cs.LG].
- [83] Steve C Jacobsen, John E Wood, DF Knutti, and Klaus B Biggers. “The UTAH/MIT dextrous hand: Work in progress”. In: *The Int. Journal of Robotics Research* 3.4 (1984), pages 21–50.
- [84] Divye Jain, Andrew Li, Shivam Singhal, Aravind Rajeswaran, Vikash Kumar, and Emanuel Todorov. “Learning deep visuomotor policies for dexterous hand manipulation”. In: *2019 international conference on robotics and automation (ICRA)*. IEEE. 2019, pages 3636–3643.
- [85] Eric Jang, Alex Irpan, Mohi Khansari, Daniel Kappler, Frederik Ebert, Corey Lynch, Sergey Levine, and Chelsea Finn. “Bc-z: Zero-shot task generalization with robotic imitation learning”. In: *Conference on Robot Learning*. PMLR. 2022, pages 991–1002.
- [86] Biao Jia, Zhe Hu, Jia Pan, and Dinesh Manocha. “Manipulating highly deformable materials using a visual feedback dictionary”. In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2018, pages 239–246.
- [87] Edward Johns. “Coarse-to-fine imitation learning: Robot manipulation from a single demonstration”. In: *2021 IEEE international conference on robotics and automation (ICRA)*. IEEE. 2021, pages 4613–4619.
- [88] Michael I. Jordan and David E. Rumelhart. “Forward models: Supervised learning with a distal teacher”. In: *Cognitive Science* 16.3 (1992), pages 307–354. ISSN: 0364-0213. DOI: [https://doi.org/10.1016/0364-0213\(92\)90036-T](https://doi.org/10.1016/0364-0213(92)90036-T). URL: <https://www.sciencedirect.com/science/article/pii/036402139290036T>.

- [89] Aditya Kannan, Kenneth Shaw, Shikhar Bahl, Pragna Mannam, and Deepak Pathak. “DEFT: Dexterous Fine-Tuning for Hand Policies”. In: *Conference on Robot Learning*. PMLR. 2023, pages 928–942.
- [90] Zhanat Kappassov, Juan-Antonio Corrales, and Véronique Perdereau. “Tactile sensing in dexterous robot hands”. In: *Robotics and Autonomous Systems* 74 (2015), pages 195–220.
- [91] Dov Katz and Oliver Brock. “Manipulating articulated objects with interactive perception”. In: *2008 IEEE International Conference on Robotics and Automation*. IEEE. 2008, pages 272–277.
- [92] Alexander Khazatsky et al. “DROID: A Large-Scale In-The-Wild Robot Manipulation Dataset”. In: (2024).
- [93] Jin-Tae Kim, Hee-Sup Shin, Jae-Young Yoo, Raudel Avila, Yonggang Huang, Yei Hwan Jung, J Edward Colgate, and John A Rogers. “Mechanics of vibrotactile sensors for applications in skin-interfaced haptic systems”. In: *Extreme Mechanics Letters* 58 (2023), page 101940.
- [94] Alexander Kirillov et al. “Segment anything”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2023, pages 4015–4026.
- [95] Oliver Kroemer, Christoph H Lampert, and Jan Peters. “Learning dynamic tactile sensing with robust vision-based training”. In: *IEEE transactions on robotics* 27.3 (2011), pages 545–557.
- [96] Oliver Kroemer, Scott Niekum, and George Konidaris. “A review of robot learning for manipulation: Challenges, representations, and algorithms”. In: *arXiv preprint arXiv:1907.03146* (2019).
- [97] Oliver Kroemer, Scott Niekum, and George Konidaris. “A review of robot learning for manipulation: Challenges, representations, and algorithms”. In: *The Journal of Machine Learning Research* 22.1 (2021), pages 1395–1476.
- [98] Oliver Kroemer and Gaurav Sukhatme. “Meta-level priors for learning manipulation skills with sparse features”. In: *International Symposium on Experimental Robotics*. Springer. 2016, pages 211–222.
- [99] Cecilia Laschi, Barbara Mazzolai, and Matteo Cianchetti. “Soft robotics: Technologies and systems pushing the boundaries of robot abilities”. In: *Science Robotics* 1.1 (2016).
- [100] Christoph Ledermann, Sascha Wirges, David Oertel, Michael Mende, and Heinz Woern. “Tactile Sensor on a Magnetic Basis using novel 3D Hall sensor-First prototypes and results”. In: *2013 IEEE 17th International Conference on Intelligent Engineering Systems (INES)*. IEEE. 2013, pages 55–60.
- [101] Sergey Levine, Chelsea Finn, Trevor Darrell, and Pieter Abbeel. “End-to-end training of deep visuomotor policies”. In: *The Journal of Machine Learning Research* 17.1 (2016), pages 1334–1373.
- [102] Qiang Li, Oliver Kroemer, Zhe Su, Filipe Fernandes Veiga, Mohsen Kaboli, and Helge Joachim Ritter. “A review of tactile information: Perception and action through touch”. In: *IEEE Transactions on Robotics* (2020).

- [103] Rui Li, Robert Platt, Wenzhen Yuan, Andreas Ten Pas, Nathan Roscup, Mandayam A Srinivasan, and Edward Adelson. “Localization and manipulation of small parts using gelsight tactile sensing”. In: *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. 2014, pages 3988–3993.
- [104] Wenzhao Lian, Tim Kelch, Dirk Holz, Adam Norton, and Stefan Schaal. “Benchmarking off-the-shelf solutions to robotic assembly tasks”. In: *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2021, pages 1046–1053.
- [105] Hongzhuo Liang, Shuang Li, Xiaojian Ma, Norman Hendrich, Timo Gerkmann, and Jianwei Zhang. “Making Sense of Audio Vibration for Liquid Height Estimation in Robotic Pouring”. In: *arXiv preprint arXiv:1903.00650* (2019).
- [106] Jacky Liang, Xianyi Cheng, and Oliver Kroemer. “Learning Preconditions of Hybrid Force-Velocity Controllers for Contact-Rich Manipulation”. In: *Conference on Robot Learning*. PMLR. 2023, pages 679–689.
- [107] Hongbin Liu, Xiaojing Song, João Bimbo, Lakmal Seneviratne, and Kaspar Althoefer. “Surface material recognition through haptic exploration using an intelligent contact sensing finger”. In: *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. 2012, pages 52–57.
- [108] Zeyi Liu, Cheng Chi, Eric Cousineau, Naveen Kuppaswamy, Benjamin Burchfiel, and Shuran Song. “Maniway: Learning robot manipulation from in-the-wild audio-visual data”. In: *arXiv preprint arXiv:2406.19464* (2024).
- [109] Beth Logan et al. “Mel Frequency Cepstral Coefficients for Music Modeling.” In: *ISMIR*. Volume 270. 2000, pages 1–11.
- [110] M López, E Castillo, G García, and A Bashir. “Delta robot: Inverse, direct, and intermediate Jacobians”. In: *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science* 220.1 (2006), pages 103–109. DOI: 10.1243/095440606X78263. eprint: <https://doi.org/10.1243/095440606X78263>. URL: <https://doi.org/10.1243/095440606X78263>.
- [111] Raymond Ma and Aaron Dollar. “Yale openhand project: Optimizing open-source hand designs for ease of fabrication and adoption”. In: *IEEE Robotics & Automation Magazine* 24.1 (2017), pages 32–40.
- [112] Raymond R Ma and Aaron M Dollar. “An underactuated hand for efficient finger-gaiting-based dexterous manipulation”. In: *IEEE Int. Conf. on Robotics and Biomimetics*. IEEE. 2014, pages 2214–2219.
- [113] Raymond R Ma, Lael U Odhner, and Aaron M Dollar. “A modular, open-source 3D printed underactuated hand”. In: *2013 IEEE Int. Conf. on Robotics and Automation*. IEEE. 2013, pages 2737–2743.
- [114] Raymond R. Ma and Aaron M. Dollar. “An underactuated hand for efficient finger-gaiting-based dexterous manipulation”. In: *2014 IEEE International Conference on Robotics and Biomimetics (ROBIO 2014)*. 2014, pages 2214–2219. DOI: 10.1109/ROBIO.2014.7090666.

- [115] Arthur W Mahoney and Jake J Abbott. “Control of untethered magnetically actuated tools with localization uncertainty using a rotating permanent magnet”. In: *2012 4th IEEE RAS & EMBS International Conference on Biomedical Robotics and Biomechatronics (BioRob)*. IEEE. 2012, pages 1632–1637.
- [116] Ajay Mandlekar, Jonathan Booher, Max Spero, Albert Tung, Anchit Gupta, Yuke Zhu, Animesh Garg, Silvio Savarese, and Li Fei-Fei. “Scaling Robot Supervision to Hundreds of Hours with RoboTurk: Robotic Manipulation Dataset through Human Reasoning and Dexterity”. In: Nov. 2019, pages 1048–1055. DOI: 10.1109/IROS40897.2019.8968114.
- [117] Ajay Mandlekar et al. “What Matters in Learning from Offline Human Demonstrations for Robot Manipulation”. In: *Conference on Robot Learning*. 2021.
- [118] Ajay Mandlekar et al. “What Matters in Learning from Offline Human Demonstrations for Robot Manipulation”. In: *Conference on Robot Learning*. PMLR. 2022, pages 1678–1690.
- [119] K.K. Maninis, S. Caelles, J. Pont-Tuset, and L. Van Gool. “Deep Extreme Cut: From Extreme Points to Object Segmentation”. In: *CVPR*. 2018.
- [120] Pragna Mannam, Oliver Kroemer, and F Zeynep Temel. “Characterization of compliant parallelogram links for 3D-printed delta manipulators”. In: *Experimental Robotics: The 17th International Symposium*. Springer. 2021, pages 75–84.
- [121] Pragna Mannam, Oliver Kroemer, and F. Zeynep Temel. “Characterization of Compliant Parallelogram Links for 3D-Printed Delta Manipulators”. In: *Proceedings of International Symposium on Experimental Robotics (ISER '20)* (Mar. 2021).
- [122] Pragna Mannam, Avi Rudich, Kevin Zhang, Manuela Veloso, Oliver Kroemer, and Zeynep Temel. “A low-cost compliant gripper using cooperative mini-delta robots for dexterous manipulation”. In: *Rob. Sci. and Systems*. 2021.
- [123] Pragna Mannam, Kenneth Shaw, Dominik Bauer, Jean Oh, Deepak Pathak, and Nancy Pollard. “A Framework for Designing Anthropomorphic Soft Hands through Interaction”. In: *arXiv preprint arXiv:2306.04784* (2023).
- [124] Jan Matas, Stephen James, and Andrew J. Davison. *Sim-to-Real Reinforcement Learning for Deformable Object Manipulation*. 2018. arXiv: 1806.07851 [cs.LG].
- [125] Carolyn Matl, Yashraj Narang, Ruzena Bajcsy, Fabio Ramos, and Dieter Fox. “Inferring the Material Properties of Granular Media for Robotic Tasks”. In: *arXiv:2003.08032* (2020).
- [126] Connor McCann, Vatsal Patel, and Aaron Dollar. “The Stewart Hand: A Highly Dexterous, Six-Degrees-of-Freedom Manipulator Based on the Stewart-Gough Platform”. In: *IEEE Robotics & Automation Magazine* 28.2 (2021), pages 23–36. DOI: 10.1109/MRA.2021.3064750.
- [127] Connor McCann, Vatsal Patel, and Aaron Dollar. “The Stewart hand: A highly dexterous, six-degrees-of-freedom manipulator based on the stewart-gough platform”. In: *IEEE Robotics & Automation Magazine* 28.2 (2021), pages 23–36.
- [128] Connor M McCann and Aaron M Dollar. “Design of a stewart platform-inspired dexterous hand for 6-DOF within-hand manipulation”. In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2017, pages 1158–1163.

- [129] Hayley McClintock, Fatma Zeynep Temel, Neel Doshi, Je-sung Koh, and Robert J Wood. “The milliDelta: A high-bandwidth, high-precision, millimeter-scale Delta robot”. In: *Science Robotics* 3.14 (2018), eaar3018.
- [130] Brian McFee, Colin Raffel, Dawen Liang, Daniel PW Ellis, Matt McVicar, Eric Battenberg, and Oriol Nieto. “librosa: Audio and music signal analysis in python”. In: *Proceedings of the 14th python in science conference*. 2015, pages 18–25.
- [131] Brian McFee et al. *librosa/librosa: 0.10.2.post1*. Version 0.10.2.post1. May 2024. DOI: 10.5281/zenodo.11192913. URL: <https://doi.org/10.5281/zenodo.11192913>.
- [132] B. Mehrafrooz, M. Mohammadi, and M. T. Masouleh. “Kinematic Sensitivity Evaluation of Revolute and Prismatic 3-DOF Delta Robots”. In: *2017 5th RSI International Conference on Robotics and Mechatronics (ICRoM)*. 2017, pages 225–231. DOI: 10.1109/ICRoM.2017.8466159.
- [133] Jean-Pierre Merlet. *Parallel robots*. Volume 128. Springer Science & Business Media, 2005.
- [134] Alireza Mohammadi, Jim Lavranos, Hao Zhou, Rahim Mutlu, Gursel Alici, Ying Tan, Peter Choong, and Denny Oetomo. “A practical 3D-printed soft robotic prosthetic hand with multi-articulating capabilities”. In: *PloS one* 15.5 (2020), e0232766.
- [135] Andrew S Morgan, Kaiyu Hang, Bowen Wen, Kostas Bekris, and Aaron M Dollar. “Complex in-hand manipulation via compliance-enabled finger gaiting and multi-modal planning”. In: *IEEE Robotics and Automation Letters* 7.2 (2022), pages 4821–4828.
- [136] Katharina Mülling, Jens Kober, Oliver Kroemer, and Jan Peters. “Learning to select and generalize striking movements in robot table tennis”. In: *The International Journal of Robotics Research* 32.3 (2013), pages 263–279.
- [137] Yashraj Narang et al. “Factory: Fast contact for robotic assembly”. In: *arXiv preprint arXiv:2205.03532* (2022).
- [138] Abby O’Neill et al. “Open x-embodiment: Robotic learning datasets and rt-x models”. In: *arXiv preprint arXiv:2310.08864* (2023).
- [139] Lael U Odhner et al. “A compliant, underactuated hand for robust manipulation”. In: *The International Journal of Robotics Research* 33.5 (2014), pages 736–752.
- [140] *OptiTrack*. <https://optitrack.com/>.
- [141] Kei Ota, Devesh K Jha, Siddarth Jain, Bill Yezazunis, Radu Corcodel, Yash Shukla, Antonia Bronars, and Diego Romeres. “Autonomous Robotic Assembly: From Part Singulation to Precise Assembly”. In: *arXiv preprint arXiv:2406.05331* (2024).
- [142] Hyeonjun Park and Donghan Kim. “An open-source anthropomorphic robot hand system: HRI hand”. In: *HardwareX* 7 (2020), e00100.
- [143] Sumit Patidar, Adrian Sieler, and Oliver Brock. “In-Hand Cube Reconfiguration: Simplified”. In: *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2023, pages 8751–8756.

- [144] Sarvesh Patil, Samuel C Alvares, Pragna Mannam, Oliver Kroemer, and F Zeynep Temel. “DeltaZ: An Accessible Compliant Delta Robot Manipulator for Research and Education”. In: *2022 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*. IEEE. 2022, pages 13213–13219.
- [145] Sarvesh Patil, Tony Tao, Tess Hellebrekers, Oliver Kroemer, and F Zeynep Temel. “Linear Delta Arrays for Compliant Dexterous Distributed Manipulation”. In: *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2023, pages 10324–10330.
- [146] Tiago Paulino, Pedro Ribeiro, Miguel Neto, Susana Cardoso, Alexander Schmitz, José Santos-Victor, Alexandre Bernardino, and Lorenzo Jamone. “Low-cost 3-axis soft tactile sensors for the human-friendly robot Vizzy”. In: *2017 IEEE international conference on robotics and automation (ICRA)*. IEEE. 2017, pages 966–971.
- [147] Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron Courville. “FiLM: visual reasoning with a general conditioning layer”. In: *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence and Thirtieth Innovative Applications of Artificial Intelligence Conference and Eighth AAAI Symposium on Educational Advances in Artificial Intelligence*. AAAI Press, 2018. ISBN: 978-1-57735-800-8.
- [148] Luis Pérez, Íñigo Rodríguez, Nuria Rodríguez, Rubén Usamentiaga, and Daniel F García. “Robot guidance using machine vision techniques in industrial environments: A comparative review”. In: *Sensors* 16.3 (2016), page 335.
- [149] Karl Pertsch, Kyle Stachowicz, Brian Ichter, Danny Driess, Suraj Nair, Quan Vuong, Oier Mees, Chelsea Finn, and Sergey Levine. “Fast: Efficient action tokenization for vision-language-action models”. In: *arXiv preprint arXiv:2501.09747* (2025).
- [150] Dean A Pomerleau. “Alvin: An autonomous land vehicle in a neural network”. In: *Advances in neural information processing systems* 1 (1988).
- [151] Dean A Pomerleau. “Efficient training of artificial neural networks for autonomous navigation”. In: *Neural computation* 3.1 (1991), pages 88–97.
- [152] Steffen Puhlmann, Jason Harris, and Oliver Brock. “RBO Hand 3: A platform for soft dexterous manipulation”. In: *IEEE Transactions on Robotics* 38.6 (2022), pages 3434–3449.
- [153] Haozhi Qi, Ashish Kumar, Roberto Calandra, Yi Ma, and Jitendra Malik. “In-hand object rotation via rapid motor adaptation”. In: *Conference on Robot Learning*. PMLR. 2023, pages 1722–1732.
- [154] Yuzhe Qin, Hao Su, and Xiaolong Wang. “From one hand to multiple hands: Imitation learning for dexterous manipulation from single-camera teleoperation”. In: *IEEE Robotics and Automation Letters* 7.4 (2022), pages 10873–10881.
- [155] Yuzhe Qin, Wei Yang, Binghao Huang, Karl Van Wyk, Hao Su, Xiaolong Wang, Yu-Wei Chao, and Dieter Fox. “AnyTeleop: A General Vision-Based Dexterous Robot Arm-Hand Teleoperation System”. In: *Proceedings of Robotics: Science and Systems*. Daegu, Republic of Korea, July 2023.
- [156] Joseph Redmon and Ali Farhadi. “Yolov3: An incremental improvement”. In: *arXiv preprint arXiv:1804.02767* (2018).

- [157] Moritz Reuss, Maximilian Li, Xiaogang Jia, and Rudolf Lioutikov. “Goal-conditioned imitation learning using score-based diffusion policies”. In: *arXiv preprint arXiv:2304.02532* (2023).
- [158] Máximo A Roa and Raúl Suárez. “Grasp quality measures: review and performance”. In: *Autonomous robots* 38 (2015), pages 65–88.
- [159] Wonik Robotics. *Allegro Hand*. http://wiki.wonikrobotics.com/AllegroHandWiki/index.php/Allegro_Hand_v4.0. 2023.
- [160] *Robotiq*. <https://thinkbotsolutions.com/collections/robotiq>. Accessed: 2021-04-21.
- [161] Joseph M Romano, Kaijen Hsiao, Günter Niemeyer, Sachin Chitta, and Katherine J Kuchenbecker. “Human-inspired robotic grasp control with tactile sensing”. In: *IEEE Transactions on Robotics* 27.6 (2011), pages 1067–1079.
- [162] Muhammad Hisyam bin Rosle, Ryo Kojima, Zhongkui Wang, and Shinichi Hirai. “Soft fingertip with tactile sensation for detecting grasping orientation of thin object”. In: *2018 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE. 2018, pages 1304–1309.
- [163] Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. “A reduction of imitation learning and structured prediction to no-regret online learning”. In: *Proceedings of the fourteenth international conference on artificial intelligence and statistics. JMLR Workshop and Conference Proceedings*. 2011, pages 627–635.
- [164] Daniela Rus and Michael T Tolley. “Design, fabrication and control of soft robots”. In: *Nature* 521.7553 (2015), pages 467–475.
- [165] Marco Santello, Martha Flanders, and John F Soechting. “Postural hand synergies for tool use”. In: *Journal of neuroscience* 18.23 (1998), pages 10105–10115.
- [166] Amrita Sawhney, Steven Lee, Kevin Zhang, Manuela Veloso, and Oliver Kroemer. “Playing with food: Learning food item representations through interactive exploration”. In: *Experimental Robotics: The 17th International Symposium*. Springer. 2021, pages 309–322.
- [167] Stefan Schaal. “Learning from demonstration”. In: *Advances in neural information processing systems* 9 (1996).
- [168] Stefan Schaal, Jan Peters, Jun Nakanishi, and Auke Ijspeert. “Learning movement primitives”. In: *Robotics research. the eleventh international symposium*. Springer. 2005, pages 561–572.
- [169] Alexander Schmitz, Perla Maiolino, Marco Maggiali, Lorenzo Natale, Giorgio Cannata, and Giorgio Metta. “Methods and technologies for the implementation of large-scale robot tactile sensors”. In: *IEEE Transactions on Robotics* 27.3 (2011), pages 389–400.
- [170] Gerrit Schoettler, Ashvin Nair, Jianlan Luo, Shikhar Bahl, Juan Aparicio Ojea, Eugen Solowjow, and Sergey Levine. “Deep reinforcement learning for industrial insertion tasks with visual inputs and natural rewards”. In: *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2020, pages 5548–5555.

- [171] Florian Schroff, Dmitry Kalenichenko, and James Philbin. “Facenet: A unified embedding for face recognition and clustering”. In: *CVPR*. 2015, pages 815–823.
- [172] *Shadow Dexterous Hand Price List October 2020*. <https://www.shadowrobot.com/wp-content/uploads/SHADOW-DEXTEROUS-HAND-PRICELIST-OCT2020.pdf>. Accessed: 2021-02-28.
- [173] Mohit Sharma, Kevin Zhang, and Oliver Kroemer. “Learning Semantic Embedding Spaces for Slicing Vegetables”. In: *arXiv:1904.00303* (2019).
- [174] Kenneth Shaw, Ananye Agarwal, and Deepak Pathak. “Leap hand: Low-cost, efficient, and anthropomorphic hand for robot learning”. In: *arXiv preprint arXiv:2309.06440* (2023).
- [175] Kenneth Shaw, Shikhar Bahl, and Deepak Pathak. “VideoDex: Learning Dexterity from Internet Videos”. In: *6th Annual Conference on Robot Learning*. 2022.
- [176] Kazuhiro Shimonomura. “Tactile Image Sensors Employing Camera: A Review”. In: *Sensors* 19.18 (2019), page 3933.
- [177] Zilin Si, Kevin Zhang, Oliver Kroemer, and F Zeynep Temel. “DELTAHANDS: A Synergistic Dexterous Hand Framework Based on Delta Robots”. In: *IEEE Robotics and Automation Letters* (2024).
- [178] Zilin Si, Kevin Zhang, Zeynep Temel, and Oliver Kroemer. “Tilde: Teleoperation for Dexterous In-Hand Manipulation Learning with a DeltaHand”. In: *Robotics: Science and Systems*. 2024.
- [179] Jivko Sinapov, Taylor Bergquist, Connor Schenck, Ugonna Ohiri, Shane Griffith, and Alexander Stoytchev. “Interactive object recognition using proprioceptive and auditory feedback”. In: *The International Journal of Robotics Research* 30.10 (2011), pages 1250–1262.
- [180] Jivko Sinapov, Connor Schenck, Kerrick Staley, Vladimir Sukhoy, and Alexander Stoytchev. “Grounding semantic categories in behavioral interactions: Experiments with 100 objects”. In: *Robotics and Autonomous Systems* 62.5 (2014), pages 632–645.
- [181] Aravind Sivakumar, Kenneth Shaw, and Deepak Pathak. “Robotic telekinesis: Learning a robotic hand imitator by watching humans on YouTube”. In: *arXiv preprint arXiv:2202.10448* (2022).
- [182] Shuran Song, Andy Zeng, Johnny Lee, and Thomas Funkhouser. “Grasping in the wild: Learning 6dof closed-loop grasping from low-cost demonstrations”. In: *IEEE Robotics and Automation Letters* 5.3 (2020), pages 4978–4985.
- [183] Gabor Soter, Andrew Conn, Helmut Hauser, and Jonathan Rossiter. “Bodily aware soft robots: integration of proprioceptive and exteroceptive sensors”. In: (2018), pages 2448–2453.
- [184] Oren Spector and Dotan Di Castro. “Insertionnet-a scalable solution for insertion”. In: *IEEE Robotics and Automation Letters* 6.3 (2021), pages 5509–5516.
- [185] Bingjie Tang, Michael A Lin, Iretiayo Akinola, Ankur Handa, Gaurav S Sukhatme, Fabio Ramos, Dieter Fox, and Yashraj Narang. “Industreal: Transferring contact-rich assembly tasks from simulation to reality”. In: *arXiv preprint arXiv:2305.17110* (2023).

- [186] Bingjie Tang et al. “AutoMate: Specialist and Generalist Assembly Policies over Diverse Geometries”. In: *Robotics: Science and Systems*. 2024.
- [187] Gyan Tatiya and Jivko Sinapov. “Deep multi-sensory object category recognition using interactive behavioral exploration”. In: *ICRA*. IEEE. 2019, pages 7872–7878.
- [188] Johan Tegin and Jan Wikander. “Tactile sensing in intelligent robotic manipulation—a review”. In: *Industrial Robot: An International Journal* (2005).
- [189] Trung Duc Than, Gursel Alici, Hao Zhou, and Weihua Li. “A review of localization systems for robotic endoscopic capsules”. In: *IEEE transactions on biomedical engineering* 59.9 (2012), pages 2387–2399.
- [190] Friederike Thonagel. “Vision-Based Teleoperation of the Compliant RBO Hand 3”. Available at <https://www.tu.berlin/en/robotics/teaching/theses/completed-theses/vision-based-teleoperation-of-the-compliant-rbo-hand-3>. Master’s thesis. Berlin, Germany: Technische Universität Berlin, May 2022.
- [191] Yunsheng Tian, Jie Xu, Yichen Li, Jieliang Luo, Shinjiro Sueda, Hui Li, Karl DD Willis, and Wojciech Matusik. “Assemble them all: Physics-based planning for generalizable assembly by disassembly”. In: *ACM Transactions on Graphics (TOG)* 41.6 (2022), pages 1–11.
- [192] Yue Peng Toh, Shan Huang, Joy Lin, Maria Bajzek, Garth Zeglin, and Nancy S Pollard. “Dexterous telemanipulation with a multi-touch interface”. In: *2012 12th IEEE-RAS International Conference on Humanoid Robots (Humanoids 2012)*. IEEE. 2012, pages 270–277.
- [193] Ryan L Truby, Cosimo Della Santina, and Daniela Rus. “Distributed proprioception of 3d configuration in soft, sensorized robots via deep learning”. In: *IEEE Robotics and Automation Letters* 5.2 (2020), pages 3299–3306.
- [194] *Ultimaker: Materials*. <https://ultimaker.com/materials>. Accessed: 2021-02-27.
- [195] *Ultraleap*. <https://www.ultraleap.com/>.
- [196] Homer Rich Walke et al. “BridgeData V2: A Dataset for Robot Learning at Scale”. In: *Proceedings of The 7th Conference on Robot Learning*. 2023.
- [197] Chen Wang, Haochen Shi, Weizhuo Wang, Ruohan Zhang, Li Fei-Fei, and C. Karen Liu. “DexCap: Scalable and Portable Mocap Data Collection System for Dexterous Manipulation”. In: *arXiv preprint arXiv:2403.07788* (2024).
- [198] Zhendong Wang, Jonathan J Hunt, and Mingyuan Zhou. “Diffusion policies as an expressive policy class for offline reinforcement learning”. In: *arXiv preprint arXiv:2208.06193* (2022).
- [199] Zhongkui Wang, Yuuki Torigoe, and Shinichi Hirai. “A prestressed soft gripper: design, modeling, fabrication, and tests for food handling”. In: *IEEE Robotics and Automation Letters* 2.4 (2017), pages 1909–1916.
- [200] Philipp Wu, Yide Shentu, Zhongke Yi, Xingyu Lin, and P. Abbeel. “GELLO: A General, Low-Cost, and Intuitive Teleoperation Framework for Robot Manipulators”. In: *ArXiv abs/2309.13037* (2023). URL: <https://api.semanticscholar.org/CorpusID:262217072>.

- [201] Philipp Wu, Yide Shentu, Zhongke Yi, Xingyu Lin, and Pieter Abbeel. “GELLO: A General, Low-Cost, and Intuitive Teleoperation Framework for Robot Manipulators”. In: *arXiv preprint arXiv:2309.13037* (2023).
- [202] Yilin Wu, Wilson Yan, Thanard Kurutach, Lerrel Pinto, and Pieter Abbeel. *Learning to Manipulate Deformable Objects without Demonstrations*. 2020. arXiv: 1910.13439 [cs.LG].
- [203] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. *Detectron2*. <https://github.com/facebookresearch/detectron2>. 2019.
- [204] Manuel Wüthrich et al. “Trifinger: An open-source robot for learning dexterity”. In: *arXiv preprint arXiv:2008.03596* (2020).
- [205] Junyuan Xie, Ross Girshick, and Ali Farhadi. “Unsupervised deep embedding for clustering analysis”. In: *ICML*. 2016, pages 478–487.
- [206] Haoyu Xiong, Russell Mendonca, Kenneth Shaw, and Deepak Pathak. “Adaptive mobile manipulation for articulated objects in the open world”. In: *arXiv preprint arXiv:2401.14403* (2024).
- [207] Akihiko Yamaguchi and Christopher G Atkeson. “Combining finger vision and optical tactile sensing: Reducing and handling errors while cutting vegetables”. In: *2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids)*. IEEE. 2016, pages 1045–1051.
- [208] Junxia Yan, Haoxian Zheng, Fuchun Sun, Huaping Liu, Yixu Song, and Bin Fang. “C-Shaped Bidirectional Stiffness Joint Design For Anthropomorphic Hand”. In: *IEEE Robotics and Automation Let.* 7.4 (2022), pages 12371–12378.
- [209] Wilson Yan, Ashwin Vangipuram, Pieter Abbeel, and Lerrel Pinto. *Learning Predictive Representations for Deformable Objects Using Contrastive Estimation*. 2020. arXiv: 2003.05436 [cs.LG].
- [210] Won Suk You, Byung June Choi, Bumsoo Kim, Hyungpil Moon, Ja Choon Koo, Wankyun Chung, and Hyouk Ryeol Choi. “Global localization for a small mobile robot using magnetic patterns”. In: *2010 IEEE International Conference on Robotics and Automation*. IEEE. 2010, pages 2618–2623.
- [211] Sarah Young, Dhiraj Gandhi, Shubham Tulsiani, Abhinav Gupta, Pieter Abbeel, and Lerrel Pinto. “Visual imitation made easy”. In: *Conference on Robot Learning*. PMLR. 2021, pages 1992–2005.
- [212] Hanna Yousef, Mehdi Boukallel, and Kaspar Althoefer. “Tactile sensing for dexterous in-hand manipulation in robotics—A review”. In: *Sensors and Actuators A: physical* 167.2 (2011), pages 171–187.
- [213] Shenli Yuan, Lin Shao, Yunhai Feng, Jiatong Sun, Teng Xue, Connor L Yako, Jeannette Bohg, and J Kenneth Salisbury. “Design and Control of Roller Grasper V3 for In-Hand Manipulation”. In: *IEEE Transactions on Robotics* (2024).
- [214] Kevin Zakka. *A PyTorch Implementation of Implicit Behavioral Cloning*. Version 0.0.1. Oct. 2021. URL: <https://github.com/kevinzakka/ibc>.
- [215] Kevin Zhang. https://github.com/firephinx/sounddevice_ros.

- [216] Kevin Zhang, Mohit Sharma, Jacky Liang, and Oliver Kroemer. “A Modular Robotic Arm Control Stack for Research: Franka-Interface and FrankaPy”. In: *arXiv preprint arXiv:2011.02398* (2020).
- [217] Kevin Zhang, Mohit Sharma, Manuela Veloso, and Oliver Kroemer. “Leveraging multi-modal haptic sensory data for robust cutting”. In: *2019 IEEE-RAS 19th International Conference on Humanoid Robots (Humanoids)*. IEEE. 2019, pages 409–416.
- [218] Tianhao Zhang, Zoe McCarthy, Owen Jow, Dennis Lee, Xi Chen, Ken Goldberg, and Pieter Abbeel. “Deep imitation learning for complex manipulation tasks from virtual reality teleoperation”. In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2018, pages 5628–5635.
- [219] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. “Pyramid scene parsing network”. In: *CVPR*. 2017, pages 2881–2890.
- [220] Jialiang Zhao and Edward H Adelson. “Gelsight svelte: A human finger-shaped single-camera tactile robot finger with large sensing coverage and proprioceptive sensing”. In: *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2023, pages 8979–8984.
- [221] Tony Z Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. “Learning fine-grained bi-manual manipulation with low-cost hardware”. In: *arXiv preprint arXiv:2304.13705* (2023).
- [222] Zhong-Qiu Zhao, Peng Zheng, Shou-tao Xu, and Xindong Wu. “Object detection with deep learning: A review”. In: *IEEE transactions on neural networks and learning systems* 30.11 (2019), pages 3212–3232.
- [223] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. “Scene parsing through ade20k dataset”. In: *CVPR*. 2017, pages 633–641.
- [224] Wenxuan Zhou and David Held. “Learning to grasp the ungraspable with emergent extrinsic dexterity”. In: *Conference on Robot Learning*. PMLR. 2023, pages 150–160.
- [225] Henry Zhu, Abhishek Gupta, Aravind Rajeswaran, Sergey Levine, and Vikash Kumar. “Dexterous manipulation with deep reinforcement learning: Efficient, general, and low-cost”. In: *2019 International Conference on Robotics and Automation (ICRA)*. IEEE. 2019, pages 3651–3657.
- [226] Yuke Zhu et al. “Reinforcement and imitation learning for diverse visuomotor skills”. In: *arXiv preprint arXiv:1802.09564* (2018).
- [227] Liang Zou, Chang Ge, Z Jane Wang, Edmond Cretu, and Xiaou Li. “Novel tactile sensor technology and smart tactile sensing systems: A review”. In: *Sensors* 17.11 (2017), page 2653.