Learning via Visual-Tactile Interaction

Helen Jiang

CMU-RI-TR-23-65

August 2023

The Robotics Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

Thesis Committee

Wenzhen Yuan	Carnegie Mellon University (*chair*)
Abhinav Gupta	Carnegie Mellon University
David Held	Carnegie Mellon University
Adithya Murali	Nvidia Research

Thesis submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Robotics

Abstract

Humans learn by interacting with their surroundings using all of their senses. The first of these senses to develop is touch [1], and it is the first way that young humans explore their environment, learn about objects, and tune their cost functions (via pain or treats). Yet, robots are often denied this highly informative and fundamental sensory information, instead relying fully on visual systems. In this thesis, we explore how combining tactile sensing with visual understanding can improve how robots learn from interaction.

We begin by understanding how robots can learn from visual interaction alone in Section 2. We propose the concept of semantic curiosity, which rewards temporal inconsistencies in object detections in a trajectory and is used as an intrinsic motivation reward to train an exploration policy. Our experiments demonstrate that exploration driven by semantic curiosity leads to better object detection performance.

Next, we propose *PoseIt*, a visual and tactile dataset for understanding how holding pose influences the grasp (Section 3). We train a classifier to predict grasp stability from the multi-modal input, and find that it generalizes well to new objects and new poses.

We then focus on more fine-grained object manipulation in Section 4. Thin, malleable objects, such as cables, are particularly susceptible to severe gripper/object occlusions, creating significant challenges in continuously sensing the cable state from vision alone. We propose using visual perception and hand-designed tactile-guided motion primitives to handle cable routing and assembly.

Finally, building on our previous work, we develop a framework that learns USB cable insertion from human demonstrations alone (Section 5). The visual-tactile policy is trained using behavior cloning without requiring any hand-coded primitives. We demonstrate that our transformer-based policy effectively fuses sequential visual and tactile features for high-precision manipulation.

Contents

Chapter 1

Introduction

From the moment a baby is born, she is learning about the world around her. In the first 6 months, babies begin to develop their visual systems, during which they begin to learn object permanence, foreground vs. background, and basic facial and object recognition. Babies start to crawl at 9 months, during which their curiosity helps them learn how to navigate their surroundings effectively and safely. Importantly, this interaction helps fine-tune their visual systems, e.g. their depth perception [2].

Inspired by babies, we start by investigating how semantic visual systems and curiosity can be used to bootstrap navigation and object detection in Section 2. Given a set of environments (and some labeling budget), our goal is to learn an object detector by having an agent select what data to obtain labels for. We explore a self-supervised approach for training our exploration policy by introducing a notion of semantic curiosity. Our semantic curiosity policy is based on a simple observation – the detection outputs should be consistent. Therefore, our semantic curiosity rewards trajectories with *inconsistent* labeling behavior and encourages the exploration policy to explore such areas. The exploration policy trained via semantic curiosity generalizes to novel scenes and helps train an object detector that outperforms baselines trained with other possible alternatives such as random exploration, prediction-error curiosity, and coverage-maximizing exploration.

Perhaps even earlier than babies learning to crawl or walk, they interact with their environment through physical interaction. For example, babies grab their toys, shake them, throw them, etc. While doing this, they learn about object properties and experiment with physical dynamics [3].

Next, we incorporate touch sensing with vision. When humans grasp objects in the real world, we often move our arms to hold the object in a different pose where we can use it. To facilitate the study of how holding poses affect grasp stability, we present *PoseIt*, a novel multi-modal dataset that contains visual and tactile data collected from a full cycle of grasping an object, re-positioning the arm to one of the sampled poses, and shaking the object (Section 3). Using data from *PoseIt*, we can formulate and tackle the task of predicting whether a grasped object is stable

in a particular held pose. Our experimental results show that multi-modal models trained on *PoseIt* achieve higher accuracy than using solely vision or tactile data and that our classifiers can also generalize to unseen objects and poses.

After the first year, most babies develop more adept and fine-grained motor skills. They learn to grasp objects using their thumbs and forefingers as opposed to just pulling the item towards their bellies using their palms. Using this newly developed pincer grasp, they can start to interact with smaller, more fine-grained objects, such as grabbing a spoon by the handle or manipulating drawing implements. [4]

We show how vision and tactile information can be used for fine-grained object manipulation, specifically cable routing and assembly (Section 4). Manipulating cables is challenging for robots because of the infinite degrees of freedom of the cables and frequent occlusion by the gripper and the environment. These challenges are further complicated by the dexterous nature of the operations required for cable routing and assembly, such as weaving and inserting, hampering common solutions with vision-only sensing. We propose to integrate tactile-guided low-level motion control with high-level vision-based task parsing for a challenging task: cable routing and assembly on a reconfigurable task board. Specifically, we build a library of tactile-guided motion primitives using a fingertip GelSight sensor, where each primitive reliably accomplishes an operation such as cable following and weaving. The overall task is inferred via visual perception given a goal configuration image, and then used to generate the primitive sequence. Experiments demonstrate the effectiveness of individual tactile-guided primitives and the integrated end-to-end solution, significantly outperforming the method without tactile sensing.

While babies can begin to imitate simple actions and expressions of their parents at 8 months (e.g. peek-a-boo) [5], they start to mimic adult actions (e.g. playing with toy brooms or hammers) after around a year [6]. Imitating their parents and other adults is an important way for babies to develop new skills and learn about the world.

To mimic how babies learn, we wish to teach robots using human demonstrations rather than rely on hand-designed primitives that are brittle and require task-specific domain knowledge (Section 5). We tackle the task of USB cable insertion by learning visuo-tactile transformer-based policies. This combines the benefits of using vision (useful for coarse-grained localization) and touch (important for force-based feedback) for precise manipulation. Through our experiments, we demonstrate that our learned policy outperforms hand-coded insertion primitives and that fusing visual and tactile modalities outperforms vision alone.

Chapter 2

Semantic Curiosity for Active Visual Learning

2.1 Introduction

Imagine an agent whose goal is to learn how to detect and categorize objects. How should the agent learn this task? In the case of humans (especially babies), learning is quite interactive in nature. We have the knowledge of what we know and what we don't, and we use that knowledge to guide our future experiences/supervision. Compare this to how current algorithms learn – we create datasets of random images from the internet and label them, followed by model learning. The model has no control over what data and what supervision it gets – it is resigned to the static biased dataset of internet images. Why does current learning look quite different from how humans learn? During the early 2000s, as data-driven approaches started to gain acceptance, the computer vision community struggled with comparisons and knowing which approaches work and which don't. As a consequence, the community introduced several benchmarks from BSDS [7] to VOC [8]. However, a negative side effect of these benchmarks was the use of static training and test datasets. While the pioneering works in computer vision focused on active vision and interactive learning, most of the work in the last two decades focuses on static internet vision. But as things start to work on the model side, we believe it is critical to look at the big picture again and return our focus to an embodied and interactive learning setup.

In an embodied interactive learning setup, an agent has to perform actions such that observations generated from these actions can be useful for learning to perform the semantic task. Several core research questions need to be answered: (a) what is the policy of exploration that generates these observations? (b) what should be labeled in these observations - one object, one frame, or the whole trajectory? (c) and finally, how do we get these labels? In this chapter, we focus on the first task – what should the exploration policy be to generate observations which can be useful

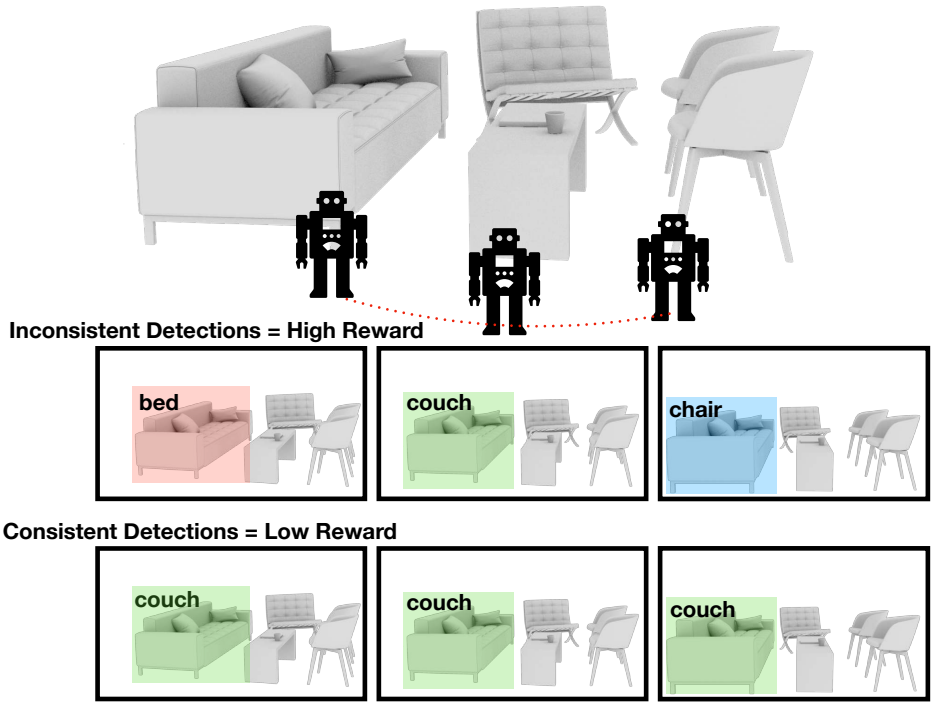


Figure 2.1: **Semantic Curiosity**: We propose semantic curiosity to learn exploration for training object detectors. Our semantically curious policy attempts to take actions such that the object detector will produce inconsistent outputs.

in training an object detector? Instead of using labels, we focus on learning these trajectories in an unsupervised/self-supervised manner. Once the policy has been learned, we use the policy in novel (previously unseen) scenes to perform actions. As observations are generated, we assume that an oracle will densely label all the objects of interest in the trajectories.

So what are the characteristics of a good exploration policy for visual learning, and how do we learn it? A good semantic exploration policy is one which generates observations of objects and not free-space or the wall/ceiling. But not only should the observations be objects, but a good exploration policy should also observe many unique objects. Finally, a good exploration policy will move to parts of the observation space where the current object detection model fails or does not work. Given these characteristics, how should we define a reward function that could be used to learn this exploration policy? Note, as one of the primary requirements, we assume the policy is learned in a self-supervised manner – that is, we do not have ground-truth objects labeled which can help us figure out where the detections work or fail.

Inspired by recent work in intrinsic motivation and curiosity for training poli-

cies without external rewards [9, 10], we propose a new intrinsic reward called semantic curiosity that can be used for the exploration and training of semantic object detectors. In the standard curiosity reward, a policy is rewarded if the predicted future observation does not match the true future observation. The loss is generally formulated in the pixel-based feature space. A corresponding reward function for semantic exploration would be to compare semantic predictions with the current model and then confirm with ground-truth labels – however, this requires external labels (and hence is not self-supervised anymore). Instead, we formulate semantic curiosity based on the meta-supervisory signal of consistency in semantic prediction – that is, if our model truly understands the object, it should predict the same label for the object even as we move around and change viewpoints. Therefore, we exploit consistency in label prediction to reward our policies. Our semantic curiosity rewards trajectories which lead to inconsistent labeling behavior of the same object by the semantic classifier. Our experiments indicate that training an exploration policy via semantic curiosity generalizes to novel scenes and helps train an object detector which outperforms baselines trained with other possible alternatives such as random exploration, pixel curiosity, and free space/map curiosity. We also perform a large set of experiments to understand the behavior of a policy trained with semantic curiosity.

2.2 Related Work

We study the problem of how to sample training data in embodied contexts. This is related to active learning (picking what sample to label), active perception (how to move around to gain more information), intrinsic motivation (picking what parts of the environment to explore). Learning in embodied contexts can also leverage spatio-temporal consistency. We survey these research areas below.

Active Perception. Active perception [11] refers to the problem of actively moving the sensors around at *test time* to improve performance on the task by gaining more information about the environment. This has been instantiated in the context of object detection [12], amodal object detection [13], scene completion [14], and localization [15, 16]. We consider the problem in a different setting and study how to efficiently move around to best *learn* a model. Furthermore, our approach to learn this movement policy is self-supervised and does not rely on end-task rewards, which were used in [14, 15, 13].

Active Learning. Our problem is more related to that of active learning [17], where an agent actively acquires labels for unlabeled data to improve its model at the fastest rate [18, 19, 20]. This has been used in a number of applications such as medical image analysis [21], training object detectors [22, 23], video segmentation [24], and visual question answering [25]. Most works tackle the setting in which the unlabeled data has already been collected. In contrast, we study learning a policy for efficiently acquiring effective unlabeled data, which is complementary

to such active learning efforts.

Intrinsic Rewards. Our work is also related to work on exploration in reinforcement learning [26, 27, 28, 29]. The goal of these works is to effectively explore a Markov Decision Process to find high reward paths. A number of works formulate this as a problem of maximizing an intrinsic reward function which is designed to incentivize the agent to seek previously unseen [30] or poorly understood [9] parts of the environment. This is similar to our work, as we also seek poorly understood parts of the environment. However, we measure this understanding via multi-view consistency in semantics. This is in a departure from existing works that measure it in 2D image space [9], or consistency among multiple models [10]. Furthermore, our focus is not effective exhaustive exploration, but exploration for the purpose of improving semantic models.

Spatio-Temporal smoothing at test time. A number of papers use spatio-temporal consistency at test time for better and more consistent predictions [31, 32]. Much like the distinction from active perception, our focus is using it to generate better data at train time.

Spatio-temporal consistency as training signal. Labels have been propagated in videos to simplify annotations [33], improve prediction performance given limited data [34, 35], as well as collect images [36]. This line of work leverages spatio-temporal consistency to propagate labels for more efficient labeling. Researchers have also used multi-view consistency to learn about 3D shape from weak supervision [37]. We instead leverage spatio-temporal consistency as a cue to identify what the model does not know. [38] is more directly related, but we tackle the problem in an embodied context and study how to navigate to gather the data, rather than analyzing passive datasets for what labels to acquire.

Visual Navigation and Exploration. Prior work on visual navigation can broadly be categorized into two classes based on whether the location of the goal is known or unknown. Navigation scenarios, where the location of the goal is known, include the most common *pointgoal* task where the coordinate to the goal is given [39, 40]. Another example of a task in this category is vision and language navigation [41] where the path to the goal is described in natural language. Tasks in this category do not require exhaustive exploration of the environment as the location of the goal is known explicitly (coordinates) or implicitly (path).

Navigation scenarios, where the location of the goal is not known, include a wide variety of tasks. These include navigating to a fixed set of objects [42, 43, 44, 45, 46, 39], navigating to an object specified by language [47, 48] or by an image [49, 50], and navigating to a set of objects in order to answer a question [51, 52]. Tasks in this second category essentially involve efficiently and exhaustively exploring the environment to search the desired object. However, most of the above approaches overlook the exploration problem by spawning the target a few steps away from the goal and instead focus on other challenges. For example, models for playing FPS games [42, 43, 44, 45] show that end-to-end RL policies are effective at reactive

navigation and short-term planning such as avoiding obstacles and picking positive reward objects as they randomly appear in the environment. Other works show that learned policies are effective at tackling challenges such as perception (in recognizing the visual goal) [49, 50], grounding (in understanding the goal described by language) [47, 48] or reasoning (about visual properties of the target object) [51, 52]. While end-to-end reinforcement learning is shown to be effective in addressing these challenges, they are ineffective at exhaustive exploration and long-term planning in a large environment as the exploration search space increases exponentially as the distance to the goal increases.

Some very recent works explicitly tackle the problem of exploration by training end-to-end RL policies maximizing the explored area [53, 54, 55]. The difference between these approaches and our method is twofold: first, we train semantically-aware exploration policies as compared spatial coverage maximization in some prior works [53, 54], and second, we train our policy in an unsupervised fashion, without requiring any ground truth labels from the simulator as compared to prior works trained using rewards based on ground-truth labels [55].

2.3 Overview

Our goal is to learn an exploration policy such that if we use this policy to generate trajectories in a novel scene (and hence observations) and train the object detector from the trajectory data, it would provide a robust, high-performance detector. In literature, most approaches use on-policy exploration; that is, they use the external reward to train the policy itself. However, training an action policy to sample training data for object recognition requires labeling objects. Specifically, these approaches would use the current detector to predict objects and compare them to the ground-truth; they reward the policy if the predictions do not match the ground-truth (the policy is being rewarded to explore regions where the current object detection model fails). However, training such a policy via semantic supervision and external rewards would have a huge bottleneck of supervision. Given that our RL policies require millions of samples (in our case, we train using 10M samples), using ground-truth supervision is clearly not the way. What we need is an intrinsic motivation reward that can help train a policy which can help sample training data for object detectors.

We propose a semantic curiosity formulation. Our work is inspired by a plethora of efforts in active learning [17] and recent work on intrinsic reward using disagreement [10]. The core idea is simple – a good object detector has not only high mAP performance but is also consistent in predictions. That is, the detector should predict the same label for different views of the same object. We use this meta-signal of consistency to train our action policy by rewarding trajectories that expose inconsistencies in an object detector. We measure inconsistencies by measuring temporal entropy of prediction – that is, if an object is labeled with different classes as the

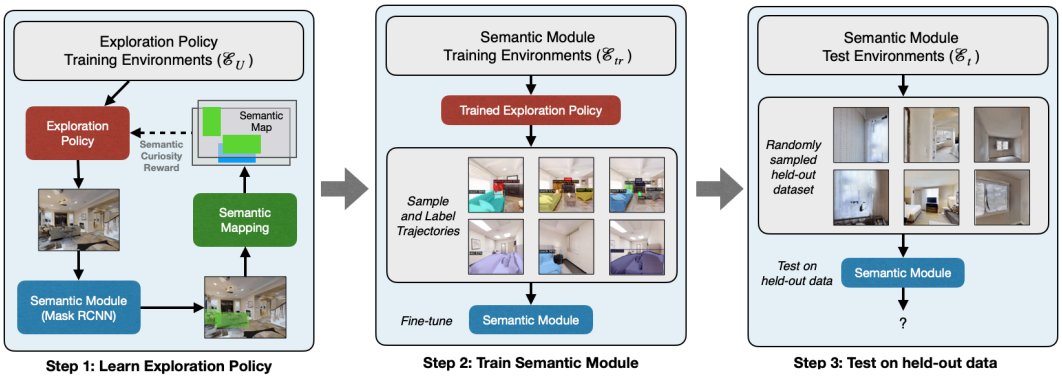


Figure 2.2: **Embodied Active Visual Learning**: We use semantic curiosity to learn an exploration policy on $\mathcal{E}_U$ scenes. The exploration policy is learned by projecting segmentation masks on the top-down view to create semantic maps. The entropy of semantic map defines the inconsistency of the object detection module. The learned exploration policy is then used to generate training data for the object detection/segmentation module. The labeled data is then used to finetune and evaluate the object detection/segmentation.

viewpoint changes, it will have high temporal entropy. The trajectories with high temporal entropy are then labeled via an oracle and used as the data to retrain the object detector (See Figure 2.2).

2.4 Methodology

Consider an agent $\mathcal{A}$ which can perform actions in environments $\mathcal{E}$. The agent has an exploration policy $a_t = \pi(x_t, \theta)$ that predicts the action that the agent must take for current observation x_t. θ represents the parameters of the policy that have to be learned. The agent also has an object detection model $\mathcal{O}$ which takes as input an image (the current observation) and generates a set of bounding boxes along with their categories and confidence scores.

The goal is to learn an exploration policy π, which is used to sample N trajectories $\tau_1 ... \tau_N$ in a set of novel environments (and get them semantically labeled). When used to train an object detector, this labeled data would yield a high-performance object detector. In our setup, we divide the environments into three non-overlapping sets $(\mathcal{E}_U, \mathcal{E}_{tr}, \mathcal{E}_t)$ – the first set is the set of environments where the agent will learn the exploration policy π, the second set is the object detection training environments where we use π to sample trajectories and label them, and the third set is the test environment where we sample random images and test the performance of the object detector on those images.

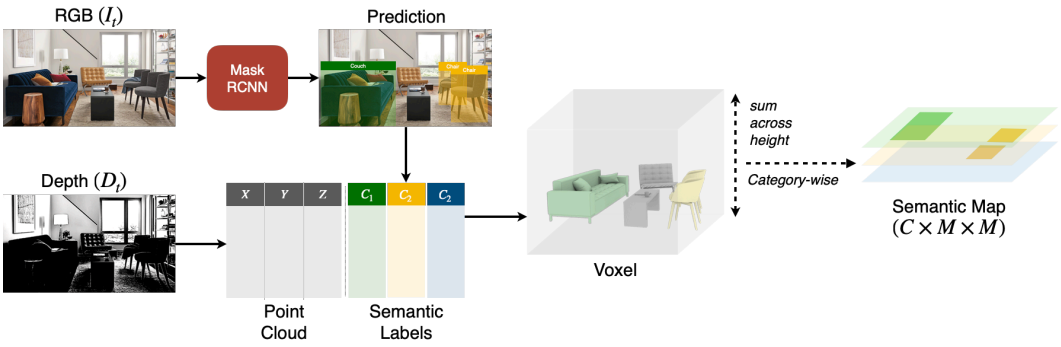


Figure 2.3: **Semantic Mapping.** The Semantic Mapping module takes in a sequence of RGB (I_t) and Depth (D_t) images and produces a top-down Semantic Map.

2.4.1 Semantic Curiosity Policy

We define semantic curiosity as the temporal inconsistency in object detection and segmentation predictions from the current model. We use a Mask RCNN to obtain the object predictions. In order to associate the predictions across frames in a trajectory, we use a semantic mapping module as described below.

Semantic Mapping. The Semantic Mapping module takes in a sequence of RGB (I_t) and Depth (D_t) images and produces a top-down semantic map (M_t^{Sem}) represented by a 3-dimensional tensor $C \times M \times M$, where M is the length of the square top-down map, and C is the number of semantic categories. Each element (c, i, j) in this semantic map is 1 if the Mask RCNN predicted the object category c at coordinates (i, j) on the map in any frame during the whole trajectory and 0 otherwise. Figure 2.3 shows how the semantic map is generated for a single frame. The input RGB frame (I_t) is passed through a current Mask RCNN to obtain object segmentation predictions, while the Depth frame is used to calculate the point cloud. Each point in the point cloud is associated with its semantic labels based on Mask RCNN predictions. Note that these are not ground-truth labels, as each pixel is assigned the category of the highest-confidence Mask RCNN segmentation prediction on the corresponding pixel. The point cloud with the associated semantic labels is projected to a 3-dimensional voxel map using geometric computations. The voxel representation is converted to a top-down map by max-pooling the values across the height. The resulting 2D map is converted to a 3-dimensional Semantic Map, such that each channel represents an object category.

The above gives a first-person egocentric projection of the semantic map at each time-step. The egocentric projections at each time step are used to compute a geocentric map over time using a spatial transformation technique similar to Chaplot et al. [53]. The egocentric projections are converted to the geocentric projections by doing a spatial transformation based on the agent pose. The semantic map at

each time step is computed by pooling the semantic map at the previous timestep with the current geocentric prediction. Please refer to [53] for more details on these transformations.

Semantic Curiosity Reward. The semantic map allows us to associate the object predictions across different frames as the agent is moving. We define the semantic curiosity reward based on the temporal inconsistency of the object predictions. If an object is predicted to have different labels across different frames, multiple channels in the semantic map at the coordinates corresponding to the object will have 1s. Such inconsistencies are beneficial for visual learning in downstream tasks, and hence, favorable for the semantic curiosity policy. Thus, we define the cumulative semantic curiosity reward to be proportional to the sum of all the elements in the semantic map. Consequently, the semantic curiosity reward per step is just the increase in the sum of all elements in the semantic map as compared to the previous time step:

$$r_{SC} = \lambda_{SC} \Sigma_{(c,i,j) \in (C,M,M)} (M_t^{Sem}[c,i,j] - M_{t-1}^{Sem}[c,i,j])$$

where λ_{SC} is the semantic curiosity reward coefficient. Summation over the channels encourages exploring frames with temporally inconsistent predictions. Summation across the coordinates encourages exploring as many objects with temporally inconsistent predictions as possible.

The proposed Semantic Curiosity Policy is trained using reinforcement learning to maximize the cumulative semantic curiosity reward. Note that although the depth image and agent pose are used to compute the semantic reward, we train the policy only on RGB images.

2.5 Experimental Setup

We use the Habitat simulator [56] with three different datasets for our experiments: Gibson [57], Matterport [58] and Replica [59]. While the RGB images used in our experiments are visually realistic as they are based on real-world reconstructions, we note that the agent pose and depth images in the simulator are noise-free unlike the real-world. Prior work has shown that both depth and agent pose can be estimated effectively from RGB images under noisy odometry [53]. In this chapter, we assume access to perfect agent pose and depth images, as these challenges are orthogonal to the focus of this work. Furthermore, these assumptions are only required in the unsupervised pre-training phase for calculating the semantic curiosity reward and not at inference time when our trained semantic-curiosity policy (based only on RGB images) is used to gather exploration trajectories for training the object detector.

In a perfectly interactive learning setup, the current model's uncertainty will be used to sample a trajectory in a new scene, followed by labeling and updating the learned visual model (Mask-RCNN). However, due to the complexity of this online training mechanism, we show results on batch training. We use a pre-trained COCO Mask-RCNN as an initial model and train the exploration policy on that model. Once the exploration policy is trained, we collect trajectories in the training environments and then obtain the labels on these trajectories. The labeled examples are then used to fine-tune the Mask-RCNN detector.

2.5.1 Implementation details

Exploration Policy: We train our semantic curiosity policy on the Gibson dataset and test it on the Matterport and Replica datasets. We train the policy on the set of 72 training scenes in the Gibson dataset specified by Savva et al. [56]. Our policy is trained with reinforcement learning using Proximal Policy Optimization [60]. The policy architecture consists of convolutional layers of a pre-trained ResNet18 visual encoder, followed by two fully connected layers and a GRU layer leading to action distribution as well as value prediction. We use 72 parallel threads (one for each scene) with a time horizon on 100 steps and 36 mini batches per PPO epoch. The curiosity reward coefficient is set to $\lambda_{SC} = 2.5 \times 10^{-3}$. We use an entropy coefficient of 0.001, the value loss coefficient of 0.5. We use Adam optimizer with a learning rate of 1×10^{-5}. The maximum episode length during training is 500 steps.
Fine-tuned Object Detector: We consider 5 classes of objects, chosen because they overlap with the COCO dataset [61] and correspond to objects commonly seen in an indoor scene: chair, bed, toilet, couch, and potted plant. To start, we pre-train a Faster-RCNN model [62] with FPN [63] using ResNet-50 as the backbone on the COCO dataset labeled with these 5 overlapping categories. We then fine-tuned our models on the trajectories collected by the exploration policies for 90000 iterations using a batch size of 12 and a learning rate of 0.001, with annealing by a factor of 0.1 at iterations 60000 and 80000. We use the Detectron2 codebase [64] and set all other hyperparameters to their defaults in this codebase. We compute the AP50 score (i.e., average precision using an IoU threshold of 50) on the validation set every 5000 iterations.

2.5.2 Baselines

We use a range of baselines to gather exploration trajectories and compare them to the proposed Semantic Curiosity policy:

- **Random.** A baseline sampling actions randomly.
- **Prediction Error Curiosity.** This baseline is based on Pathak et al. [9], which trains an RL policy to maximize error in a forward prediction model.

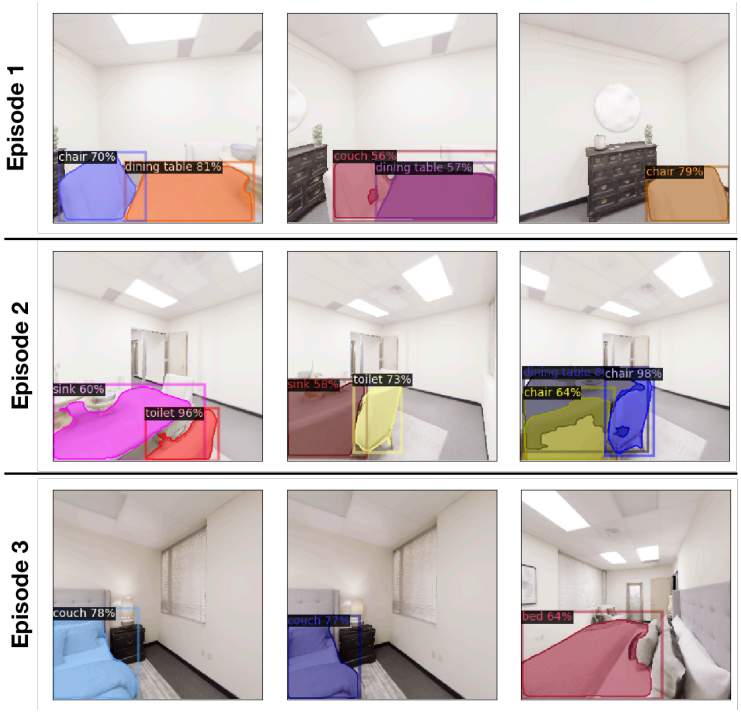


Figure 2.4: **Temporal Inconsistency Examples.** Figure showing example trajectories sampled from the semantic curiosity exploration policy. We highlight the segmentation/detection inconsistencies of Mask RCNN. By obtaining labels for these images, the Mask RCNN pipeline improves the detection performance significantly.

- **Object Exploration.** Object Exploration is a naive baseline where an RL policy is trained to maximize the number of pre-trained Mask R-CNN detections. The limitation of simply maximizing the number of detections is that the policy can learn to look at frames with more objects but might not learn to look at different objects across frames or objects with low confidence.

- **Coverage Exploration.** This baseline is based on Chen et al. [54], where an RL policy is trained to maximize the total explored area.

- **Active Neural SLAM.** This baseline is based on Chaplot et al. [53] and uses a modular and hierarchical system to maximize the total explored area.

After training the proposed policy and the baselines in the Gibson domain, we use them directly (without fine-tuning) in the Matterport and Replica domains. We sample trajectories using each exploration policy, using the images and ground-truth labels to train an object detection model.

Table 2.1: **Analysis.** Comparing the proposed Semantic Curiosity policy with the baselines along different exploration metrics.

Method Name	Semantic Curiosity Reward	Explored Area	Number of Object Detections
Random	1.631	4.794	82.83
Curiosity [9]	2.891	6.781	112.24
Object exploration reward	2.168	6.082	382.27
Coverage Exploration [54]	3.287	10.025	203.73
Active Neural SLAM [53]	3.589	11.527	231.86
Semantic Curiosity	4.378	9.726	291.78

Table 2.2: **Quality of object detection on training trajectories.** We also analyze the training trajectories in terms of how well the pre-trained object detection model works on the trajectories. We want the exploration policy to sample hard data where the pre-trained object detector fails. Data on which the pre-trained model already works well would not be useful for fine-tuning. Thus, lower performance is better.

Method Name	Chair	Bed	Toilet	Couch	Potted Plant	Average
Random	46.7	28.2	46.9	60.3	39.1	44.24
Curiosity [9]	49.4	18.3	1.8	67.7	49.0	37.42
Object Exploration	54.3	24.8	5.7	76.6	49.6	42.2
Coverage Exploration [54]	48.5	23.1	69.2	66.3	48.0	51.02
Active Neural SLAM [53]	51.3	20.5	49.4	59.7	45.6	45.3
Semantic Curiosity	51.6	14.6	14.2	65.2	50.4	39.2

2.6 Analyzing Learned Exploration Behavior

Before we measure the quality of the learned exploration policy for the task of detection/segmentation, we first want to analyze the behavior of the learned policy. This will help characterize the quality of data that is gathered by the exploration policy. We will compare the learned exploration policy against the baselines described above. For all the experiments below, we trained our policy on Gibson scenes and collected statistics in 11 Replica scenes.

Figure 2.4 shows some examples of temporal inconsistencies in trajectories sampled using the semantic curiosity exploration policy. The pre-trained Mask-RCNN detections are also shown in the observation images. Semantic curiosity prefers trajectories with inconsistent detections. In the top row, the chair and couch detector fire on the same object. In the middle row, the chair is misclassified as a toilet and there is inconsistent labeling in the last trajectory. The bed is misclassified as a couch. By selecting these trajectories and obtaining their labels from an oracle, our approach learns to improve the object detection module.

Table 2.1 shows the behavior of all of the policies on three different metrics. The first metric is the semantic curiosity reward itself which measures uncertain detections in the trajectory data. Since our policy is trained for this reward, it gets the highest score on the sampled trajectories. The second metric is the amount of explored area. Both [54] and [53] optimize this metric and hence perform the best (they cover a lot of area but most of these areas will either not have objects or not enough contradictory overlapping detections). The third metric is the number of objects in the trajectories. The object exploration baseline optimizes for this reward and hence performs the best but it does so without exploring diverse areas or uncertain detections/segmentations. If we look at the three metrics together it is clear that our policy has the right tradeoff – it explores a lot of area but still focuses on areas where objects can be detected. Not only does it find a large number of object detections, but our policy also prefers inconsistent object detections and segmentations. In Figure 2.5, we show some examples of trajectories seen by the semantic curiosity exploration along with the semantic map. It shows examples of the same object having different object predictions from different viewpoints and also the representation in the semantic map. In Figure 2.6, we show a qualitative comparison of maps and objects explored by the proposed model and all the baselines. Example trajectories in this figure indicate that the semantic curiosity policy explores more unique objects with higher temporal inconsistencies.

Next, we analyze the trajectories created by different exploration policies during the object detection training stage. Specifically, we want to analyze the kind of data that is sampled by these trajectories. How is the performance of a pre-trained detector on this data? If the pre-trained detector already works well on the sampled trajectories, we would not see much improvement in performance by fine-tuning with this data. In Table 2.2, we show the results of this analysis for these trajectories. As the results indicate, the mAP50 score is low for the data obtained by the semantic curiosity policy.[1] As the pre-trained object detector fails more on the data sampled by semantic curiosity, labeling this data would intuitively improve the detection performance.

2.7 Actively Learned Object Detection

Finally, we evaluate the performance of our semantic curiosity policy for the task of object detection. The semantic curiosity exploration policy is trained on 72 Gibson scenes. The exploration policy is then used to sample data on 50 Matterport scenes. Finally, the learned object detector is tested on 11 Matterport scenes. For each training scene, we sample 5 trajectories of 300 timesteps leading to 75,000 total training images with ground-truth labels. For test scenes, we randomly sample images from test scenes.

[1] Note that curiosity-based policy has the lowest mAP because of outlier toilet category.

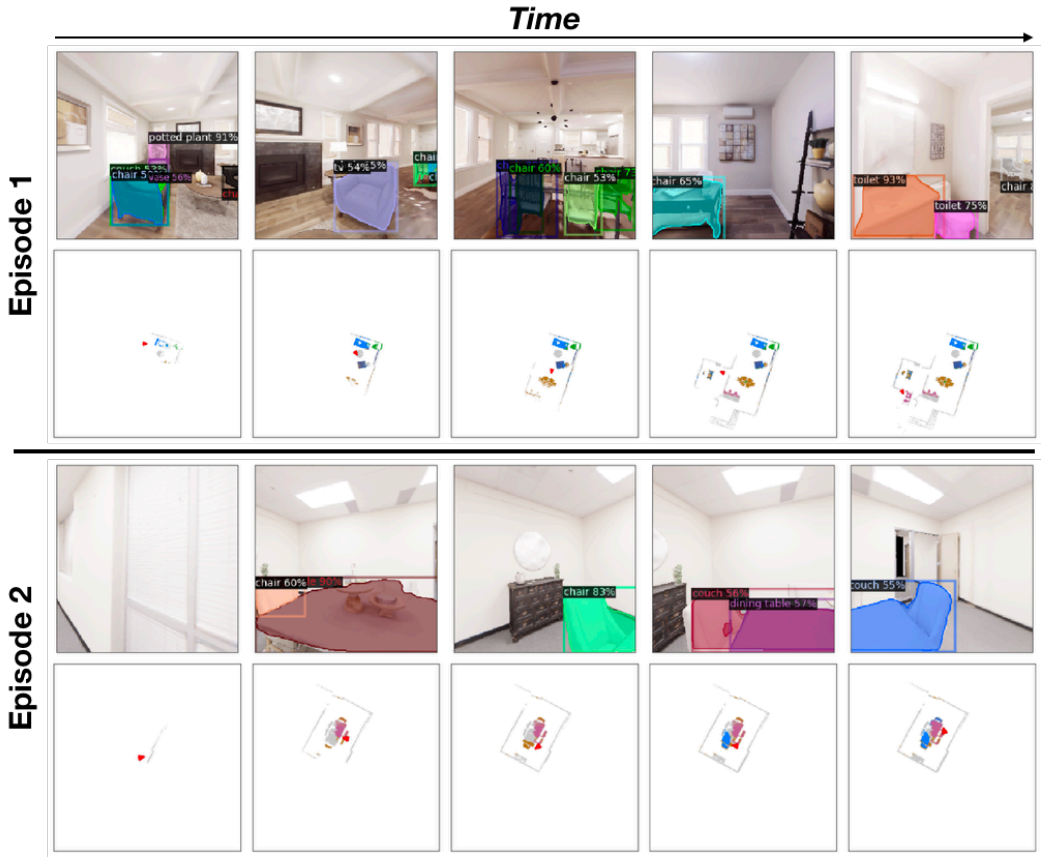


Figure 2.5: **Example trajectories.** Figure showing example trajectories sampled from the semantic curiosity exploration policy. In each episode the top row shows the first-person images seen by the agent and the pre-trained Mask R-CNN predictions. The bottom rows show a visualization of the semantic map where colors denote different object categories. Different colors for the same object indicate that the same object is predicted to have different categories from different view points.

In Table 2.3, we report the top AP50 scores for each method. Our results demonstrate that the proposed semantic curiosity policy obtains higher quality data for performing object detection tasks over alternative methods of exploration. First, we outperform the policy that tries to see maximum coverage area (and hence the most novel images). Second, our approach also outperforms the policy that detects a lot of objects. Finally, apart from outperforming the random policy, visual curiosity [9], and coverage; we also outperform the highly-tuned approach of [53]. The underlying algorithm is tuned on this data and was the winner of the RGB and RGBD challenge in Habitat.

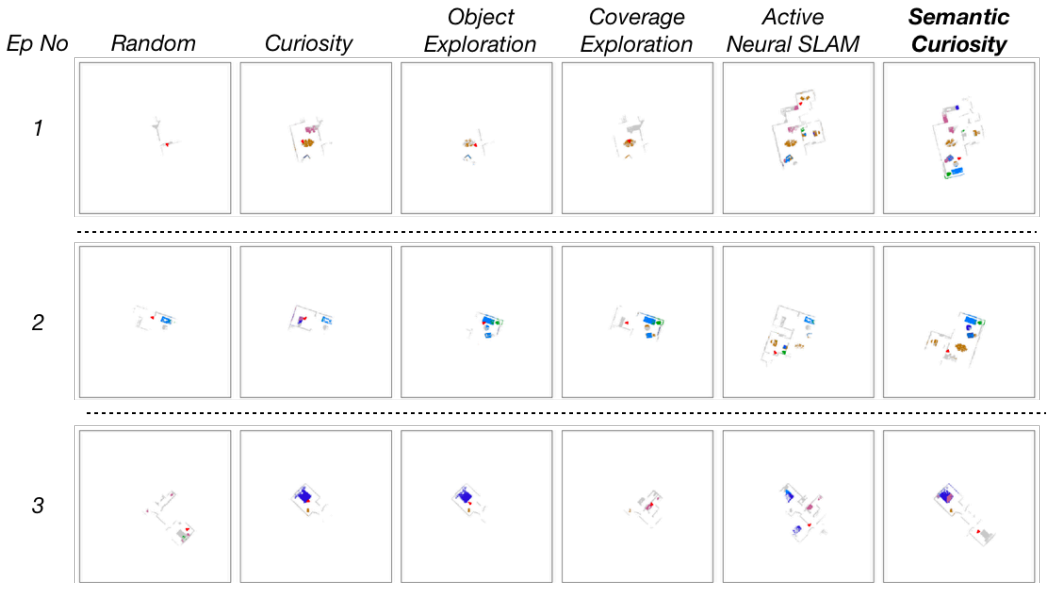


Figure 2.6: **Qualitative Comparison.** Figure showing map and objects explored by the proposed Semantic Curiosity policy and the baselines in 3 example episodes. Semantic Curiosity Policy explores more unique objects with higher temporal inconsistency (denoted by different colors for the same object).

Table 2.3: **Object Detection Results.** Object detection results in the Matterport domain using the proposed Semantic Curiosity policy and the baselines. We report AP50 scores on randomly sampled images in the test scenes. Training on data gathered from the semantic curiosity trajectories results in improved object detection scores.

Method Name	Chair	Bed	Toilet	Couch	Potted Plant	Average
PreTrained	41.8	17.3	34.9	41.6	23.0	31.72
Random	51.7	17.2	43.0	45.1	30.0	37.4
Curiosity [9]	48.4	18.5	42.3	44.3	32.8	37.26
Object Exploration	50.3	16.4	40.0	39.7	29.9	35.26
Coverage Exploration [54]	50.0	19.1	38.1	42.1	33.5	36.56
Active Neural SLAM [53]	53.1	19.5	42.0	44.5	33.4	38.5
Semantic Curiosity	52.3	22.6	42.9	45.7	36.3	**39.96**

2.8 Conclusion and Future Work

We argue that we should go from detection/segmentation driven by static datasets to a more embodied active learning setting. In this setting, an agent can move in the scene and create its own datapoints. An oracle labels these datapoints and helps the

agent learn a better semantic object detector. This setting is closer to how humans learn to detect and recognize objects. In this work, we focus on the exploration policy for sampling images to be labeled. We ask a basic question – how should an agent explore to learn how to detect objects? Should the agent try to cover as many scenes as possible in the hopes of seeing more diverse examples, or should the agent focus on observing as many objects as possible?

We propose semantic curiosity as a reward to train the exploration policy. Semantic curiosity encourages trajectories which will lead to inconsistent detection behavior from an object detector. Our experiments indicate that exploration driven by semantic curiosity shows all of the good characteristics of an exploration policy: uncertain/high entropy detections, attention to objects rather than the entire scene and also high coverage for diverse training data. We also show that an object detector trained on trajectories from a semantic curiosity policy leads to the best performance compared to a plethora of baselines. For future work, this work is just the first step in embodied active visual learning. It assumes perfect odometry, localization and zero trajectory labeling costs. It also assumes that the trajectories will be labeled – a topic of interest would be to sample trajectories with which minimal labels can learn the best detector. Finally, the current approach is demonstrated in simulators - it will be interesting to see whether the performance can transfer to real-world robots.

Chapter 3

PoseIt: A Visual-Tactile Dataset of Holding Poses for Grasp Stability Analysis

3.1 Introduction

Grasping is a core component of many complicated manipulation tasks in robotics. Traditionally, research in grasping focuses on detecting grasping locations [65, 66, 67], and maintaining the grasp stability [68, 69, 70, 71, 72]. These prior works focus on a setting where the gripper holds the object vertically in the robot's hands. Stability is only evaluated in a fixed pose after the robot lifts the object.

This does not translate well in the real world where humans rarely hold the object perfectly still immediately after lifting — for functional purposes, we often need to move the object to a different pose. However, the stability of the grasp can vary significantly with the pose, which is a shortcoming of prior settings which only study the pose immediately after lifting. For example, if a sword is held with its blade pointing vertically to the sky, gravity doesn't create any torque on the sword. If the blade runs parallel to the ground, the torque from gravity could cause it to slip, which is potentially dangerous.

We use "*holding pose*" to describe the pose of the object when it is held in the gripper. Humans have the ability to select a holding pose that is both stable and appropriate for using the object. Humans use the "feeling" from fingers to quickly understand whether the current pose is a good one or if the object is at risk of slipping. The key insight is that the tactile information enables this capability as opposed to using solely the visual signals. We believe robots could work in a similar way, by using both tactile and visual feedback to evaluate holding poses for objects.

In this chapter, we propose a data-based method for predicting the grasp stability of objects in different holding poses. We build a visual-tactile dataset, *PoseIt*, to record the sensory feedback when a robot with a parallel-jaw gripper grasps the

object, moves to a holding pose, and shakes the object. We label whether the grasp was stable during each of these steps and collect data for 26 different objects at 16 different poses. We aim to create a comprehensive dataset with many different modalities of sensory information that the community can refer to when studying holding poses. To collect tactile data, we use a high-resolution GelSight sensor [73] on the fingers. To collect the visual data, we set three cameras to record the grasping point, object geometry, and the overall view of the robot's and object's motion. We also use a force-torque sensor on the robot's wrist.

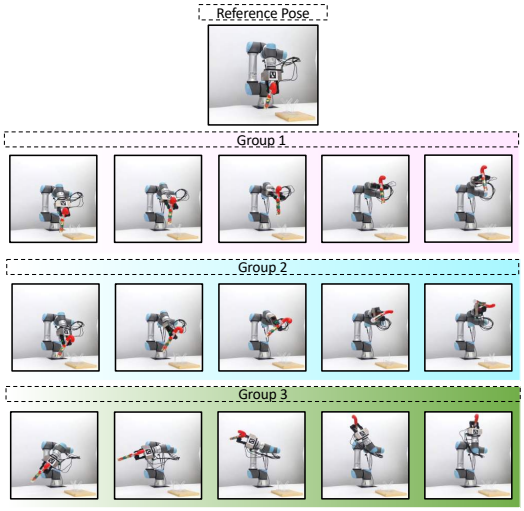


Figure 3.1: **Holding pose sample space.** An example of the 16 sampled poses. Each trial starts at the reference pose, which is the base position where the gripper faces downwards. We create 3 groups of 5 holding poses each, based on the gripper's final orientation.

We use data from *PoseIt* to formulate and tackle the task of predicting grasp stability in a particular holding pose. We use tactile and vision data obtained during grasping and moving the object to the holding pose to predict whether the object is stable. In contrast, most prior work focuses on studying stability in a single pose immediately after the object is lifted.

An important and challenging requirement for solving our task is to correctly classify cases where the object appears stable but will slip if the robot shakes it. Such cases are in the minority ($\approx$ 20% of the full dataset), but important to accurately detect in practical scenarios. To this end, we train an LSTM classifier on visual and tactile data from *PoseIt* using techniques for learning with imbalanced datasets [74].

Our classifier trained on more poses using tactile and vision data achieves 85.2% accuracy on held-out unseen poses, which is 3.4% better than a model which trains on fewer unique poses (Section 3.6.1). This demonstrates that a diverse set of holding poses for each object can improve generalization to unseen poses. We also found

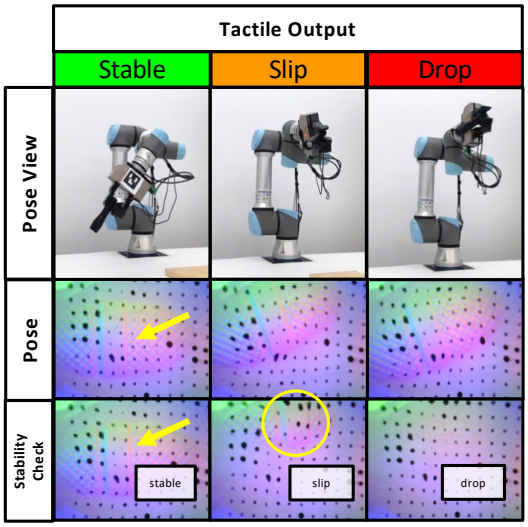


Figure 3.2: The tactile data for a flashlight grasped at the tail using 80N force. For the stable pose in the left column, the grasp is stable where the flashlight is clearly imprinted during the stability check. In the middle column, the flashlight slips during the shaking phase – this is visible where the imprint of the object attenuates gradually and moves toward the top of the sensor surface. Whereas in the pose where the flashlight dropped in the right column, the imprint on the sensor surface disappears during stability check.

that using tactile and vision together is 13.2% better than using vision alone and 3.4% better than using tactile alone, demonstrating the value of collecting multi-modal data.

In summary, we have two primary contributions: 1) we propose *PoseIt*, a novel dataset with multi-modal tactile and visual information and labels for the stability of an object through various stages of grasping, moving to a holding pose, and shaking. 2) we use *PoseIt* to train a classifier which predicts whether the grasp is stable in the current holding pose.

3.2 Related Work

The grasping literature can be roughly categorized into two styles of approach: analytic and data-driven [75, 76]. Analytic approaches rely on known physical models of the object, environments, and grippers to construct the grasps and reason about their quality [77, 78, 79, 75]. However, these approaches could fail if the correct modeling assumptions are unknown or misspecified. Our work is more related to the long line of data-driven approaches, which rely on observations of past trials to build a model or classifier for grasping. Prior works in the data-driven cate-

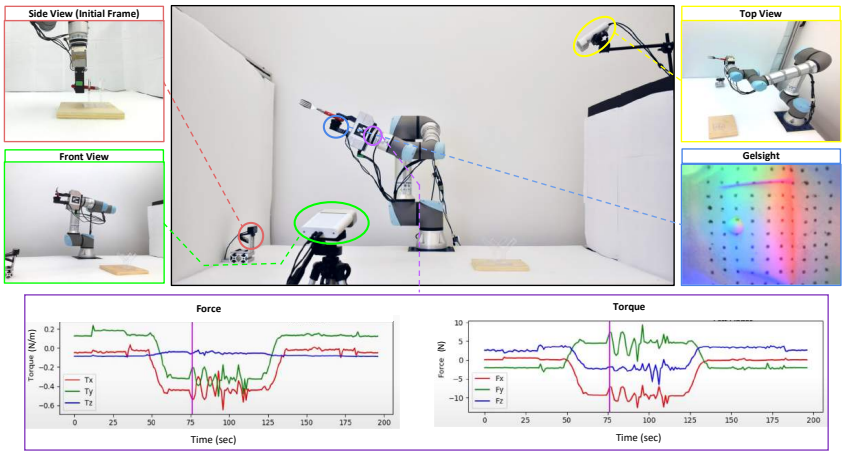


Figure 3.3: **Robot Setup and Data Modalities.** We use this setup to collect *PoseIt*, which consists of multi-modal data on 26 objects, including RGB-D cameras, GelSight tactile sensor, force/torque sensor, and robot trajectory.

gory vary widely in terms of the data modalities and modeling techniques that they use [80, 70, 81, 76, 82, 83, 84, 85].

3.2.1 Tactile sensing for robotics

The sense of touch, one of the main reasons that imparts dexterous and fine manipulation skills to human hands, has been inspiring robotics researchers since early days. A multitude of tactile sensing technologies [86] have been developed to aid robot manipulation. High-resolution tactile sensing technologies such as GelSight [73] and its derivatives [87, 88], which combine deformable elastomer with optical sensors, have allowed robots to more accurately sense richer contact geometries and forces. This has enabled progress in applications such as shape reconstruction [89], object localization [90, 91], and controlling and detecting slip [92, 93, 94].

3.2.2 Predicting grasp stability

Our work is most closely related to papers which use tactile data for data-driven grasp stability prediction [69, 95, 96, 85, 72, 97, 98, 99]. A common theme in these papers is to use tactile data collected from multiple trials of grasping and lifting to predict stability after lifting (and possibly adjust the grasp). [69] use machine learning models such as Support Vector Machines (SVMs) and Hidden Markov Models (HMMs) to predict slip based on data from a Weiss robotics tactile sensor. [96] predicts slip using a neural network trained on visual and tactile data from the GelSight sensor, and in follow-up work [72], combine slip prediction with an

action-conditional model to learn grasping and re-grasping sequences in an end-to-end manner. [85] and [98] also train neural networks using visual and GelSight data for the task of slip detection as the object is lifted. All of these prior works consider the stability of the object in the pose obtained immediately after grasping. In contrast, the primary contribution of our work is to collect visual and tactile data for grasping and moving the object to a diverse set of holding poses, as objects can behave differently in these holding poses than the pose obtained after lifting. This allows us to tackle the (to the best of our knowledge, novel) task of predicting grasp stability in a particular holding pose.

3.2.3 Grasping datasets

Large-scale data is a crucial driving force behind many of the recent advancements in grasping. For example, [83] collect a dataset of 50000 grasp trials, which was 40x larger than prior work, and show that this dataset can be used to predict grasp locations from image patches (they do not use tactile sensing). [100] collect a dataset of 1000 grasp trials using BioTac sensors [101] and perform a shaking-based stability check to determine whether a grasp is stable. The datasets of [96, 85], which contain GelSight and visual data, are also publicly available, with 9269 and 1102 grasp trials, respectively. [102] collect a dataset of 7800 grasp interactions involving localizing the object, grasping, and regrasping and train an iterative regrasping policy based on tactile feedback. [103] collect a dataset of 2550 grasp trials with the Eagle Shoal robotic hand. While there are now many choices of publicly available tactile datasets for initial grasp stability, prior to our work, no dataset existed for evaluating grasp stability in various holding poses.

3.3 Collecting the *PoseIt* dataset

In this section, we describe the collection process for the *PoseIt* dataset. Two main components of *PoseIt* differ from prior works: 1) the holding pose sample space, a set of 16 distinct holding poses to mimic typical human-like holding poses 2) the multi-phase collection cycle, which moves an object to a particular pose and performs a shaking stability check at that pose.

3.3.1 Holding pose sample space

The stability of a particular object in the gripper depends on the holding pose. To attempt to model how humans typically hold objects for use (rather than simply lifting them up vertically), we design a diverse set of 16 holding poses.

We set the first pose as a reference pose, as it is the typical base position where the gripper faces downwards. Inspired by human arm movements observed during manipulating routine objects, the other 15 holding poses are categorized into

3 possible groups of 5 poses, each based on the final orientation of the gripper. We expect poses within the same group to behave more similarly. We obtain the poses within a group by rotating the end-effector in fixed-degree increments. Examples of the 16 holding poses are shown in Figure 3.1. Group 1 poses take advantage of gravity to stabilize potential rotational movements in the case of objects with varying mass distributions. Whereas, poses in group 2 gradually rotate the gripper to counterbalance the gravitational force to avoid slippage for objects with non-uniformly distributed mass. Group 3 poses aid in analyzing the stability of the overhead (above the shoulder joint) arm movements by incrementally increasing the object's distance from the ground.

Figure 3.4: **Dataset Objects.** To test grasp quality on a diverse range of objects, we collected data for 26 various household objects with a diverse range of size, shape, material, mass, and texture. For each object, we grasp it at multiple locations while it is resting in most likely orientations. We also grasp each object with 2 different gripper forces, the minimum force required to lift the object from (5N, 15N, or 40N), and 80N. We collect data for all possible combinations of grasp point, force value, and target holding pose, resulting in (Number of grasp points $\times 2 \times 16$) choices per object.

3.3.2 Data collection pipeline

At the start of every data collection cycle, the object is placed at the center of the table in varying orientations. It is then grasped and moved through 4 phases in succession:

1. Grasp: The robot grasps and lifts the object. Data collection begins 1 second before grasping.

2. Pose: After lifting the object, the robot arm moves to one of 16 possible holding poses. The robot waits for 1 second after reaching the pose.

3. Shaking/stability check: This phase shakes the object in-hand, inspired by prior work who also performs in-hand shaking [100, 94]. We apply a quick

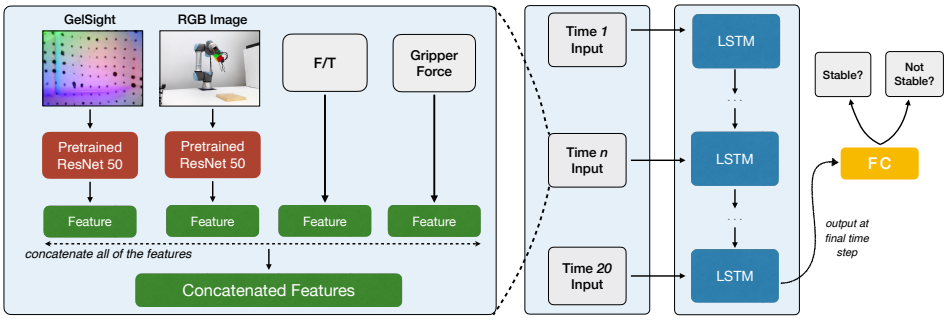


Figure 3.5: **Model architecture for grasp stability prediction.** Our model takes as input 4 modalities of data: tactile, visual, F/T sensor data, and the gripper force value (which is fixed across all timesteps). We use the sequence of concatenated features from these modalities as input to an LSTM, which predicts "stable" or "not stable".

rotational movement along the gripper axis, followed by rigorous arm shaking movements along all 3 axes. This phase tests whether the object is stable in the holding pose.

4. Retract and release: During the retract stage, the robot arm returns to the beginning position, and the gripper releases the object at its starting location.

Data annotation. We manually label the stability of each of the 4 phases with one of 4 categories:

- Pass: Firm grasp. The object doesn't move relative to the gripper.
- Slip: Object is in still contact with the gripper, but some rotational or translational slip is manually observed.
- Drop: Object falls off the gripper.
- Not present: Object dropped before the current phase.

3.4 Analysis of dataset statistics

In total, *PoseIt* consists of datapoints from 26 diverse objects across 16 different poses.

We show the phase-wise data division in Table 3.1. During the grasp phase, we manually collect the roughly same number of stable (*Pass*) and unstable (*Slip+Drop*) initial grasp cases to form a balanced dataset. We observe that in the pose phase, the number of *Slip+Drop* category samples decreases by 14% (compared to the grasp phase). This shows that post-grasp arm re-positioning can help stabilize the object.

In the shaking phase, there is a 4% increase in slip and drop cases (compared to the pose phase). This conveys that the shaking phase is important because even seemingly stable poses may not withstand external disturbances.

Stage	Label			Slip+Drop
	Pass	Slip	Drop	
Grasp	1037	778	25	**43.64%**
Pose	1278	192	345	**29.18%**
Shake	1003	255	365	**33.69%**

Table 3.1: **Dataset statistics.** We collected 1840 data points on 26 objects with stability labels on different stages. The last column shows the percentage of unstable cases.

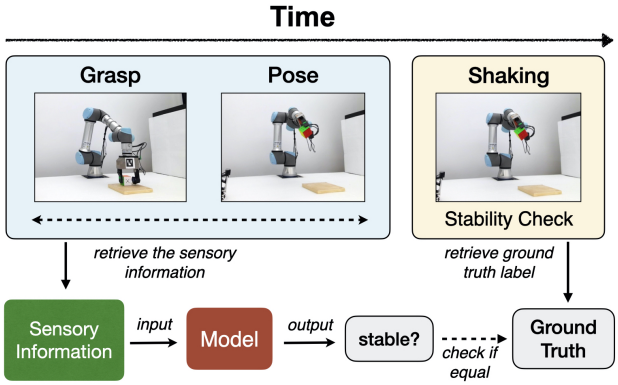


Figure 3.6: **Task definition.** Our task aims to predict the stability of the grasp in the holding pose. Given sensory information in the grasp and pose stage, we aim to predict whether the object will be stable during the shaking phase.

3.5 Predicting stability in the holding pose

Using the data collected from *PoseIt*, we predict the grasp stability when the object is in the holding pose. More formally, we formulate our task as follows:

Given sensory data from the grasp and pose phase, predict whether the object is stable during the shaking phase.

Our task is illustrated in Figure 3.6. The motivation is that understanding and predicting the stability of objects in holding poses is important for manipulating

objects in practical settings. We note that our task is a prediction, and not a detection task because the inputs are sensory information before the shaking occurs.

For simplicity, we combine the "slip" and "drop" labels so the task is to predict "stable" or "not stable". From now on, "label" denotes the binary label unless specified otherwise.

3.5.1 Model architecture

As in [85], our primary classification model is a Long Short-Term Memory (LSTM) which aggregates multi-modal sensor data over a sequence of time steps to obtain a prediction. Figure 5.2 illustrates the model. For a single example, the input is tactile, RGB, and force/torque sensor readings gathered during a sequence of timesteps. The gripping force is also an input, but it is fixed across all timesteps.

To obtain features for the tactile data, we subtract the pre-contact tactile image from all frames in the sequence and feed the resulting frames through a pre-trained ResNet50 [104] ImageNet backbone. We obtain features for the RGB data using the same pre-trained classifier. We hold the pre-trained classifier throughout training. For a given timestep, we concatenate the tactile features, RGB features, and raw force/torque values for this timestep into a single feature vector. We additionally concatenate the gripper force (a fixed value through all the timesteps) to this vector. The sequence of feature vectors for all timesteps is fed into a 2-layer bidirectional LSTM. The prediction is computed as a linear function of the LSTM hidden layers corresponding to the last timestep in the sequence.

Data features and pre-processing: We report performance on three possible combinations of the data features as input:

1. Vision + F/T + Force (V)
2. Tactile + F/T + Force (T)
3. Vision + Tactile + F/T + Force (V + T)

We avoid using datapoints where the object was dropped during the grasp or pose phase, as in such cases the object would not be present during the shaking phase. Unless otherwise specified, we use 20 timesteps spaced evenly from the start of the grasp to the end of the pose phase. We standardize each coordinate of the input features to have a mean of 0 and a standard deviation of 1 over all examples and timesteps.

3.5.2 Improved classification using Deferred-Resampling

It is possible that the labels during the pose phase and the shaking phase can differ, because an object can be stable when moving to the pose but unstable when shaking in the holding pose (or vice-versa). Examples, where these labels differ are particularly challenging because accurate classification requires predicting a different label

than the one currently reflected in the sensor readings. This difficulty is also compounded by the fact only ≈20% of examples have different labels during the pose and shaking phases.

We propose to improve performance on such examples by adapting techniques used for learning with dataset imbalance which under-sample the majority class [105, 106, 74]. For our setting, the source of imbalance is not from the label distribution, but rather, whether an example has the same label in the pose and shaking phases. Overall, the Deferred-Resampling method [74] in this section improves test accuracy on unseen objects by 2.63% over the vanilla LSTM baseline.

3.6 Experiments and Discussion

Stability classification requires understanding the dynamics at different holding poses, thus making it a good probing task as the first step to modeling how humans grasp objects in the real world. We train models to use sensor readings from the grasp and pose phase to predict stability during the shaking phase and evaluate accuracy on held-out test sets. We find that training an LSTM with DRS on Tactile + Vision + F/T + Force value data performs best, achieving 85.21% accuracy when train and test come from the same distribution, showing the usefulness of the multi-modal nature of *PoseIt*. (See Table 3.2.)

The contributions of this section are:

1. We show that although the classifier can generalize to new holding poses, test accuracy suffers when the test and train poses are very different. This demonstrates the value of data from diverse holding poses.

2. We show that our models generalize to new poses and objects. This points to the potential for pre-training grasp stability models using *PoseIt*, which could be finetuned and deployed for real-world robotic grasping tasks.

Classifiers: The subsequent sections will frequently refer to the following classifiers and algorithms:

LSTM: This is the vanilla architecture described in Section 3.5.1.

LSTM+DRS: We train the LSTM described in Section 3.5.1 using the Deferred-Resampling (DRS) technique (Section 3.5.2). DRS ensures that the model places more emphasis on learning from examples where the label changed between pose and shaking phases.

3.6.1 Generalization to unseen poses

In this section, we demonstrate the value of having data from diverse holding poses. We test the generalization of the classifier to unseen poses and verify that generalization performance is improved when similar poses are in the train and test sets.

Features	Pose Group	Random Poses (5 test)	Uniform Random Split
Tactile	81.55%	83.85%	84.52%
Vision + Tactile	82.4%	84.28%	**85.21%**

Table 3.2: **Generalization to unseen poses:** We display test accuracy results for 3 different methods of splitting holding poses into train and test. The model obtains 85.21% accuracy when the same poses are in train and test. It performs worst when tested on poses than it was not trained on, as the train and test distributions are least similar in this setting. This demonstrates the value of collecting data for a large and diverse set of holding poses for each object.

To conduct this experiment, we split the dataset into train and test in 3 different ways:

Uniform random split: We randomly partition the data into train and test. Thus, the train and test sets are drawn from the same distribution, with no difference between object or pose. To keep the training set size consistent with the dataset splits below, we put 68.75% of the data cycles in the training set. To reduce variance, we obtain 15 different splits of the dataset and report the average test accuracy of classifiers trained on these 15 splits of the dataset.

Random 5 poses: We randomly select 5 of the 16 poses to put in the test set and partition data corresponding to all other poses into the training set. To reduce variance, we split the dataset this way 15 different times and report average test accuracy over classifiers trained on each split of the dataset.

Pose group: Recall that poses 2-16 consist of 3 groups of 5 similar poses each. We put 1 group in the test set and the other 2 groups in the train set. We report average test results over all choices of the training set, where for each choice we train 5 classifiers (with different random seeds) to reduce variance.

Results: We report results for LSTM+DRS in Table 3.2. When we split train and test by the pose group, the classifier performs the worst on the test set, as the poses used in training and testing are the least similar. The classifier trained on the uniform random split of the data performs best, achieving 85.21% accuracy. This demonstrates clear value in having a dataset with diverse held poses for each object.

3.6.2 Generalization to unseen objects

In this section, we analyze the ability of our classifiers trained on tactile data to generalize to unseen objects. We provide a baseline that uses a proxy label of whether the object slips during the *pose* phase. This baseline performs poorly, showing that checking stability after the object moves to the holding pose is necessary for our task.

Classifier	Vision	Tactile	Both
LSTM-WC (Ceiling)	*67.31*	*77.89*	*80.8*
Majority Classifier (Baseline)	63.21	63.21	63.21
LSTM-P (Baseline)	64.47	66.57	66.43
Linear (Baseline)	63.31	64.22	66.57
LSTM (Ours)	66.2	73.45	74.66
LSTM+DRS (Ours)	**68.25**	**74.76**	**77.29**

Table 3.3: **Generalization to new objects with tactile data:** We show the test accuracy of our classifiers on three different combinations of modalities. Our described model to predict stability in unseen objects (LSTM+DRS) outperforms baselines. Using all of the modalities (Vision+Tactile+F/T+Force) produces the highest accuracy. For reference, we also include LSTM-WC, a ceiling that uses sensory data from the entire data collection cycle.

As our dataset consists of data gathered from 26 objects, we use 22 objects in the training set and test generalization to the unseen test set of 4 objects. To reduce the variance over the choice of objects in the train and test set, we perform experiments on 20 randomly selected combinations of 4 objects for the test set. We report average numbers over the 20 ways of splitting train and test and 5 different random seeds for training the classifier. We compare our approaches against the following baseline/ceiling classifiers:

Majority classifier: This trivial classifier labels all examples as either stable or not stable, depending on the majority.

LSTM pose label (LSTM-P): This baseline LSTM is trained on the stability label for the *pose* phase.

LSTM whole cycle (LSTM-WC): This classifier is trained on 20 evenly spaced sensor readings from the start of grasping to releasing. We expect this classifier to outperform others because it has access to data from the shaking phase, so it only needs to detect, rather than predict grasp failure.

Results: We show our results in Table 3.3. Our LSTM+DRS classifier produces the highest accuracy across all different variations of data input, besides the LSTM-WC whole cycle ceiling (which we expect to perform best, given it has access to data across all timesteps). Additionally, using all four modalities (RGB, Tactile, F/T, and Force) together outperforms using a subset of those features.

We also note that the pose label baseline only performs 3% better than the majority classifier, and performs much worse than the DRS classifier. This demonstrates the importance of the labeled data from the shaking phase, as simply knowing the label from the pose phase is *not* sufficient.

3.7 Conclusion and Future Work

We propose *PoseIt*, a dataset that contains visual and tactile data from grasping objects and moving them to a holding pose for a stability check. This data is the next step in modeling how humans interact with objects in the real world: humans want to hold objects at various stable poses, rather than just lifting them vertically. Our experiments show that *PoseIt* provides unique data that allows us to predict grasp stability at specific holding poses.

For future work, we would like to take even further advantage of *PoseIt* to tackle the question of re-positioning an object to more stable holding poses. We would like to directly predict which poses are stable, using only sensor readings taken from a single reference pose.

Chapter 4

Cable Routing and Assembly using Tactile-driven Motion Primitives

4.1 Introduction

Robotic manipulation and planning has seen substantial progress in recent years. However, manipulating flexible objects such as cables and wires remains an open problem [107, 108, 109, 110]. In particular, tasks such as cable assembly require robot systems to be able to effectively track narrow cables and route them in between obstacles. Because the object of interest is thin, malleable, and subject to severe occlusions, this often requires advanced sensing, planning, and control.

While previous efforts in cord manipulation have considered knot tying [107], wire insertion [108], and (most relevant to us) cable routing [109], they are limited in task complexity and generalizability. Specifically, simplistic operations such as picking, moving, and placing prevent the robot from handling more complex tasks such as weaving parts through slots [111]. In addition, the above mentioned methods often cannot recover from failure due to the open-loop nature of the execution [109]. The main difficulty lies in how to continuously estimate the cable state, particularly when vision alone suffers from severe gripper/object occlusions, and how to generate motion commands accordingly.

Tactile-guided manipulation is a promising approach, but has only been demonstrated in a simple cable following task [112]. In this chapter, we consider a full task of cable routing and assembly inspired by the NIST Assembly Task Board 3 [111]. We design a reconfigurable setup with fixtures that require 4 operations covering common tethered object manipulation, as illustrated in Fig. 5.1. These tasks have longer horizons, and require reliable generalization across configurations.

To solve this problem, we propose to use visual perception for task understanding, i.e., a task description file is automatically extracted from a goal configuration

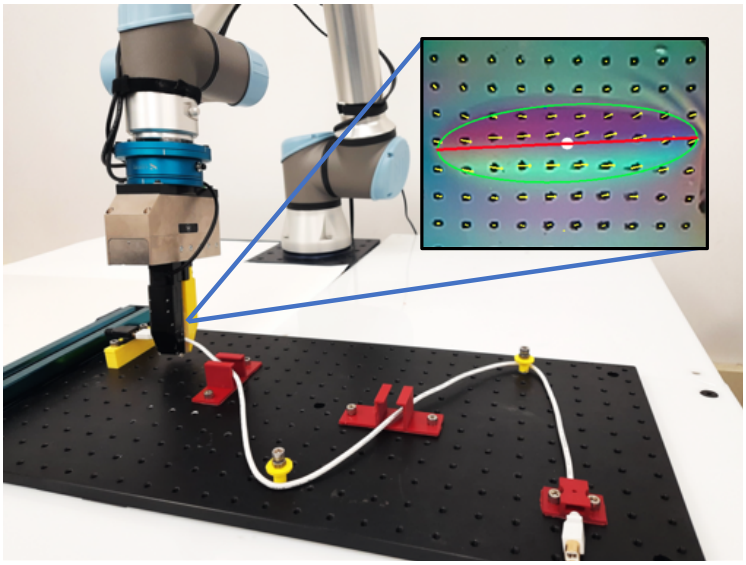


Figure 4.1: **Goal-conditioned cable manipulation using visual-tactile perception.** Given an RGBD image of a goal configuration, our system parses the task, generates reference trajectories, and applies a sequence of tactile-guided motion primitives to accomplish a cable routing and assembly task.

image. The parsed task file is then mapped to a sequence of tactile-guided motion primitives. Notably, we leverage tactile sensing to design a library of primitives to continuously estimate the cable state and manipulate it in a closed-loop manner. Concretely, four primitives (cable following, pivoting, weaving, and insertion) are defined, each as a state machine, where the state transitions are governed by tactile perception. State-dependent controllers are then activated sequentially to realize the desired primitive behavior. Because of this modular design where a task is parsed via visual perception into primitives and each tactile-guided primitive is task context independent, our approach is able to generalize across different task board configurations with zero adaptation effort. Experiment results demonstrate the effectiveness and generalizability of the tactile-guided motion primitives, as well as our fully integrated pipeline.

Our contributions are summarized as follows. First, we propose a novel integrated solution for cable routing and assembly, where visual perception enables automatic high-level task parsing. Second, we design a library of tactile-guided motion primitives for low-level motion control to accomplish complex cable operations. Third, we provide baselines with and without tactile sensing on a reconfigurable cable routing and assembly task for future research.

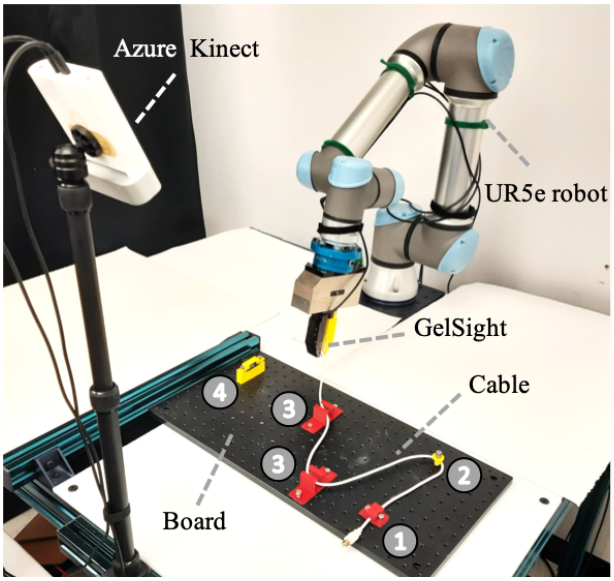


Figure 4.2: **System setup and task board configuration.** The task board contains 4 types of fixtures, labeled 1-4 in the figure (1 = start, 2 = pivot, 3 = slot, 4 = USB connector). The cable begins at the start and ends being inserted into the USB connector. In between, the cable may be wrapped around the yellow pivots or woven through red slots. A UR5e robot with a GelSight sensor attached to the gripper is used to manipulate the cable. We also have a Azure Kinect camera overlooking the board to capture the board configuration.

4.2 Related Work

4.2.1 Deformable linear object manipulation and assembly

Manipulating deformable linear objects such as cables and ropes with robots is generally challenging due to the objects' infinite degrees of freedom. Previous works have attempted to design methods ranging from state estimation, representation learning, motion planning, to end-to-end learning.

In an early work [113], the cable deformation due to external forces is estimated using stereo vision, and manipulation techniques are developed to straighten the cable for through-hole insertion. [109] proposes a novel spatial representation between the cable and environment objects for motion planning. In recent years, end-to-end approaches were proposed to learn the deformation model from simulation and manipulate cables to a target shape[114]. These works commonly assume the system to be quasistatic and achieves manipulation using repetitive pick and place actions [115] [116]. Few approaches leverage environment contacts when manipulating cables, e.g., leveraging force-torque sensing [117] or visual perception [118]. [119] proposes a task-space planner, which builds a roadmap from predefined tasks

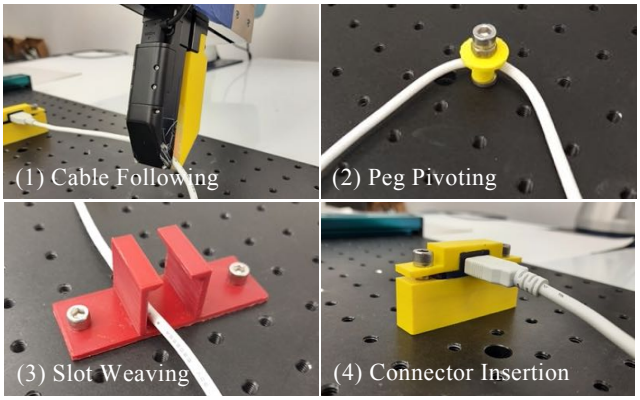


Figure 4.3: **Task board operations.** Our task is split into 4 operations: 1) cable following (gripping and following along the cable), 2) peg pivoting (wrapping the cable around the vertical peg), 3) slot weaving (threading the cable through the horizontal slot), and 4) connector insertion (inserting the USB head at the end of the cable into the connector).

and employs a replanning strategy based on a genetic algorithm which executes cable routing using a dual-arm robot. [120] developed a system for insertion of wire-terminal insertion via visuo-tactile methods. [121] developed Bayesian state estimation methods to predict symbolic states with predicate classifiers for connector insertion. In this work, we aim to build an entire cable routing and connector insertion system using visual sensing for initial plan generation and then tactile perception to monitor the cable-environment contact state, and adjust the control policy accordingly.

4.3 Problem Statement

4.3.1 Task

We aim to solve the cable manipulation problem inspired by the NIST Assembly Task Board 3[111]. We consider a taskboard that consists of fixtures such as pegs and channels to route and manipulate the cable to a pre-specified configuration. The taskboard consists of 4 different types of fixtures in various configurations: 1) Start fixture, 2) Vertical pegs / pivots, 3) Horizontal slots, and 4) USB connector. An example of the taskboard with fixtures with its goal configuration is shown in Fig. 4.2.

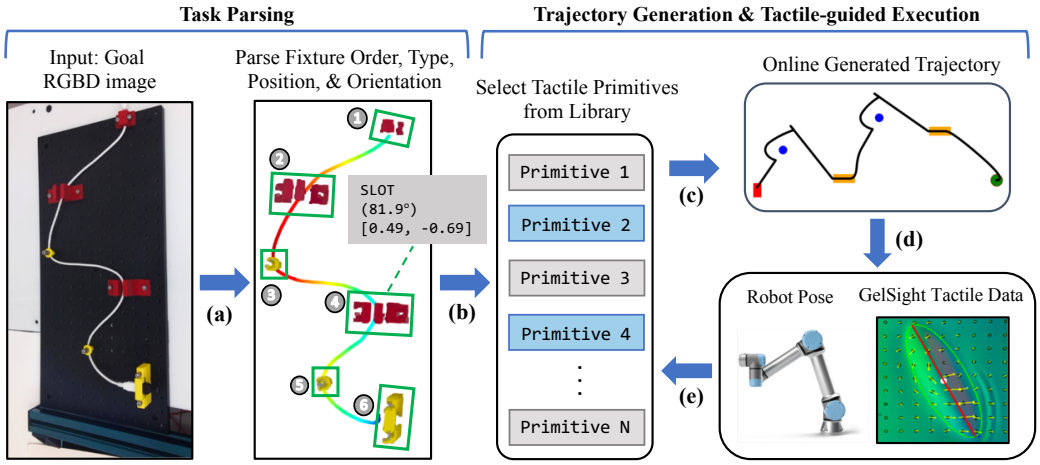


Figure 4.4: **Pipeline overview.** We propose an end-to-end framework for cable routing and assembly. (**a**) Our vision module takes in RGBD images of the goal taskboard with and without the cable to output the task specification. We use color filtering to determine the fixtures' positions and types, Principal Component Analysis (PCA) to determine the orientations and shapes, and Coherent Point Drift (CPD) to detect the cable thus determining the fixture order. (**b**) The parsed task description (fixture order, type, position and orientation) is mapped to a sequence of parameterized tactile primitives, meanwhile generating an initial reference robot trajectory. (**c, d, e**) Sequentially, the robot executes each primitive as an individual state machine, where the state transitions are governed by the sensed tactile data. In each state, a parametric trajectory generator is activated to generate the trajectory online, for example, with lines and splines.

4.3.2 Operations

To move the cable around the fixtures on the board, we divide the entire task into the following subtasks, as illustrated in Fig. 4.3.

4.3.2.1 Cable following

The robot holds the cable and follows along without dropping it.

4.3.2.2 Pivoting around vertical pegs

The robot pivots the cable's heading direction by rotating it around the pegs.

4.3.2.3 Weaving through horizontal slots

The robot weaves or threads the cable through the horizontal channel slots that "lock" the cable inside.

4.3.2.4 Connector insertion

The robot inserts the USB head at the cable's end into the connector fixture.

4.4 Method

The proposed method starts with estimating the configuration of the taskboard and the goal cable configuration from an assembled taskboard. Once the taskboard configuration is estimated, a task description is generated and mapped to a sequence of parameterized tactile primitives. Each primitive is designed as an individual state machine, where the state transitions are governed by the sensed tactile data. The robot executes primitives sequentially until task completion.

4.4.1 Perception systems & task parsing

The perception system uses RGBD data from a Microsoft Azure Kinect camera overlooking the task board and tactile data generated by a GelSight R1.5 optical tactile sensor.

4.4.1.1 Visual perception

The visual perception module requires an assembled task board from human demonstration, and infers the goal configuration of the cable. Given a point cloud from the Kinect camera, we aim to recover the type, position, orientation and ordering of each fixture on the task board. We also track the demonstration cable in the image to parse the order of the fixtures along it. The task parsing section in Fig. 4.4 shows an example input-output pair of the visual perception module.

Fixture pose estimation: We first use a point cloud of the peg board with the fixtures but without the cable. We locate all fixtures via simple color filtering and clustering. We apply Principal Component Analysis (PCA) to estimate the type (using the shape, or oblongness) and orientation of each detected fixture. The "start" and "connector" fixtures needs their orientation ambiguity to be resolved, so we use the cable heading direction to infer their orientations. Specifically, we take the point from the cable leading up to the given fixture and the current point of the cable the fixture is on to calculate the directional vector.

Cable state estimation: We track the cable in the demonstration board to get the correct order of the routing task. To segment the points corresponding to the white cable, we take all of the white-colored points from the point cloud from the second RGBD image defining the goal cable configuration. The resulting cable point cloud has occlusions created by the fixtures, so we use Reeb Graph [122] to construct a set of cable nodes. This gives an initialization for Coherent Point Drift (CPD) [123] to complete the cable. With the completed sequence of cable nodes, the order of the fixtures are readily determined by tracing the cable. This cable node sequence also

disambiguates the orientation of the "start" and "connector" fixtures, as described in the paragraph above.

Using this visual perception pipeline, the type, location, orientation and order of the fixtures are estimated, which defines the routing and assembly task. This task description then serves as the input for the motion primitives introduced in Section 4.4.2.

4.4.1.2 Tactile system

We use the tactile reading to estimate the cable state in different stages and adjust the manipulation strategies accordingly. The fingertip GelSight sensor provides tactile images of the contact area and marker displacement information corresponding to the 3-axis forces and in-plane torque on the contact surface. As introduced in [73], the change of the color in the GelSight images corresponds to the contact geometry and the motion of the markers in the images indicates the contact force and torque: a "spreading out" pattern of the markers' motion indicates normal force, a uniform motion pattern of the markers indicates shear force towards the motion direction, and a spiral pattern indicates an in-plane torque. The magnitude of the marker's motion is approximately linear to the magnitude of the force.

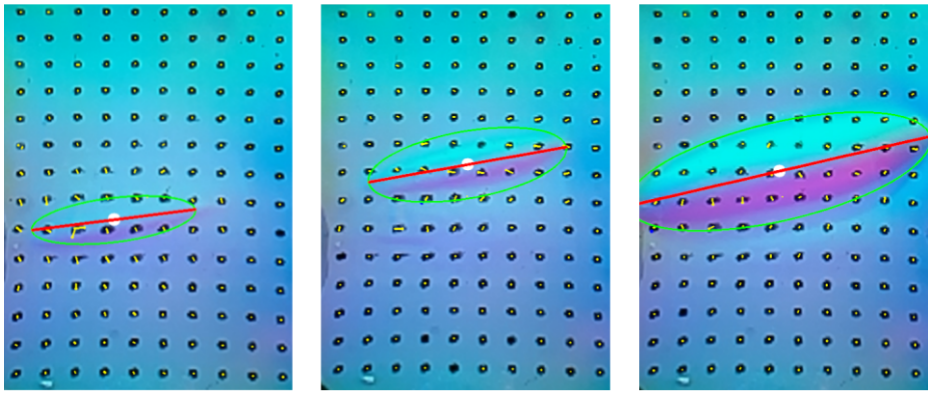
In this work, we estimate the contact area of the cable based on color and background image subtraction. The contact area of the cable is elliptical in shape, as shown in Fig. 4.5. The area also corresponds to the normal force, which is used to detect the firmness of the cable gripping. We fit an ellipse to the contact area and use its center and major axis to estimate the pose of the cable in hand. This helps the robot to re-center and re-orient the cable in the gripper.

Force and torque changes from the markers are tracked to identify cable states, such as a slowly increasing shear force along the cable indicating tight cable hold, and fast changing contact force/torque indicating a collision between the cable and the fixture. Fig. 4.6 shows some example tactile images used for state estimation at different stages.

The GelSight marker magnitudes also serve as input for the hybrid force-position controller used in connector insertion. GelSight data is sampled at 60 Hz with a latency of around 75 ms, while the force-position controller is a cascaded PID controller with an update frequency of 250 Hz. To match the controller frequency, the low frequency GelSight data is linearly interpolated in time. To mitigate the destabilizing effects of latency, high-level commands are issued to the controller at a considerably lower frequency than the control loop's operating frequency.

4.4.2 Motion primitives and trajectory generation

The vision system infers an ordered list of fixture type, position, and orientation for task specification. This information is utilized to generate a reference robot path, as illustrated in the Online Generated Trajectory subplot in Fig. 4.4, which is divided



(a) Grip Force = 5N (b) Grip Force = 10N (c) Grip Force = 25N

Figure 4.5: When holding the cable with different forces, the contact area measured from GelSight is an elliptical shape. The area of the ellipse is roughly linear to the gripping force, and its center and orientation help estimate the cable pose.

into sections for each fixture. Motion primitives designed for each task are executed for the corresponding path sections. The robot begins sequentially sets the end position of the previous primitive as the starting position for the next until the task is completed.

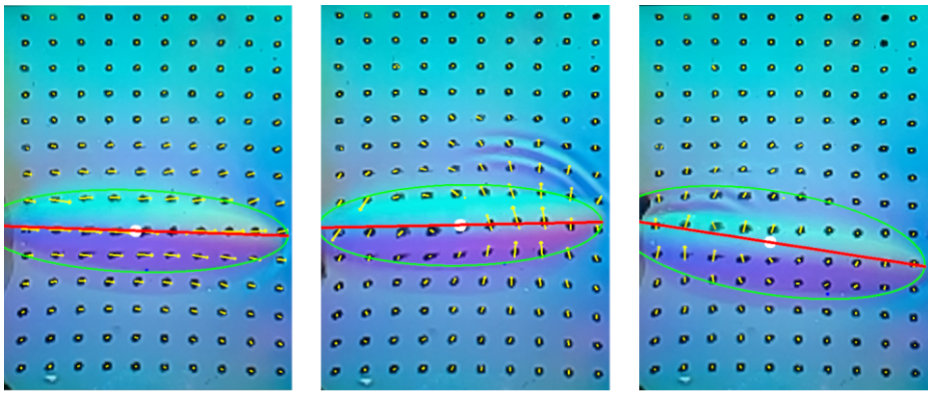
The primitives are parameterized and modeled from observing a human performing the task and deriving heuristics. The primitives are designed as state machines, whose state transitions are triggered by the tactile signals generated by the external forces on the cable, as shown in Fig. 4.7. Following are the four primitives:

4.4.2.1 Cable following

The robot performs this primitive to guide the cable from one fixture to another. As illustrated in the first row of Fig. 4.7, the primitive executes the following state sequence in a loop: i) the gripper slowly closes until the gripping force detected by GelSight exceeds a threshold; ii) the robot pulls the cable until the sensed shear force exceeds a threshold, indicating the cable tension; iii) the gripper slowly opens until the gripping force falls below a threshold; iv) based on the detected cable pose in the gripper, the robot slides along the cable while keeping the cable centered.

4.4.2.2 Pivoting around pegs

The pivoting primitive changes the direction of the cable to make it pivot around a vertical peg, as shown in the second row in Fig. 4.7. This primitive starts with following along the cable until a waypoint, determined by the peg position. The robot tilts backwards and moves down until the tactile signals indicate the cable



(a) Shear Force (b) CCW Torque (c) CW Torque

Figure 4.6: **Motion patterns of the GelSight markers.** We use the motion patterns of the GelSight markers to determine contact forces and torques which are used to estimate the cable states in different stages. (a) Shear forces along the cable indicate tension when dragging the cable. (b) A counterclockwise (CCW) torque is generated when the USB cable head hits the connector fixture from the top. (c) A clockwise (CW) torque is produced when the cable rear makes contact with a slot fixture from the top.

is in contact with the taskboard, as illustrated in Fig. 4.7 top row, plot (b) . These tactile signals looks similar to the clockwise torque signals in Fig. 4.6 (c), ensuring that the cable gets tucked under the pivoting peg. The robot then moves in a circular trajectory centered at the initial position, and continues until the cable makes contact with the peg as in plot (c). After establishing contact between the cable and peg, a new circular trajectory is generated using the peg's position as the center.

4.4.2.3 Weaving through slots

The weaving primitive is used to guide the cable through horizontal slots, as shown in the third row of Fig. 4.7. It is parameterized by the slot position and orientation, which are used to generate a reference trajectory to place the cable into the slot from the top. If the cable is not aligned with the slot during the downward motion, as indicated by a sudden increase of the in-plane torque from the GelSight image, the robot executes a horizontal wiggling action while moving down. It is continued until the in-plane torque disappears, signaling that the collision is resolved and the cable is aligned with the slot.

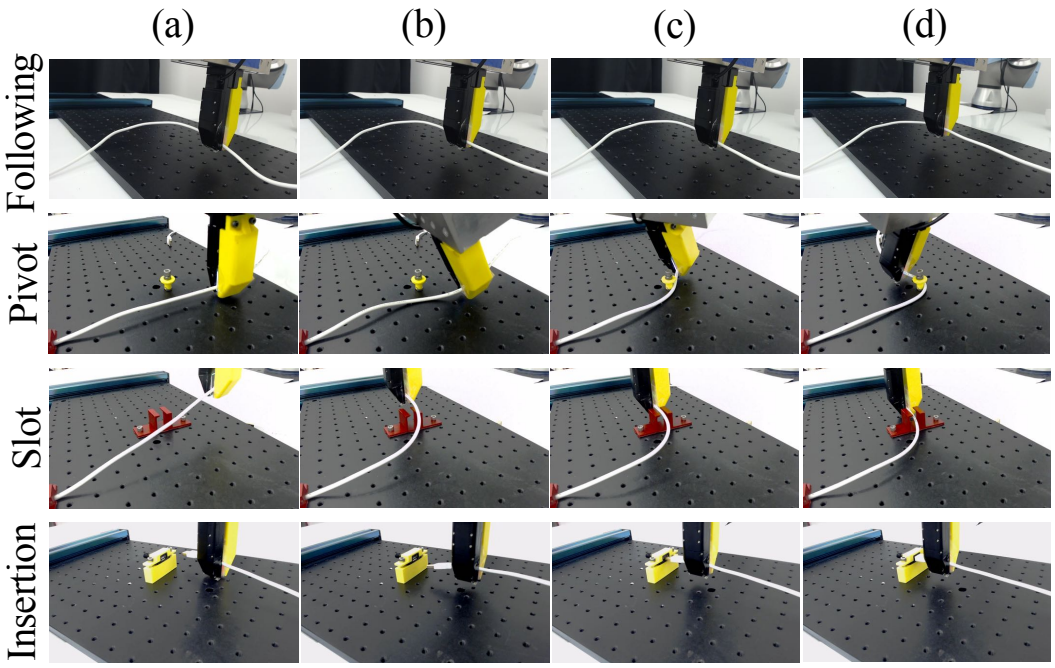


Figure 4.7: **Motion primitives.** To complete our cable routing and assembly task, we construct a library of motion primitives. The cable following primitive moves the robot along the cable, while pulling it in a desired direction. The pivot primitive wraps the cable around a peg while maintaining cable tension. The slot primitive inserts the cable into the slot and "locks it" in place. The insertion primitive aligns and inserts the USB connector into the socket.

4.4.2.4 Connector insertion

This primitive conducts tethered USB connector insertion given the socket position and orientation, as shown in the bottom row of Fig. 4.7. The primitive starts from a grasp on the cable near the connector, which enables a certain amount of freedom for the connector to bend when hitting the rigid side of the socket fixture. When the connector contacts the socket fixture, the GelSight sensor detects a normal force along the cable. We then command the robot using a hybrid force-position controller to rotate around the normal axis while keeping the normal force. The motion enables the connector to move for a small distance around the initial contact location. When the connector falls into the socket, there will be a sudden drop in the normal force on the cable and a strong torque that stops the cable from continuing rotation. Both the change of force and torque can be detected by GelSight, and we will make the robot stop this "exploration" procedure and push forward to insert the connector into the socket. In some cases, if the initial contact point is too far away from the socket, the rotary exploration motion will not get the connector in-

Primitive	Cable Following	Peg Pivoting	Slot Weaving	Connector Insertion
w/o Tactile (Baseline)	43/50	28/50	32/50	0/50
w/ Tactile (Ours)	**50/50**	**50/50**	**50/50**	**48/50**

Table 4.1: **Success rate of individual primitives.** For each primitive, we calculate the success rate with and without tactile sensing.

side the socket. We will then make the robot retreat and start the exploration from another randomly-sampled location near the socket.

4.5 Experiments

In this section, we evaluate the performance of our cable routing and assembly pipeline. Specifically, we investigate the benefits of tactile sensing in cable manipulation tasks and the reliability of our designed tactile primitives.

4.5.1 Robot setup

Our full setup is shown in Fig. 4.2. We use a UR5e 6-DoF robot arm switching between position control and hybrid force-position control modes. To grasp the cable, we use a Weiss Robotics WSG-50 gripper, which operates in an indirect force control. Mounted on the gripper is a GelSight R1.5 sensor along with a custom designed and 3D printed finger which has design features to prevent accidental cable drops. These design features include an inner surface with an elastomer pad with adequate friction, a shape which bulges slightly towards the tip, and a beak on one side which helps picking up thin cables from the flat surface and holds it from falling down. We also use a Microsoft Azure Kinect camera to capture the entire taskboard and acquire RGBD data for task parsing.

4.5.2 Motion primitives with and without tactile sensing

We demonstrate the robustness of the tactile-guided motion primitives by comparing the success rate against a baseline without tactile sensing. The baseline applies the same trajectory which is generated analytically and does not use tactile sensing for adapting it.

We run 50 trials for each primitive (see Fig. 4.7) with varying fixture positions, and present the results in Table 4.1.A trial is considered success if the robot is able to properly perform the corresponding primitive on that fixture. For all four primitives, our tactile-guided approach significantly outperforms the baseline without tactile sensing. The connector insertion primitive shows this the most clearly: the

	Success Rate	Failure Modes
GT + Tactile (Ceiling)	53/60	A(3), B(2), C(2), D(0)
GT + No Tactile (Baseline)	0/60	A(0), B(2), C(10), D(48)
Vision + Tactile (Ours)	**51/60**	A(4), B(3), C(2), D(0)

Table 4.2: **Success rate of entire pipeline.** We summarize the success rate of the whole task across 6 different board configurations with 10 trials each. Failure Modes: A (Connector insertion failed), B (Cable got stuck), C (Cable escaped from previous fixture), D (Accumulated error in motion)

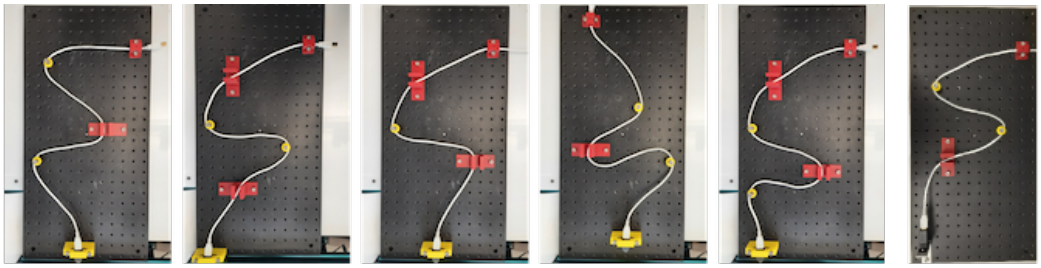


Figure 4.8: **Taskboard configurations.** We evaluate cable routing and assembly on each of these 6 boards 10 times.

baseline has a success rate of 0% because the robot cannot align the orientation of the USB head with the connector, whereas with the tactile sensing, the controller can reorient the connector pose to align with the socket. The other baseline primitives fail due to the inability to estimate contacts and maintain tension in the cable.

4.5.3 End-to-end cable routing and assembly

We analyze the reliability of our full pipeline over various board configurations. First to evaluate the performance of our vision pipeline described in Section 4.4.1.1, we compare our approach (Vision + Tactile) against an oracle baseline that uses the ground truth (i.e. human-specified) board configuration (GT + Tactile). We consider this to be an upper bound on performance. To evaluate the effectiveness of using tactile sensing, we experimented with a baseline without tactile sensing but using the ground truth board configuration (GT + No Tactile).

We compare the success rate and failure modes by evaluating each approach across 6 different board configurations (Fig. 4.8), with 10 trials each, summarized in Table 4.2. We label the trial as a success if the entire task was completed. The oracle which uses ground truth board configurations (53/60) performs only slightly better than our Vision + Tactile pipeline (51/60), suggesting that our vision pipeline can effectively parse the board task configuration. We find that the baseline without tactile sensing fails catastrophically due to the accumulated motion error, typically

after the second or third fixture. While individual primitives could be completed without tactile sensing, although with limited success, as shown in Table 4.1, we find that chaining the primitives reliably is difficult. This happens due to the lack of tactile feedback which corrects the trajectory while following the parsed path. The robot may also experience failures beyond its direct control, such as cable entanglement in fixtures or cable detachment from previously completed fixtures. During complex operations such as pivoting, weaving, and insertion, tactile signals are crucial for inferring the cable-environment interaction state to online adapt the trajectory. As summarized in the failure modes, our method occasionally fails because of the cable getting stuck or slack, and connector insertion remains the most challenging operation. We postulate such "global" state changes cannot be sufficiently captured by "local" tactile signals. Thus we plan to fuse the global visual sensing with local tactile sensing in the future to further improve the reliability.

4.6 Conclusion

This chapter studies a long-horizon task that involves cable routing and tethered object assembly. Particularly, we propose a novel integrated pipeline where a task is parsed via visual perception and trajectories are planned and executed in a closed-loop manner with tactile-guided motion primitives. We compare the designed method and other baselines on a reconfigurable task board, which may serve as a benchmark for future research. Experiment results indicate the necessity of tactile sensing in such tasks involving dexterous operations with cables in a realistic environment.

One key limitation of our work is that the primitives are hand-designed rather than learned from data, thus limiting their generalizability. It is interesting to create more diverse and powerful primitives leveraging imitation learning and reinforcement learning. Another limitation is the visual and tactile perception being used at different stages. We aim to improve on these limitations in the next work.

Chapter 5

A Touch of Precision: Learning Visuo-tactile Policies for Fine-grained Manipulation from Demonstrations

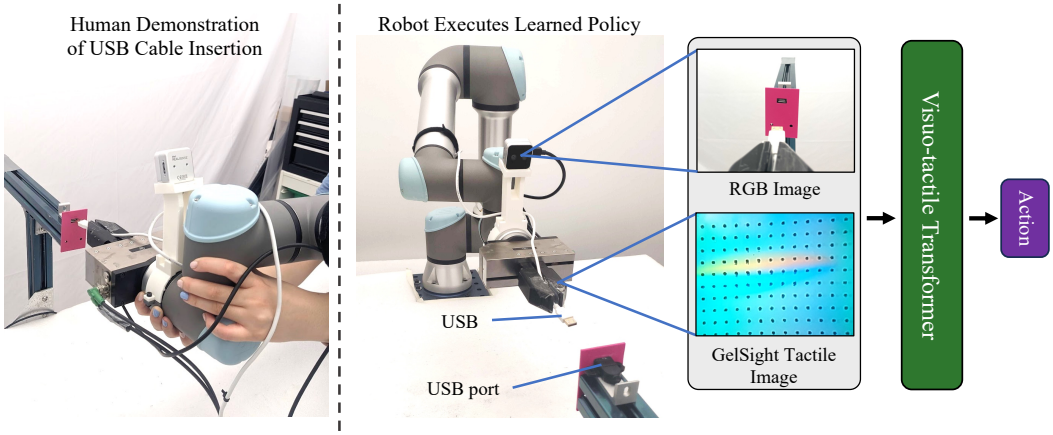


Figure 5.1: **Transformer-based policy for USB cable insertion.** We consider the task of teaching a robot to insert a USB cable into the port, which is especially challenging due to deformations of the cable (e.g. slipping, bending), and the task also requires sub-millimeter precision. We propose a learning-based policy that fuses tactile data from the finger-mounted GelSight with RGB data from a wrist-mounted camera. We train this multimodal model using 30 human demonstrations collected using teleoperation and kinesthetic teaching.

5.1 Introduction

Robot manipulation has seen a lot of progress in recent years. However, much of this progress has been limited to interacting with rigid objects and performing coarse-grained actions for manipulation, such as pushing and stacking. To accomplish more, home and industrial robots need to achieve fine-grained manipulation which involves high-precision physical interactions. These interactions are challenging to understand from vision alone, and may require additional sensory modalities.

When humans perform fine-grained manipulation, especially for small objects, we rely primarily on touch for localization, detecting task states, and adjusting manipulation motions. For robots to perform these tasks, they should similarly be provided the same sensory feedback, but current systems typically rely on visual sensing alone. We thus consider how to fuse visual inputs and high-resolution tactile information from a GelSight sensor for performing precise manipulation.

We consider the task of USB cable insertion, which requires sub-millimeter precision and handling deformable objects (cable). In contrast to typical peg insertion tasks, manipulating a soft cable is particularly challenging because its flexibility causes uncertain deformations, its thin nature increases the likelihood of slipping, and its small size leaves little margin for error. Previous work in this area has proposed creating a tactile-guided primitive library [124], but such solutions are hand-designed specifically for their task definition which limits their generalizability.

To address these shortcomings, we propose to learn a visual and tactile policy from data alone, specifically human demonstrations. This way, we avoid any hard coding and require no domain knowledge. Additionally, this allows the robot to imitate how humans would actually perform a task like USB insertion. We collect 30 human demonstrations of cable insertion using teleoperation and kinesthetic teaching. We then use behavior cloning to learn a policy that fuses vision and high-resolution tactile signals using a novel transformer-based architecture.

In this work, we have three main contributions as we: 1) propose a novel multimodal transformer policy that fuses vision with high-resolution tactile inputs, 2) demonstrate a learning-based solution for a fine-grained task that requires submillimeter precision while manipulating a deformable object (USB cable insertion), and 3) show that our policy leveraging human demonstrations significantly outperforms non-learning and learning baselines. As a proof-of-concept, we show that our model can also generalize to other cable types.

5.2 Related Work

Vision + Tactile. Combining vision and tactile sensing has proven to be beneficial to multiple manipulation tasks, including object identification [125], in-hand object pose estimation [126], and increasing grasping success rate over a wider range of objects [72]. In particular, the combination of these two modalities helps contact-

rich tasks, such as contact localization [127], peg-in-hole insertion [128], and contour following [129]. Learning-based papers have also explored how to fuse high-resolution tactile and visual data together via concatenation [130, 72], a two-stage process [131] or a VAE [132].

Additional approaches have also explored fusing other tactile modalities, like force/torque feedback [128, 133]. In particular, [133] extends visual transformers [134] to handle vision and force-torque feedback. While we similarly use a transformer-based architecture, we design our approach to handle visual and GelSight's high-resolution tactile data.

Fine-grained Insertion. Previous works have explored (USB) cable insertion. [124] used a library of hand-designed tactile primitives to perform cable routing and insertion. We aim to learn this task automatically using behavior cloning. [131, 132] both consider the task of USB insertion, similar to us. However, they simplify the problem by grabbing the rigid USB head whereas we focus on previous cable manipulation works by grabbing the deformable cord itself. While this makes the task more challenging, it better tests the robot's ability to handle thin and deformable objects, provides richer tactile feedback, and could potentially generalize to other cables.

Behavior Cloning. Behavior cloning is a popular approach in machine learning that involves training an agent to imitate the behavior of an expert. Behavior cloning has been used to tackle tasks such as playing video games [135] and driving autonomous vehicles [136, 137]. In robotics, this form of visual imitation learning has been applied to pushing [138, 139], stacking [139, 140], and in-hand object manipulating [141]. We build off of these works, combining ego-centric videos with high-resolution tactile feedback collected using expert demonstrations. We train a behavior cloning policy that learns from these demonstrations for the task of USB cable insertion. For a comprehensive summary of imitation learning, see [142].

5.3 Method

We propose a method that uses imitation learning from human demonstrations to perform the task of USB cable insertion based on visuo-tactile input. In this section, we describe our setup and task, our multimodal transformer-based behavior learning policy, and our data collection process.

5.3.1 Robot and Task Setup

We use a Universal Robotics UR5e 6-DoF robot arm and a Weiss Robotics WSG-50 gripper to manipulate the cable. Unlike previous methods that tackle insertion [131, 132], our gripper holds the cable instead of the USB head. This provides richer sensory feedback, tests the robot's ability to handle finer-grained deformable object manipulation, and potentially allows for better generalization to other cables. The

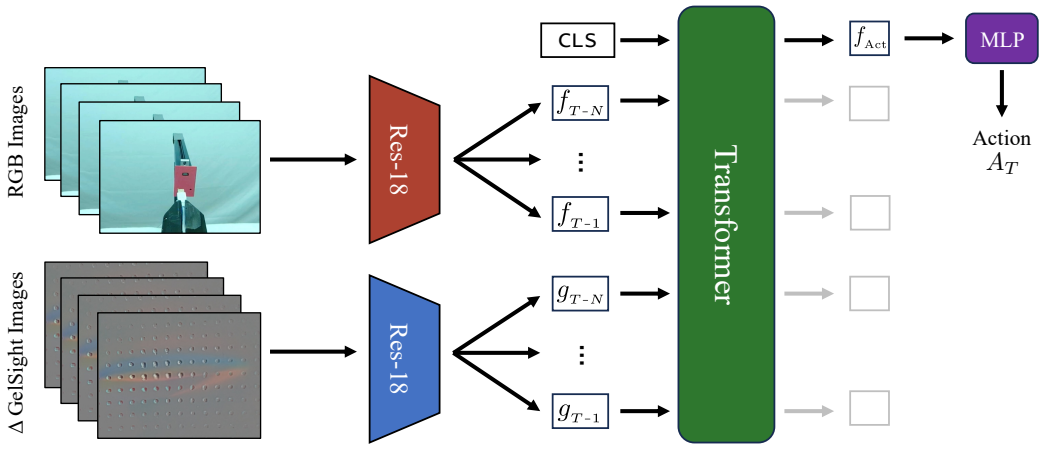


Figure 5.2: **Visuo-tactile transformer architecture.** Given a sequence of RGB and delta GelSight images, we first extract per-frame ResNet features. We then treat these features as tokens along with a CLS token [134] as input into a transformer. We then pass the output features corresponding to the CLS token into a light-weight MLP which then outputs the next predicted action to take in the form of direction and magnitude.

gripper has a GelSight R1.5 optical tactile sensor mounted on it for high-resolution tactile data. For ego-centric RGB data, a RealSense D405 camera is mounted on the wrist of the robot end effector.

The goal of the USB insertion task is to fully insert the USB cable into a USB port without prior knowledge of the port position relative to the current robot position. Our setup is shown in 5.1. We mount a 3D-printed magenta board with a USB port on the side of the table. The board is in a fixed position. At test time, the start position is randomized to show the robustness of our method.

5.3.2 Multimodal Fusion Using Transformers for Behavior Cloning

We now describe our approach for the task of USB cable insertion. Given trajectories containing sequences of ego-centric RGB images, tactile images, and the ground truth robot end-effector poses collected through human demonstrations, our goal is to learn a policy that predicts the next action to take. In practice, we subtract the initial tactile image from all frames in the trajectory, similar to [130], to keep the focus on the deformation of the gel (See 5.3). We train a neural network f that predicts an action given the previous L images and tactile inputs:

$$f(I_{T-N}, G_{T-N} ... I_{T-1}, G_{T-1}) = A_T \tag{5.1}$$

for sequence length N, RGB image $I_i \in \mathbb{R}^{H \times W \times 3}$, delta GelSight image $G_i \in \mathbb{R}^{H \times W \times 3}$, and predicted action $A_T \in \mathbb{R}^4$. We represent actions as a 4D vector consisting of di-

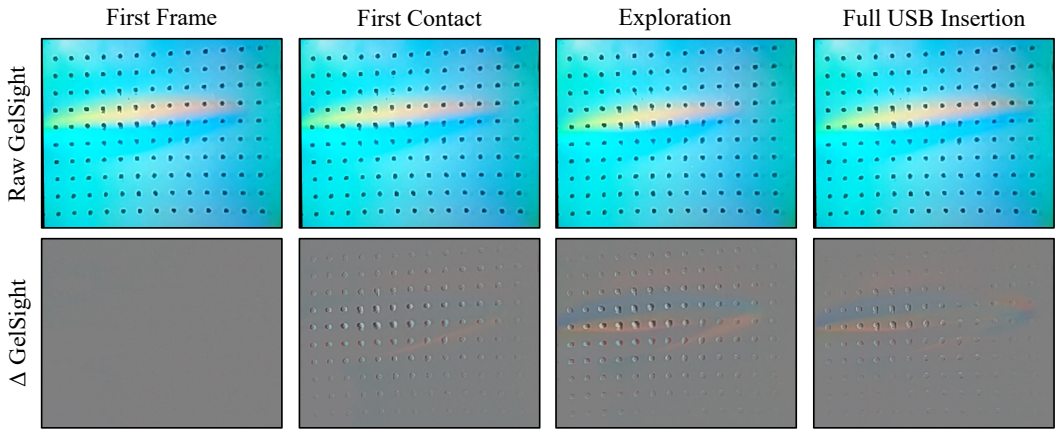


Figure 5.3: **Informativeness of raw versus delta GelSight images.** *Top:* Raw GelSight Images with marker motion at different stages of the robot trajectory. *Bottom:* Following [130], we subtract each GelSight image from the first GelSight frame in the trajectory. This highlights only the changes in the deformation of the gel, especially the motion of the markers that denote the contact shear forces and torques.

rection ($D_T \in \mathbb{R}^3$) and magnitude ($M_T \in \mathbb{R}$). We use a sequence length of $N = 4$, which we validate in our experiments to provide optimal performance.

Our model first extracts features for all of the RGB and GelSight images. Each RGB image is passed through a ResNet18 [104] with pre-trained R3M weights [143]. Because R3M is trained on ego-centric data, it generalizes better to ego-centric robotic tasks. To extract the tactile features, we pass the delta tactile images in the sequence into a different ResNet18. We find that the default ImageNet [144] pre-trained initialization is sufficient. The ResNet features are spatially pooled into a 512-dimensional vector. We input all $2N$ ResNet features as tokens to a transformer. Because we only want one token output to correspond to the predicted action, we follow ViT [134] and include a randomly initialized learnable CLS token as input to the transformer. After the transformer processes all $2N + 1$ image, tactile, and CLS tokens, a small MLP maps the CLS token feature to a 3D direction vector and an action magnitude value. We show our transformer-based model architecture in 5.2.

We treat the *change* in the end-effector position from the demonstration as the ground truth action. During training, we use three losses. We use a cosine similarity loss for the predicted direction and a robust Huber loss ($\delta = 1$) for the predicted magnitude. Even though at inference time the direction vector is normalized, to regularize it during training, we add an additional loss that penalizes the predicted direction vector from deviating from the unit norm: $\mathcal{L}_{\text{reg}} = \|D_T - 1\|^2$. We train with all losses equally weighted.

5.3.3 Data Collection

To train our behavior cloning model, we collect a dataset of human demonstrations using kinesthetic teaching and teleoperation. We use kinesthetic teaching, where the robot is physically guided by the human, to collect the demonstrations involving the actual insertion of the USB into the port because this task requires a high level of fine-grained manipulation. To increase diversity, we also collect trajectories using teleoperation where the robot starts at a random point and ends at a point close to the port. Combining these two types of human demonstrations allows for less noisy data when approaching the USB port while not sacrificing the important information required for the challenging task of insertion. Example videos from the dataset are shown in the supplementary.

We collect 12 trajectories using teleoperation and 18 trajectories using kinesthetic teaching. The teleoperation trajectories contain about 50 frames while the kinesthetic trajectories are 100-150 frames each. Each trajectory takes $\sim$1 minute to collect from start to finish, highlighting the convenience of using human demonstrations as data. See more implementation details in the supplementary.

5.4 Experiments

5.4.1 Baselines

We compare our proposed method with two non-learning baselines.

Visual Servoing Baseline. We develop a hand-crafted policy that combines a heuristic vision detection solution with a simple control sequence. We use RGBD data from the wrist-mounted ego-centric camera to manually detect the location of the USB hole in 3D by color thresholding the magenta backboard and the USB hole. The robot then moves towards the determined position of the port and slowly moves forward until there is either insertion or contact with the backboard.

Visual Servoing + Tactile Primitive Baseline. We also evaluate the tactile-guided cable insertion method proposed by [124] in combination with visual servoing. The baseline builds a library of tactile primitives that maps GelSight marker motion patterns to action sequences. Whenever a target contact condition is met, the robot executes a corresponding primitive chosen from the library, and repeats until insertion is detected. We initialize the start position of the robot using the output of the same vision detection algorithm as the visual servoing baseline (described above) and followed by this tactile-guided insertion method when the head is in front of the USB port.

In addition to the above baselines, we also run ablations of our method to test our design decisions.

Vision Transformer. We compare our multimodal transformer policy to the same architecture trained and evaluated on only the RGB images. We still use a CLS token

	Contact with Port	Full Insertion
Visual Servoing Baseline	19/20	3/20
Visual Servoing + Tactile Primitive Baseline [124]	19/20	9/20
Vision-only Transformer	19/20	1/20
Vision + Tactile Concatenation	16/20	7/20
Vision + Tactile Transformer (Ours)	17/20	**14/20**

Table 5.1: **Comparison with baselines and ablations on USB cable insertion.** The Visual Servoing Baseline employs a hand-designed USB port detector using RGBD data from the ego-centric camera and moves the robot end-effector into position before moving forward. While it is able to localize the port with 1-2mm error and get close contact almost perfectly, the baseline is unable to fully insert the USB consistently. The Visual Servoing + Tactile Primitive Baseline [124] uses the same port detector as the previous baseline and then uses a hand-designed library of tactile-guided primitives to insert the USB cable. Because it uses the GelSight information during insertion, it is able to recover from the noise more often, leading to an improved success rate of full insertion. Our transformer-based policy learns to fuse vision and tactile information from human demonstrations and achieves the best insertion performance. We consider a vision-only ablation that struggles to insert the cable despite coming in contact with the port, suggesting that tactile information is necessary for recovering from millimeter misalignment. We also consider an ablation that replaces the transformer architecture with an MLP that concatenates the modalities directly, but it does not perform as well as our method.

	1	2	4	8
Success Rate	0/10	0/10	**8/10**	5/10

Table 5.2: **Evaluation of temporal context for transformer policy.** We train and evaluate our visuo-tactile transformer policy with various sequence lengths. We find that a sequence length of 4 seems to perform best. Too short of a sequence length provides insufficient temporal information. Too long of a sequence length may provide too many uninformative inputs in addition to being computationally expensive.

to read out the action, so the transformer uses $N + 1$ tokens in total (as opposed to $2N + 1$ for RGB + GelSight).

Multimodal Fusion Using Concatenation. In our method, we proposed a transformer-based architecture to fuse the features from vision and tactile. In the spirit of previ-

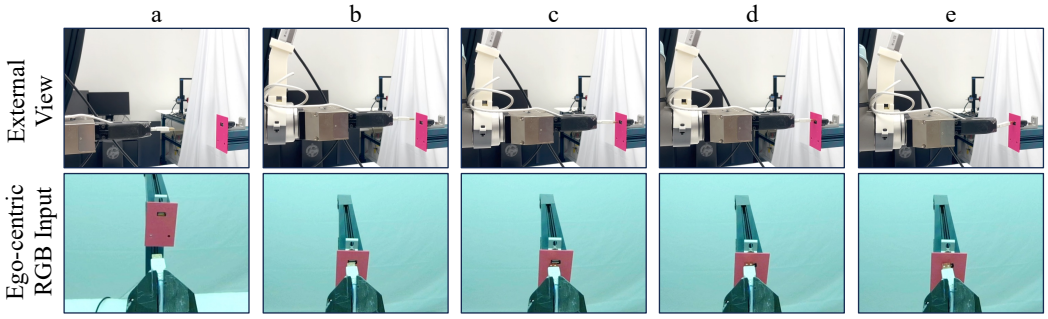


Figure 5.4: **Challenging depth ambiguity for the vision-only transformer.** To successfully insert the USB cable into the port, the policy must first get the cable in front of the hole and then push forward while wiggling. However, for the vision-only model, it is very difficult to determine when the USB head is directly in front of the port. For instance, from the camera view, (b) - (d) look almost identical and appear to be very close to the insertion stage. However, as seen in the side view, they are at varying distances from the port. As a result, the model often begins pushing forward and wiggling too soon, causing misalignments once contact is finally reached. As the model is unable to determine contact from the ego-centric view alone, the robot continues to push forward despite misalignment, causing the cable to slip and fail (e).

ous works that fuse these modalities using concatenation [130, 72], we also implement a concatenation-based architecture. More specifically, we extract a sequence of vision and tactile features using ResNet18s and concatenate them into a single vector that is then fed into an MLP. We ensure that this model has a similar number of learnable parameters to our transformer model for a fair comparison. As with our method, we use a sequence length of 4.

Sequence Length. To test the importance of temporal information as well as search for the optimal amount of context, we train and test our model on varying sequence lengths: $N \in \{1, 2, 4, 8\}$.

5.4.2 Experimental Results

We evaluate our multimodal transformer policy with both baselines and the two architecture ablations in 5.1. For this experiment, we start from 20 pre-specified positions with the robot end effector about a foot away from the USB port. We evaluate two metrics: success rate of contact with the port and full insertion. A trial is considered to have achieved contact with the port if the USB head touches any portion of the port. A full insertion occurs when the USB head is fully lodged into the USB port. The trial is considered finished when either 1) the USB is fully inserted, 2) the change in end effector position is less than 1mm for 5 consecutive

frames, or 3) the cable has slipped in a way that could damage the robot, gelpad, or USB port.

The Visual Servoing Baseline, which detects the USB port using the RGBD information, is able to localize the hole with an error of roughly 1-2mm. For reference, the USB port is around 15 $\times$ 21mm. However, this is still not accurate enough to fully insert the USB, with only 3 of the 20 trials achieving full insertion. This is because a simple push-forward policy is unable to recover from even the most minor of misalignments.

The Visual Servoing + Tactile Primitive Baseline is able to better recover from the same 1-2mm misalignments of the vision pipeline from the Visual Servoing Baseline by using GelSight marker information and applying a hand-designed primitive for insertion. While it was able to achieve significantly better insertion performance (9/20) than the push-forward baseline, we found that it was prone to cable slipping. To see qualitative results of both of these baselines, please refer to the supplemental video.

We find that our vision-only transformer using behavior cloning is able to achieve close contact with the port, showing that learning from human demonstrations can perform comparably to hand-designed vision detection solutions. However, it also fails at full insertion because it is not able to gather the depth information accurately. There is a lot of occlusion near the end of the trial when the USB is close to the hole. Refer to 5.4 for an illustrative example. From the camera view, the USB looks like it is very close to insertion, but the side view shows that it is actually quite far. Because of the angle of the wrist-mounted camera and the occlusion caused by the USB itself, it is not able to accurately determine the cable's relative position to the port. This is a problem because in order to successfully insert, the robot must first place the USB head directly in front of the port and then push forward while wiggling the cable. Since the vision-only model is unable to determine how far from the port it still is, it begins to push forward and wiggle too soon. This causes misalignment or cable slipping once there is contact.

Our multimodal transformer policy has a slightly lower contact with port accuracy than the previous models because it is learning both visual and tactile information. The model learns to use tactile sensing to determine collision and contact, but sometimes there is cable slipping due to the cable's infinite degrees of freedom, and this slip can confuse the model and cause it to drift after initial contact. However, our model is able to achieve the highest performance for full insertion, as tactile sensing is necessary for detecting contact and recovering from minor misalignment, as 5.5 shows.

We also demonstrate that transformers are an effective way to fuse tactile and vision features when compared to the concatenation-based approach as done in previous work. While both the concatenation model and our transformer model use multimodal data, our policy is able to learn when to weigh vision or tactile images throughout the trial. 5.6 shows the relative attention on each modality over

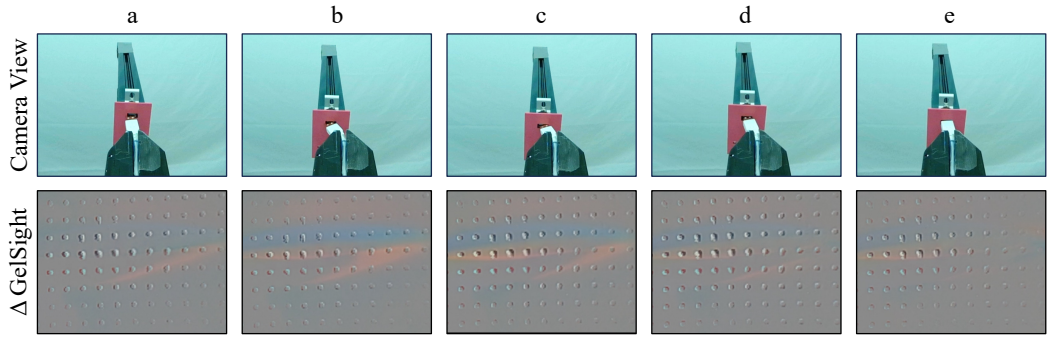


Figure 5.5: **Visuo-tactile transformer successfully recovering from misalignment.** When there is minor misalignment during initial contact (a), the policy uses the tactile signals to find the direction of misalignment and adjust until the USB fits snugly into the port (b, c). The policy then continues to make more adjustments (d) until finally inserting the USB into the port (e).

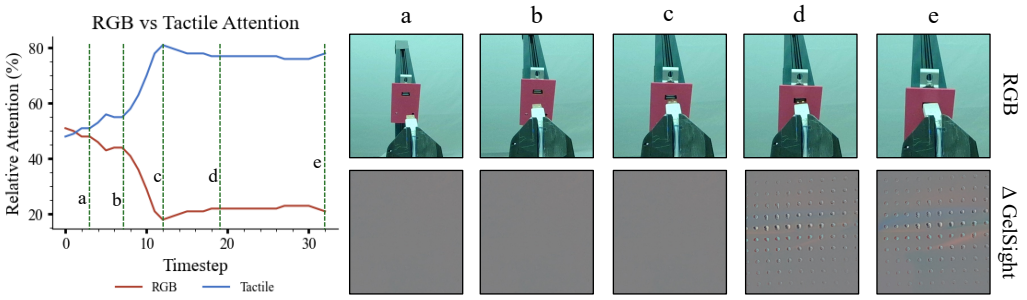


Figure 5.6: **Relative RGB and tactile attention during successful trial.** On the left, we plot the proportion of the CLS token's attention on GelSight images versus RGB images to see what the model attends to over the course of the trial. The images on the top right correspond to the RGB images at the specified timestamps; the images on the bottom right correspond to the delta GelSight images. The network only pays attention to the visual features early in the trajectory when localizing the USB port (a, b). The network pays the most attention to the tactile features when the USB head is near but not quite touching the board (c). This is probably because the tactile information is able to signal when there is contact. After there is contact with the port, the relative weight on the GelSight images remains high (d, e). The tactile information is more informative than RGB images in detecting and fixing misalignments and ultimately completing the insertion.

the course of a successful trial. At the beginning of the trial, the policy weighs vision more heavily, but as the trial continues and the USB gets closer to contact, the policy starts prioritizing the GelSight features much more strongly.

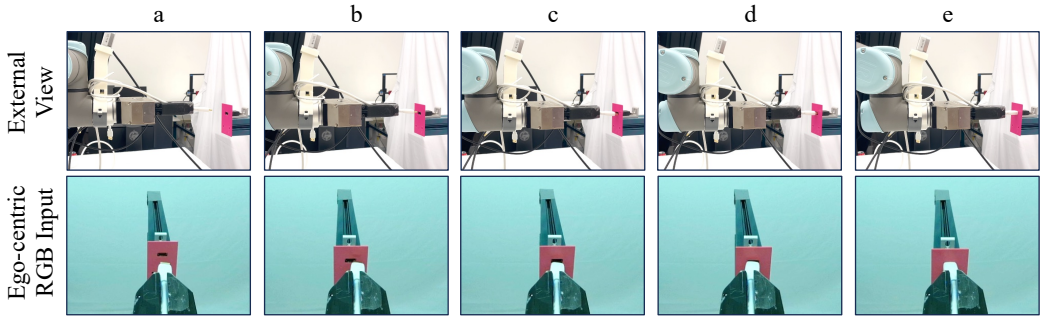


Figure 5.7: **Generalization to unseen cable type (HDMI).** We demonstrate that our visuo-tactile policy trained only on USB insertion can generalize to HDMI cables with no fine-tuning (zoom in for better view). This is one benefit of the robot grasping the cable rather than the head, as the tactile readings likely generalize better.

To evaluate the importance of temporal context, we evaluate the success rate of full insertion for the transformer policy trained on sequence lengths of 1, 2, 4, and 8. For this experiment, we run on 10 trials each. Results are shown in 5.2. Short sequence lengths do not provide sufficient temporal history, resulting in the policy failing to fully insert the USB at inference time. We hypothesize that too long of a temporal context may overwhelm the policy with uninformative information in addition to being much more computationally expensive to train and run.

Finally, we show a proof of concept that our visuo-tactile policy trained on USB cable insertion alone could generalize to other cable types (HDMI) in 5.7. This is a benefit of grasping the cable rather than the head as the tactile readings could generalize better. We ran a total of 5 trials, and 5 achieved contact with port and 1 achieved full insertion.

5.5 Conclusion

In this chapter, we propose a learning-based multimodal policy that we apply to the task of USB cable insertion. Our experiments emphasize the importance of incorporating high-resolution tactile data with egocentric visual feedback for fine-grained manipulation. While our models trained on vision data is able to achieve coarse localization, adding tactile information is vital for achieving precise manipulation by fully inserting the USB. We also found that behavior cloning is a convenient and effective way of learning from human demonstrations. Finally, we believe the idea of using visuo-tactile feedback while learning from humans is an expandable idea for potentially more challenging fine-grained manipulation tasks in the future, like opening/closing drawers or assembly.

Chapter 6

Conclusion and Future Work

In summary, we first discussed how visual interaction can improve object detection performance through semantic curiosity 2. Next, we combine the ideas of visual and tactile interaction to build a new multimodal dataset called *PoseIt* for grasping objects in different holding poses 3. Then, we designed motion primitives that incorporated tactile feedback for the task of cable routing and assembly to handle deformable and thin objects 4. Finally, we learn to fuse tactile and visual feedback directly from human demonstrations for fine-grained manipulation 5.

For future work, it could be interesting to collect visuo-tactile data in-the-wild. We could use a hand-held gripper with multi-modal sensors (specifically the egocentric RGBD camera on the wrist and high-resolution tactile sensors on the gripper) to collect human demonstrations to teach robots. Since we found that using a transformer-based architecture for behavior cloning could work well for the task of USB cable insertion, we could expand on this by trying other challenging tasks such as food manipulation, and opening and closing drawers, and show that our approach can be generalized to different scenes in the real world.

Chapter 7

Acknowledgments

I would like to thank my advisor Wenzhen Yuan for my time here in graduate school. I would also like to thank Professors Abhinav Gupta, David Held, Saurabh Gupta, and Shubham Tulsiani for being such great mentors and people to go to for advice and mentorship throughout my PhD.

I would like to thank the incredible colleagues and collaborators in my lab and Ph.D. program who had a major impact on my graduate experience: Achu Wilson, Xiaofeng Guo, Tim Man, Devendra Chaplot, Shubham Kanitkar, Arpit Agarwal, Joe Huang, Uksang Yoo, Ruihan Guo, Yufan Zhang, Pengyang Shi, Arka Chaudhuri, Sudeep Dasari, Shikhar Bahl, Jason Zhang, Yufei Ye, Raunaq Bhirangi, Alex Li, Victoria Dean, Sam Powers, Adithya Murali, Kenny Marino, Lerrel Pinto, Xiaolong Wang, and many more.

And most importantly, I would like to thank my family, my mom Song, my dad Charlie, and my baby sister Connie (who is no longer a baby but will be one to me forever).

Bibliography

[1] J. Morgan, "Womb with a view: Sensory development in utero," *Your Pregnancy Matters Blog*, 2017. I

[2] E. J. Gibson and R. D. Walk, "The" visual cliff"," *Scientific American*, vol. 202, no. 4, pp. 64–71, 1960. 1

[3] J. F. Hagan, J. S. Shaw, P. M. Duncan *et al.*, *Bright futures*. American Academy of Pediatrics, 2017. 1

[4] R. J. Gerber, T. Wilks, and C. Erdie-Lalena, "Developmental milestones: motor development," *Pediatrics in review*, vol. 31, no. 7, pp. 267–277, 2010. 2

[5] J. Coplan, *ELM scale: the early language milestone scale*. Pro-Ed, 1983. 2

[6] C. Lerner and L. Ciervo, "Healthy minds: Nurturing children's development from 0 to 36 months," in *Zero to Three Press and American Academy of Pediatrics*, 2003. 2

[7] D. Martin, C. Fowlkes, D. Tal, and J. Malik, "A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics," in *Proc. 8th Int'l Conf. Computer Vision*, vol. 2, July 2001, pp. 416–423. 3

[8] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The PASCAL Visual Object Classes Challenge 2007 (VOC2007) Results," http://www.pascal-network.org/challenges/VOC/voc2007/workshop/index.html. 3

[9] D. Pathak, P. Agrawal, A. A. Efros, and T. Darrell, "Curiosity-driven exploration by self-supervised prediction," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2017, pp. 16–17. 5, 6, 11, 13, 15, 16

[10] D. Pathak, D. Gandhi, and A. Gupta, "Self-supervised exploration via disagreement," in *ICML*, 2019. 5, 6, 7

[11] R. Bajcsy, "Active perception," *Proceedings of the IEEE*, vol. 76, no. 8, pp. 966–1005, 1988. 5

[12] P. Ammirato, P. Poirson, E. Park, J. Košecká, and A. C. Berg, "A dataset for developing and benchmarking active vision," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2017, pp. 1378–1385. 5

[13] J. Yang, Z. Ren, M. Xu, X. Chen, D. J. Crandall, D. Parikh, and D. Batra, "Embodied amodal recognition: Learning to move to perceive objects," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 2040–2050. 5

[14] D. Jayaraman and K. Grauman, "Learning to look around: Intelligently exploring unseen environments for unknown tasks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 1238–1247. 5

[15] D. S. Chaplot, E. Parisotto, and R. Salakhutdinov, "Active neural localization," in *International Conference on Learning Representations*, 2018. [Online]. Available: https://openreview.net/forum?id=ry6-G_66b 5

[16] D. Fox, W. Burgard, and S. Thrun, "Active markov localization for mobile robots," *Robotics and Autonomous Systems*, vol. 25, no. 3-4, pp. 195–207, 1998. 5

[17] B. Settles, "Active learning literature survey," University of Wisconsin-Madison Department of Computer Sciences, Tech. Rep., 2009. 5, 7

[18] O. Sener and S. Savarese, "Active learning for convolutional neural networks: A core-set approach," in *International Conference on Learning Representations*, 2018. [Online]. Available: https://openreview.net/forum?id=H1aIuk-RW 5

[19] Y. Gal, R. Islam, and Z. Ghahramani, "Deep bayesian active learning with image data," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, pp. 1183–1192. 5

[20] D. Yoo and I. S. Kweon, "Learning loss for active learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 93–102. 5

[21] W. Kuo, C. Häne, E. Yuh, P. Mukherjee, and J. Malik, "Cost-sensitive active learning for intracranial hemorrhage detection," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2018, pp. 715–723. 5

[22] S. Vijayanarasimhan and K. Grauman, "Large-scale live active learning: Training object detectors with crawled data and crowds," *International Journal of Computer Vision*, vol. 108, no. 1-2, pp. 97–114, 2014. 5

[23] J. Yang, J. Lu, S. Lee, D. Batra, and D. Parikh, "Visual curiosity: Learning to ask questions to learn visual recognition," in *Conference on Robot Learning*, 2018, pp. 63–80. 5

[24] A. Fathi, M. F. Balcan, X. Ren, and J. M. Rehg, "Combining self training and active learning for video segmentation," in *Proceedings of the British Machine Vision Conference*. Georgia Institute of Technology, 2011. 5

[25] I. Misra, R. Girshick, R. Fergus, M. Hebert, A. Gupta, and L. van der Maaten, "Learning by Asking Questions," in *CVPR*, 2018. 5

[26] J. Schmidhuber, "A possibility for implementing curiosity and boredom in model-building neural controllers," in *Proc. of the international conference on simulation of adaptive behavior: From animals to animats*, 1991, pp. 222–227. 6

[27] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. MIT Press, 1998. [Online]. Available: http://www.cs.ualberta.ca/~sutton/book/the-book.html 6

[28] P. Auer, "Using confidence bounds for exploitation-exploration trade-offs," *Journal of Machine Learning Research*, vol. 3, no. Nov, pp. 397–422, 2002. 6

[29] T. Jaksch, R. Ortner, and P. Auer, "Near-optimal regret bounds for reinforcement learning," *Journal of Machine Learning Research*, vol. 11, no. Apr, pp. 1563–1600, 2010. 6

[30] B. Eysenbach, A. Gupta, J. Ibarz, and S. Levine, "Diversity is all you need: Learning skills without a reward function," in *International Conference on Learning Representations*, 2019. [Online]. Available: https://openreview.net/forum?id=SJx63jRqFm 6

[31] S. Chandra, C. Couprie, and I. Kokkinos, "Deep spatio-temporal random fields for efficient video segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8915–8924. 6

[32] R. Gadde, V. Jampani, and P. V. Gehler, "Semantic video cnns through representation warping," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 4453–4462. 6

[33] C. Vondrick, D. Patterson, and D. Ramanan, "Efficiently scaling up crowdsourced video annotation," *International Journal of Computer Vision*, pp. 1–21, 2013, 10.1007/s11263-012-0564-1. [Online]. Available: http://dx.doi.org/10.1007/s11263-012-0564-1 6

[34] V. Badrinarayanan, F. Galasso, and R. Cipolla, "Label propagation in video sequences," in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE, 2010, pp. 3265–3272. 6

[35] Y. Bengio, O. Delalleau, and N. Le Roux, "Label propagation and quadratic criterion," in *Semi-Supervised Learning*, 2006. 6

[36] X. Chen, A. Shrivastava, and A. Gupta, "Neil: Extracting visual knowledge from web data," in *Proceedings of the IEEE International Conference on Computer Vision*, 2013, pp. 1409–1416. 6

[37] S. Tulsiani, T. Zhou, A. A. Efros, and J. Malik, "Multi-view supervision for single-view reconstruction via differentiable ray consistency," *TPAMI*, 2019. 6

[38] Y. Siddiqui, J. Valentin, and M. Nießner, "Viewal: Active learning with viewpoint entropy for semantic segmentation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 9433–9443. 6

[39] S. Gupta, J. Davidson, S. Levine, R. Sukthankar, and J. Malik, "Cognitive mapping and planning for visual navigation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 2616–2625. 6

[40] M. Savva, A. X. Chang, A. Dosovitskiy, T. Funkhouser, and V. Koltun, "MINOS: Multimodal indoor simulator for navigation in complex environments," *arXiv:1712.03931*, 2017. 6

[41] P. Anderson, Q. Wu, D. Teney, J. Bruce, M. Johnson, N. Sünderhauf, I. Reid, S. Gould, and A. van den Hengel, "Vision-and-language navigation: Interpreting visually-grounded navigation instructions in real environments," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 3674–3683. 6

[42] G. Lample and D. S. Chaplot, "Playing FPS games with deep reinforcement learning," in *Thirty-First AAAI Conference on Artificial Intelligence*, 2017. 6

[43] A. Dosovitskiy and V. Koltun, "Learning to act by predicting the future," in *ICLR*, 2017. 6

[44] Y. Wu and Y. Tian, "Training agent for first-person shooter game with actor-critic curriculum learning," in *ICLR*, 2017. 6

[45] D. S. Chaplot and G. Lample, "Arnold: An autonomous agent to play fps games," in *Thirty-First AAAI Conference on Artificial Intelligence*, 2017. 6

[46] P. Mirowski, R. Pascanu, F. Viola, H. Soyer, A. J. Ballard, A. Banino, M. Denil, R. Goroshin, L. Sifre, K. Kavukcuoglu *et al.*, "Learning to navigate in complex environments," *ICLR*, 2017. 6

[47] K. M. Hermann, F. Hill, S. Green, F. Wang, R. Faulkner, H. Soyer, D. Szepesvari, W. Czarnecki, M. Jaderberg, D. Teplyashin *et al.*, "Grounded language learning in a simulated 3d world," *arXiv preprint arXiv:1706.06551*, 2017. 6, 7

[48] D. S. Chaplot, K. M. Sathyendra, R. K. Pasumarthi, D. Rajagopal, and R. Salakhutdinov, "Gated-attention architectures for task-oriented language grounding," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018. 6, 7

[49] Y. Zhu, R. Mottaghi, E. Kolve, J. J. Lim, A. Gupta, L. Fei-Fei, and A. Farhadi, "Target-driven visual navigation in indoor scenes using deep reinforcement learning," in *Robotics and Automation (ICRA), 2017 IEEE International Conference on*. IEEE, 2017, pp. 3357–3364. 6, 7

[50] D. S. Chaplot, R. Salakhutdinov, A. Gupta, and S. Gupta, "Neural topological slam for visual navigation," in *CVPR*, 2020. 6, 7

[51] A. Das, S. Datta, G. Gkioxari, S. Lee, D. Parikh, and D. Batra, "Embodied question answering," in *CVPR*, 2018. 6, 7

[52] D. Gordon, A. Kembhavi, M. Rastegari, J. Redmon, D. Fox, and A. Farhadi, "Iqa: Visual question answering in interactive environments," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 4089–4098. 6, 7

[53] D. S. Chaplot, D. Gandhi, S. Gupta, A. Gupta, and R. Salakhutdinov, "Learning to explore using active neural slam," in *ICLR*, 2020. [Online]. Available: https://openreview.net/forum?id=HklXn1BKDH 7, 9, 10, 12, 13, 14, 15, 16

[54] T. Chen, S. Gupta, and A. Gupta, "Learning exploration policies for navigation," in *International Conference on Learning Representations*, 2019. [Online]. Available: https://openreview.net/forum?id=SyMWn05F7 7, 12, 13, 14, 16

[55] K. Fang, A. Toshev, L. Fei-Fei, and S. Savarese, "Scene memory transformer for embodied agents in long-horizon tasks," in *CVPR*, 2019. 7

[56] M. Savva, A. Kadian, O. Maksymets, Y. Zhao, E. Wijmans, B. Jain, J. Straub, J. Liu, V. Koltun, J. Malik *et al.*, "Habitat: A platform for embodied ai research," in *ICCV*, 2019. 10, 11

[57] F. Xia, A. R. Zamir, Z.-Y. He, A. Sax, J. Malik, and S. Savarese, "Gibson Env: real-world perception for embodied agents," in *Computer Vision and Pattern Recognition (CVPR), 2018 IEEE Conference on*. IEEE, 2018, License: http://svl.stanford.edu/gibson2/assets/GDS_agreement.pdf. 10

[58] A. Chang, A. Dai, T. Funkhouser, M. Halber, M. Niessner, M. Savva, S. Song, A. Zeng, and Y. Zhang, "Matterport3d: Learning from rgb-d data in indoor environments," *International Conference on 3D Vision (3DV)*, 2017, License: http://kaldir.vc.in.tum.de/matterport/MP_TOS.pdf. 10

[59] J. Straub, T. Whelan, L. Ma, Y. Chen, E. Wijmans, S. Green, J. J. Engel, R. Mur-Artal, C. Ren, S. Verma, A. Clarkson, M. Yan, B. Budge, Y. Yan, X. Pan, J. Yon, Y. Zou, K. Leon, N. Carter, J. Briales, T. Gillingham, E. Mueggler, L. Pesqueira, M. Savva, D. Batra, H. M. Strasdat, R. D. Nardi, M. Goesele, S. Lovegrove, and R. Newcombe, "The Replica dataset: A digital replica of indoor spaces," *arXiv preprint arXiv:1906.05797*, 2019, License: https://github.com/facebookresearch/Replica-Dataset/blob/master/LICENSE. 10

[60] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017. 11

[61] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft coco: Common objects in context," in *ECCV*, 2014. 11

[62] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," in *Advances in neural information processing systems*, 2015, pp. 91–99. 11

[63] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2117–2125. 11

[64] Y. Wu, A. Kirillov, F. Massa, W.-Y. Lo, and R. Girshick, "Detectron2," https://github.com/facebookresearch/detectron2, 2019. 11

[65] A. T. Miller, S. Knoop, H. I. Christensen, and P. K. Allen, "Automatic grasp planning using shape primitives," in *2003 IEEE International Conference on Robotics and Automation (Cat. No. 03CH37422)*, vol. 2. IEEE, 2003, pp. 1824–1829. 18

[66] A. Saxena, J. Driemeyer, and A. Y. Ng, "Robotic grasping of novel objects using vision," *The International Journal of Robotics Research*, vol. 27, no. 2, pp. 157–173, 2008. 18

[67] J. Redmon and A. Angelova, "Real-time grasp detection using convolutional neural networks," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2015, pp. 1316–1322. 18

[68] J. M. Romano, K. Hsiao, G. Niemeyer, S. Chitta, and K. J. Kuchenbecker, "Human-inspired robotic grasp control with tactile sensing," *IEEE Transactions on Robotics*, vol. 27, no. 6, pp. 1067–1079, 2011. 18

[69] Y. Bekiroglu, J. Laaksonen, J. A. Jorgensen, V. Kyrki, and D. Kragic, "Assessing grasp stability based on learning and haptic data," *IEEE Transactions on Robotics*, vol. 27, no. 3, pp. 616–629, 2011. 18, 21

[70] Y. Bekiroglu, R. Detry, and D. Kragic, "Learning tactile characterizations of object- and pose-specific grasps," in *2011 IEEE/RSJ international conference on Intelligent Robots and Systems*. IEEE, 2011, pp. 1554–1560. 18, 21

[71] Z. Su, K. Hausman, Y. Chebotar, A. Molchanov, G. E. Loeb, G. S. Sukhatme, and S. Schaal, "Force estimation and slip detection/classification for grip control using a biomimetic tactile sensor," in *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*. IEEE, 2015, pp. 297–303. 18

[72] R. Calandra, A. Owens, D. Jayaraman, J. Lin, W. Yuan, J. Malik, E. H. Adelson, and S. Levine, "More than a feeling: Learning to grasp and regrasp using vision and touch," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 3300–3307, 2018. 18, 21, 45, 46, 51

[73] W. Yuan, S. Dong, and E. H. Adelson, "Gelsight: High-resolution robot tactile sensors for estimating geometry and force," *Sensors*, vol. 17, no. 12, p. 2762, 2017. 19, 21, 37

[74] K. Cao, C. Wei, A. Gaidon, N. Arechiga, and T. Ma, "Learning imbalanced datasets with label-distribution-aware margin loss," *Advances in neural information processing systems*, vol. 32, 2019. 19, 27

[75] A. Sahbani, S. El-Khoury, and P. Bidaud, "An overview of 3d object grasp synthesis algorithms," *Robotics and Autonomous Systems*, vol. 60, no. 3, pp. 326–336, 2012. 20

[76] J. Bohg, A. Morales, T. Asfour, and D. Kragic, "Data-driven grasp synthesis—a survey," *IEEE Transactions on Robotics*, vol. 30, no. 2, pp. 289–309, 2013. 20, 21

[77] V.-D. Nguyen, "Constructing stable grasps in 3d," in *Proceedings. 1987 IEEE International Conference on Robotics and Automation*, vol. 4. IEEE, 1987, pp. 234–239. 20

[78] K. B. Shimoga, "Robot grasp synthesis algorithms: A survey," *The International Journal of Robotics Research*, vol. 15, no. 3, pp. 230–266, 1996. 20

[79] J.-W. Li, H. Liu, and H.-G. Cai, "On computing three-finger force-closure grasps of 2-d and 3-d objects," *IEEE Transactions on Robotics and Automation*, vol. 19, no. 1, pp. 155–161, 2003. 20

[80] C. Goldfeder, M. Ciocarlie, J. Peretzman, H. Dang, and P. K. Allen, "Data-driven grasping with partial sensor data," in *2009 IEEE/RSJ international conference on intelligent robots and systems*. IEEE, 2009, pp. 1278–1283. 21

[81] Y. Bekiroglu, D. Song, L. Wang, and D. Kragic, "A probabilistic framework for task-oriented grasp stability assessment," in *2013 IEEE International Conference on Robotics and Automation*. IEEE, 2013, pp. 3040–3047. 21

[82] S. Levine, P. Pastor, A. Krizhevsky, J. Ibarz, and D. Quillen, "Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection," *The International journal of robotics research*, vol. 37, no. 4-5, pp. 421–436, 2018. 21

[83] L. Pinto and A. Gupta, "Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours," in *2016 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2016, pp. 3406–3413. 21, 22

[84] D. Guo, F. Sun, T. Kong, and H. Liu, "Deep vision networks for real-time robotic grasp detection," *International Journal of Advanced Robotic Systems*, vol. 14, no. 1, p. 1729881416682706, 2016. 21

[85] J. Li, S. Dong, and E. Adelson, "Slip detection with combined tactile and visual information," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 7772–7777. 21, 22, 26

[86] C. Chi, X. Sun, N. Xue, T. Li, and C. Liu, "Recent progress in technologies for tactile sensors," *Sensors*, vol. 18, no. 4, 2018. [Online]. Available: https://www.mdpi.com/1424-8220/18/4/948 21

[87] E. Donlon, S. Dong, M. Liu, J. Li, E. Adelson, and A. Rodriguez, "Gelslim: A high-resolution, compact, robust, and calibrated tactile-sensing finger," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 1927–1934. 21

[88] M. Lambeta, P.-W. Chou, S. Tian, B. Yang, B. Maloon, V. R. Most, D. Stroud, R. Santos, A. Byagowi, G. Kammerer *et al.*, "Digit: A novel design for a low-cost compact high-resolution tactile sensor with application to in-hand manipulation," *IEEE Robotics and Automation Letters*, vol. 5, no. 3, pp. 3838–3845, 2020. 21

[89] M. Bauza, O. Canal, and A. Rodriguez, "Tactile mapping and localization from high-resolution tactile imprints," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 3811–3817. 21

[90] R. Li, R. Platt, W. Yuan, A. ten Pas, N. Roscup, M. A. Srinivasan, and E. Adelson, "Localization and manipulation of small parts using gelsight tactile sensing," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2014, pp. 3988–3993. 21

[91] X. Xu, Y. Liu, W. Chen, H. Yuan, H. Wang, J. Xu, R. Chen, and L. Yi, "Enhancing generalizable 6d pose tracking of an in-hand object with tactile sensing," *arXiv preprint arXiv:2210.04026*, 2022. 21

[92] S. Dong, W. Yuan, and E. H. Adelson, "Improved gelsight tactile sensor for measuring geometry and slip," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 137–144. 21

[93] Y. Chen, C. Prepscius, D. Lee, and D. D. Lee, "Tactile velocity estimation for controlled in-grasp sliding," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 1614–1621, 2021. 21

[94] C. Wang, S. Wang, B. Romero, F. Veiga, and E. Adelson, "Swingbot: Learning physical features from in-hand tactile exploration for dynamic swing-up manipulation," in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 5633–5640. 21, 23

[95] M. Madry, L. Bo, D. Kragic, and D. Fox, "St-hmp: Unsupervised spatio-temporal feature learning for tactile data," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2014, pp. 2262–2269. 21

[96] R. Calandra, A. Owens, M. Upadhyaya, W. Yuan, J. Lin, E. H. Adelson, and S. Levine, "The feeling of success: Does touch sensing help predict grasp outcomes?" in *Conference on Robot Learning*. PMLR, 2017, pp. 314–323. 21, 22

[97] F. Veiga, J. Peters, and T. Hermans, "Grip stabilization of novel objects using slip prediction," *IEEE transactions on haptics*, vol. 11, no. 4, pp. 531–542, 2018. 21

[98] Y. Zhang, Z. Kan, Y. A. Tse, Y. Yang, and M. Y. Wang, "Fingervision tactile sensor design and slip detection using convolutional lstm network," *arXiv preprint arXiv:1810.02653*, 2018. 21, 22

[99] B. S. Zapata-Impata, P. Gil, and F. Torres, "Learning spatio temporal tactile features with a convlstm for the direction of slip detection," *Sensors*, vol. 19, no. 3, p. 523, 2019. 21

[100] Y. Chebotar, K. Hausman, Z. Su, A. Molchanov, O. Kroemer, G. Sukhatme, and S. Schaal, "Bigs: Biotac grasp stability dataset," in *ICRA 2016 Workshop on Grasping and Manipulation Datasets*, 2016. 22, 23

[101] N. Wettels, V. J. Santos, R. S. Johansson, and G. E. Loeb, "Biomimetic tactile sensor array," *Advanced Robotics*, vol. 22, no. 8, pp. 829–849, 2008. 22

[102] A. Murali, Y. Li, D. Gandhi, and A. Gupta, "Learning to grasp without seeing," in *International Symposium on Experimental Robotics*. Springer, 2018, pp. 375–386. 22

[103] T. Wang, C. Yang, F. Kirchner, P. Du, F. Sun, and B. Fang, "Multimodal grasp data set: A novel visual–tactile data set for robotic manipulation," *International Journal of Advanced Robotic Systems*, vol. 16, 2019. 22

[104] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778. 26, 48

[105] N. Japkowicz and S. Stephen, "The class imbalance problem: A systematic study," *Intelligent data analysis*, vol. 6, no. 5, pp. 429–449, 2002. 27

[106] M. Buda, A. Maki, and M. A. Mazurowski, "A systematic study of the class imbalance problem in convolutional neural networks," *Neural Networks*, vol. 106, pp. 249–259, 2018. 27

[107] J. Grannen, P. Sundaresan, B. Thananjeyan, J. Ichnowski, A. Balakrishna, V. Viswanath, M. Laskey, J. Gonzalez, and K. Goldberg, "Untangling dense knots by learning task-relevant keypoints," in *Conference on Robot Learning*. PMLR, 2021, pp. 782–800. 31

[108] S. Pirozzi and C. Natale, "Tactile-based manipulation of wires for switchgear assembly," *IEEE/ASME Transactions on Mechatronics*, vol. 23, no. 6, pp. 2650–2661, 2018. 31

[109] S. Jin, W. Lian, C. Wang, M. Tomizuka, and S. Schaal, "Robotic cable routing with spatial representation," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 5687–5694, 2022. 31, 33

[110] K. Galassi and G. Palli, "Robotic wires manipulation for switchgear cabling and wiring harness manufacturing," in *2021 4th IEEE International Conference on Industrial Cyber-Physical Systems (ICPS)*, 2021, pp. 531–536. 31

[111] "https://www.nist.gov/el/intelligent-systems-division-73500/iros-2020-robotic-grasping-and-manipulation-competition." [Online]. Available: https://www.nist.gov/el/intelligent-systems-division-73500/iros-2020-robotic-grasping-and-manipulation-competition 31, 34

[112] Y. She, S. Wang, S. Dong, N. Sunil, A. Rodriguez, and E. Adelson, "Cable manipulation with a tactile-reactive gripper," *The International Journal of Robotics Research*, vol. 40, no. 12-14, pp. 1385–1401, 2021. 31

[113] H. Nakagaki, K. Kitagaki, T. Ogasawara, and H. Tsukune, "Study of deformation and insertion tasks of a flexible wire," in *Proceedings of International Conference on Robotics and Automation*, vol. 3, 1997, pp. 2397–2402 vol.3. 33

[114] C. Wang, Y. Zhang, X. Zhang, Z. Wu, X. Zhu, S. Jin, T. Tang, and M. Tomizuka, "Offline-online learning of deformation model for cable manipulation with graph neural networks," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 5544–5551, apr 2022. 33

[115] M. Yan, Y. Zhu, N. Jin, and J. Bohg, "Self-supervised learning of state estimation for manipulating deformable linear objects," *IEEE robotics and automation letters*, vol. 5, no. 2, pp. 2372–2379, 2020. 33

[116] A. Nair, D. Chen, P. Agrawal, P. Isola, P. Abbeel, J. Malik, and S. Levine, "Combining self-supervised learning and imitation for vision-based rope manipulation," in *2017 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2017, pp. 2146–2153. 33

[117] F. Süberkrüb, R. Laezza, and Y. Karayiannidis, "Feel the tension: Manipulation of deformable linear objects in environments with fixtures using force information," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2022, pp. 11 216–11 222. 33

[118] J. Zhu, B. Navarro, R. Passama, P. Fraisse, A. Crosnier, and A. Cherubini, "Robotic manipulation planning for shaping deformable linear objects with environmental contacts," *IEEE Robotics and Automation Letters*, vol. 5, no. 1, pp. 16–23, 2020. 33

[119] G. A. Waltersson, R. Laezza, and Y. Karayiannidis, "Planning and control for cable-routing with dual-arm robot," in *2022 International Conference on Robotics and Automation (ICRA)*, 2022, pp. 1046–1052. 33

[120] D. De Gregorio, R. Zanella, G. Palli, S. Pirozzi, and C. Melchiorri, "Integration of robotic vision and tactile sensing for wire-terminal insertion tasks," *IEEE Transactions on Automation Science and Engineering*, vol. 16, no. 2, pp. 585–598, 2019. 34

[121] T. Migimatsu, W. Lian, J. Bohg, and S. Schaal, "Symbolic state estimation with predicates for contact-rich manipulation tasks," in *2022 International Conference on Robotics and Automation (ICRA)*, 2022, pp. 1702–1709. 34

[122] J. Schulman, A. Lee, J. Ho, and P. Abbeel, "Tracking deformable objects with point clouds," in *2013 IEEE International Conference on Robotics and Automation*. IEEE, 2013, pp. 1130–1137. 36

[123] A. Myronenko and X. Song, "Point set registration: Coherent point drift," *IEEE transactions on pattern analysis and machine intelligence*, vol. 32, no. 12, pp. 2262–2275, 2010. 36

[124] A. Wilson, H. Jiang, W. Lian, and W. Yuan, "Cable routing and assembly using tactile-driven motion primitives," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 10 408–10 414. 45, 46, 49, 50

[125] J. Lin, R. Calandra, and S. Levine, "Learning to identify object instances by touch: Tactile recognition via multimodal matching," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 3644–3650. 45

[126] J. Bimbo, L. D. Seneviratne, K. Althoefer, and H. Liu, "Combining touch and vision for the estimation of an object's pose during manipulation," in *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2013, pp. 4021–4026. 45

[127] A. N. Chaudhury, T. Man, W. Yuan, and C. G. Atkeson, "Using collocated vision and tactile sensors for visual servoing and localization," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 3427–3434, 2022. 46

[128] M. A. Lee, Y. Zhu, K. Srinivasan, P. Shah, S. Savarese, L. Fei-Fei, A. Garg, and J. Bohg, "Making sense of vision and touch: Self-supervised learning of multimodal representations for contact-rich tasks," in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 8943–8950. 46

[129] C. Yang and N. F. Lepora, "Object exploration using vision and active touch," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 6363–6370. 46

[130] S. Kanitkar, H. Jiang, and W. Yuanl, "Poseit: A visual-tactile dataset of holding poses for grasp stability analysis," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 71–78. 46, 47, 48, 51

[131] L. Fu, H. Huang, L. Berscheid, H. Li, K. Goldberg, and S. 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)*, 2023, pp. 10 380–10 386. 46

[132] R. Okumura, N. Nishio, and T. Taniguchi, "Tactile-sensitive newtonianvae for high-accuracy industrial connector insertion," in *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2022, pp. 4625–4631. 46

[133] Y. Chen, M. Van der Merwe, A. Sipos, and N. Fazeli, "Visuo-tactile transformers for manipulation," in *6th Annual Conference on Robot Learning*, 2022. 46

[134] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," in *International Conference on Learning Representations*. 46, 47, 48

[135] S. Ross, G. Gordon, and D. 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, pp. 627–635. 46

[136] S. Sharma, G. Tewolde, and J. Kwon, "Behavioral cloning for lateral motion control of autonomous vehicles using deep learning," in *2018 IEEE International Conference on Electro/Information Technology (EIT)*. IEEE, 2018, pp. 0228–0233. 46

[137] T. V. Samak, C. V. Samak, and S. Kandhasamy, "Robust behavioral cloning for autonomous vehicles using end-to-end imitation learning," *SAE International Journal of Connected and Automated Vehicles*, vol. 4, no. 12-04-03-0023, 2021. 46

[138] T. Zhang, Z. McCarthy, O. Jow, D. Lee, X. Chen, K. Goldberg, and P. Abbeel, "Deep imitation learning for complex manipulation tasks from virtual reality teleoperation," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 5628–5635. 46

[139] S. Young, D. Gandhi, S. Tulsiani, A. Gupta, P. Abbeel, and L. Pinto, "Visual imitation made easy," in *Conference on Robot Learning*. PMLR, 2021, pp. 1992–2005. 46

[140] Y. Zhu, Z. Wang, J. Merel, A. A. Rusu, T. Erez, S. Cabi, S. Tunyasuvunakool, J. Kramár, R. Hadsell, N. de Freitas, and N. Heess, "Reinforcement and imitation learning for diverse visuomotor skills," *CoRR*, vol. abs/1802.09564, 2018. [Online]. Available: http://arxiv.org/abs/1802.09564 46

[141] S. P. Arunachalam, S. Silwal, B. Evans, and L. Pinto, "Dexterous imitation made easy: A learning-based framework for efficient dexterous manipulation," *arXiv preprint arXiv:2203.13251*, 2022. 46

[142] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robotics and autonomous systems*, vol. 57, no. 5, pp. 469–483, 2009. 46

[143] S. Nair, A. Rajeswaran, V. Kumar, C. Finn, and A. Gupta, "R3m: A universal visual representation for robot manipulation," in *6th Annual Conference on Robot Learning*. 48

[144] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, "Imagenet large scale visual recognition challenge," *International journal of computer vision*, vol. 115, pp. 211–252, 2015. 48