Active Keyframe Learning (AKL): Learning Interaction and Constraint Keyframes from a Single Demonstration of a Task

by

Thavishi Illandara

B.Sc.Eng (Hons), University of Moratuwa (2017)

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Science in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May 2022

Author . . . .
Department of Electrical Engineering and Computer Science
May 13, 2022

Certified by. . . . .
Julie A. Shah
Professor of Aeronautics and Astronautics
Thesis Supervisor

Accepted by . . . .
Leslie A. Kolodziejski
Professor of Electrical Engineering and Computer Science
Chair, Department Committee on Graduate Students

Active Keyframe Learning (AKL): Learning Interaction and Constraint Keyframes from a Single Demonstration of a Task

by

Thavishi Illandara

Submitted to the Department of Electrical Engineering and Computer Science
on May 13, 2022, in partial fulfillment of the
requirements for the degree of
Master of Science in Electrical Engineering and Computer Science

Abstract

Although recent advances in robotics enable the automation of manual tasks in manufacturing, integrating robots into a factory remains time and resource intensive, as it requires conventional robot programming and robot experts. In order to increase the feasibility of robot integration into industrial processes, the programming of robots must be easily accessible to domain experts with little to no experience in robotics. In this thesis, we present Active Keyframe Learning (AKL) for learning the task specification as an ordered sequence of keyframes to capture the physical interactions and geometric constraints from a single demonstration of a task given by a non-expert. We learn the least restrictive task specification that maximizes the flexibility given to a motion planner by learning the human intent for demonstrated constrained motion online and performing interaction-based and constraint-based segmentation offline. We conduct a user study to evaluate the keyframe, pose, constraint accuracies, workload, and teaching efficiency of AKL against two state-of-the-art techniques in keyframe and constraint learning and demonstrate the significant benefits of utilizing AKL to teach tasks to robots.

Thesis Supervisor: Julie A. Shah
Title: Professor of Aeronautics and Astronautics

Acknowledgments

First, I would like to thank my advisor, Professor Julie Shah, for her immense support and guidance during this research. Her valuable insights, from formulating exciting research questions to maintaining a healthy work-life balance, supported me in many ways. I would also like to extend my sincerest gratitude to my collaborators, Nadia Figueroa, whose amazing skills at playing with robots proved invaluable for the success of this thesis, and the MEAU team, William Nguyen, Shinsuke Kawasaki, James Knauer, and Keita Kubo. I would also like to thank my labmates at the Interactive Robotics Group for all their support, ranging from asking tough research questions to organizing virtual movie nights during the pandemic.

I am profoundly grateful to all my friends here at MIT and the Sri Lankan community in Boston for their support and guidance throughout the years. Along those lines, I would like to thank my fellow Dota2 addicts, especially Nethmal, Shenan, and Shanalee, for our fun, relaxing gaming sessions. I would also like to thank Ranuka for his constant words of encouragement and support in ensuring I remain a functioning human when meeting deadlines.

I would like to thank my brother, Gamith, and his wife, Naamini, for all their support and encouragement. Most importantly, I am grateful for my parents, Rathna Kumara and Sepalika Illandara, without whose unwavering support I would never be here. This thesis is for them.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Recent advances in robotics enable the automation of traditionally manual tasks in manufacturing. However, integrating robots into a factory using conventional robot programming methods remains time and resource intensive. Even small changes to the task will require reprogramming and can result in high reintegration costs [59]. Learning from demonstration (LfD) has been explored as a potential solution, as it can facilitate domain experts with minimal robot programming experience to teach robots manual tasks efficiently [56]. We are interested in leveraging LfD techniques to increase robots' ease of use, allowing non-experts to teach multistep tasks.

Research on LfD has primarily focused on learning a cost or reward function that can be optimized to produce the intended behavior [2, 19, 23, 26]. Although this enables generalizability to dynamic environments, these methods tend to be data-intensive. Furthermore, they can only model soft constraints, whereas tasks like carrying a mug upright require learning hard geometric constraints for successful execution. Another branch of LfD learns a policy that can generate the desired behavior [62]. These methods learn compact representations of skills but may fail at learning complex tasks composed of a sequence of subtasks and hard constraints. LfD work has also explored learning a plan at a high level of task abstraction as a primitive sequence [3, 42, 53] or a primitive hierarchy [44, 45, 60]. These methods perform well in multistep tasks and long-horizon planning. However, they cannot handle continuous action spaces. Our research will focus on learning a high-level plan as an ordered sequence of steps or keyframes, a concept introduced by Akgun et al. [3], that captures the physical interactions and local geometric constraints of a task. Learning keyframes will allow us to learn a compact agent-independent representation of a multistep task that can be provided to an existing motion planner equipped with collision avoidance for task execution.

Learning a sequenced task plan from a continuous demonstration requires trajectory segmentation. Perez-D'Arpino and Shah [53] use end-effector poses explicitly defined by the user to learn a plan and a set of geometric constraints. However, in this approach, the quality of the sequence learned hinges on the operator's knowledge of what constitutes an optimal set of keyframes. For instance, if the teacher defines too few or too many keyframes, the system may learn incorrect or over-constrained task specifications. On the other hand, automated segmentation techniques can be relatively robust to user expertise and allow users to provide more intuitive trajectory demonstrations. General approaches for automated segmentation include the use of probabilistic models [19, 27, 33, 35, 49], clustering algorithms [36, 46, 48], inverse reinforcement learning [43, 50], and task characteristics or events [30, 32, 38]. These methods largely focus on demonstration segmentation given multiple demonstrations and can be sensitive to model parameters. Learning from a single demonstration has been explored in segmentation literature [41, 42, 44, 53] as means of mitigating the time and resource intensive nature of providing multiple demonstrations as input. However, learning constraints from a single demonstration often results in under-, or over-constrained task specifications due to inadequate data and unintentional human errors [42, 53]. In our work, we explore trajectory segmentation of a single demonstration based on physical interactions with objects and move-in-line constraints, introduced in [53], to learn interaction and constraint keyframes defined at the boundaries of these segments. To reduce over-constraining task specifications when learning from a single demonstration, we take an active learning approach to leverage the teacher's intent for demonstrated constraint motion.

Active learning in LfD allows learners to elicit the teacher's feedback, preference, or intent to improve task representations learned, such as refining the learned reward or cost function [8, 9, 17, 58, 65], updating the belief over learned task specifications [61], and to more efficiently learn task constraints [31], semantic constraints [63], and behavioral constraints [47]. However, to teach a robot naturally and effectively, a human needs to have an accurate mental model of the robot [54, 55]. Mueller et al. explore the use of Robot Behaviour Counterfactuals (RBCs) and Behavioral Verification Indicators (BVIs) in their active learning framework, which improves the teacher's understanding of each constraint's impact on task execution [47]. Following guidelines laid by Cakmak et al., in [10, 11, 12], for human-robot interactive communication, we design an online active learning technique that leverages human feedback to improve the accuracy of the constraints learned offline. Additionally, we enable learner feedback to the teacher as a suggestion to improve their teaching of unconstrained motion, taking a step towards improving the teacher's mental model of the learner.

This thesis proposes Active Keyframe Learning (AKL), a proof-of-concept system for learning physical interactions and move-in-line constraints of a task, from a single demonstration, as an ordered sequence of keyframes. Our contribution to this thesis is threefold. First, as multiple keyframe-encoded specifications can exist for a given task, we prove that the specification with the least number of keyframes, which we call the least restrictive task specification, maximizes the flexibility given to a motion planner during task execution. Second, we design an interactive demonstration framework that allows human-robot communication, based on online constraint inference, to learn human intent for the demonstrated constrained motion and guide the teacher to provide better demonstrations. Third, we develop an offline keyframe learning algorithm that performs interaction-based segmentation and constraint-based segmentation augmented with human feedback to learn the least restrictive task specification. We evaluate AKL's performance and usability against two baseline methods for three multistep manipulation tasks using a within-subjects study and show significant keyframe accuracy and teaching efficiency improvements.

1.1 Task Specification Learning in LfD

Learning high-level task plans as a sequence or hierarchy, or primitives requires trajectory segmentation of a continuous trajectory of the task. Some segmentation approaches perform interaction-based segmentation by learning action segments through task events or task space characteristics [30, 32, 38]. However, their learned task specifications disallow the representation of motion-level geometric constraints by limiting the specification to interaction-based primitives. Another class of segmentation research is segmentation based on Gaussian mixture models primarily employed to capture the variability of motion trajectory features and estimate a set of segmentation points in a demonstration [13, 39, 40, 52]. Although these methods efficiently encode motion primitives, it is unclear how the geometric transformations required for the learned motion primitives to adapt to task variations, such as changes in object positions, guarantee the satisfaction of the motion primitive constraints.

Hidden Markov Models (HMMs) and statistical model-based change point detection algorithms, inspired by the changepoint detection introduced by Fearnhead and Liu [20, 21, 22], are utilized to segment demonstrations based on changes in specified models or latent variables [27, 33, 35, 49]. However, such segmentation techniques are limited to inference over parametrized models and cannot recognize trajectory segments that cannot be modeled, such as arbitrary unconstrained motion. Research in Inverse Reinforcement Learning (IRL) such as Bayesian Nonparametric IRL (BN-IRL) [43], and Constraint-based BN-IRL (CBN-IRL) [50] can learn local properties and constraints of a task by partitioning demonstrations into subtasks. Nevertheless, due to the unconstrained nature of the latent assignment variable and information losses that arise in these techniques, they cannot guarantee the precise delineation of tasks into constrained and unconstrained subtasks.

Modeling global and local task constraints are often essential to successfully execute a task. To this end, LfD research has explored global task space constraints [5, 18], safety constraints [67], position and force constraints [64], conceptual constraints [46, 48], shared constraints across tasks [14], high dimensional parametric constraints [1, 15, 16] and geometric constraints [24, 37, 53, 66]. Among these techniques, Task space regions (TSR) [7] have been widely adopted to represent geometric constraints [41, 42, 44, 53, 66] learned through trajectory segmentation of demonstrations based on constraints. Liu et al. perform Total Variation Denoising (TVD) [57] on constraint model fitting error ratios to learn constraint segments represented as TSRs. However, the accuracy of the learned constraints is sensitive to predefined parameters and is dependent on the assumption that the teacher will only demonstrate constrained motion when crucial to the task's success. Keyframe demonstrations, a concept introduced by Akgun et al. [3], are ordered sparse sets of sequential poses that can function as a compact task representation independent of an agent's kinematics [53]. These demonstrations have been extensively utilized in LfD literature to eliminate noisy undesirable motion [4, 24, 28, 37, 53, 66] and have been extended to include geometric task constraints [24, 37, 53, 66]. Keyframe demonstrations allow user-defined task segmentation; however, continuous trajectory demonstrations retain critical speed and timing information and appear more intuitive to novice teachers. Furthermore, the demonstrator's ability to provide an accurate set of keyframes impacts the quality of the task representation learned.

Active learning, a framework that allows robots to elicit human feedback for improved learning, has been utilized in robotics to create more intuitive teaching interfaces [10] and reduce the time and effort spent teaching new skills [12]. This learning approach has been explored in the context of updating its belief over reward functions [17], inferring human preferences for a dynamical system's behavior [9, 8, 58], learning user preferences for complex task specifications [65], updating belief over LTL task specifications [61]. Research in constraint learning has likewise leveraged active learning to learn task constraints [31], semantic constraints [63], and behavioral constraints [47] more efficiently. Although the aforementioned active learning frameworks in LfD allow novice users to teach robots in an increasingly natural and time-efficient manner, the impact of the given human feedback on the robot's mental model and the learned model remains unknown to the teacher. To improve the naturalistic nature and effectiveness of robot teaching interfaces, the human teacher requires an accurate mental model of the robot [54, 55]. Mueller et al. explore using Robot Behaviour Counterfactuals (RBCs) and Behavioral Verification Indicators (BVIs) to visualize the robot's expected behavior and the task execution success for the new model with changes in task constraints [47]. Although this can improve the human mental model of the robot, these visualizations are provided after the demonstrations and do not explicitly show the user how their teaching techniques can be improved.

In this thesis, we present Active Keyframe Learning (AKL), a proof-of-concept system for learning the least restrictive task specification encoded by an ordered sequence of TSR keyframes. Similar to [32, 38], we learn object interactions utilizing the demonstrated object and grasp poses; however, we also learn move-in-line constraints, introduced in [53], to capture both physical interactions and constraints in the task. To ensure our learned task plan can adapt to positional variances of objects, we leverage the object interactions inferred from demonstrated grasp poses to ground learned position and geometric constraints to relevant objects in the environment. We design a novel online constraint detection technique that can distinguish between constrained motion segments following an underlying straight-line motion model and arbitrary unmodeled unconstrained motion segments. To ensure precise delineation of tasks into constrained and unconstrained subtasks, we augment this online model with an offline counterpart consisting of a fitting model to force segmentation at unconstrained and constrained motion boundaries. Additionally, this offline technique performs interaction segmentation and a combination process to learn the least number of keyframes required to capture all the physical interactions and move-in-line constraints of the task.

We utilize a single continuous trajectory demonstration of a task as input to make the teaching interface more intuitive and independent of the teacher's ability to provide the correct set of keyframes. However, learning from a single demonstration often requires the assumption that the teacher has a clear understanding that constrained motion should be displayed only when essential to the task. In our work, we leverage human feedback to clarify the validity of demonstrated straight-line motion to eliminate this assumption effectively. Human feedback is obtained through interactive communication, designed based on the guidelines presented by Cakmak et al. [10, 11, 12], between the teacher and robot during the demonstration. Additionally, we allow the robot to provide suggestions to the teacher on improving their teaching efficiency, further increasing the teacher's understanding of the learner's mental model. To the best of our knowledge, we believe this is the first work that combines online constraint detection, online human feedback for active learning, and online suggestions for teaching improvement in the context of human-robot interaction.

1.2 Active Keyframe Learning

Active Keyframe Learning learns a task represented as the least restrictive ordered sequence of keyframes through a single interactive demonstration. Here, the least restrictive ordered sequence of keyframes refers to the task specification that results in a successful execution trace while allowing the most flexibility to the motion planner. During a demonstration, the learner maintains a continuous record of the end-effector poses, the gripper state, and the objects' positions in the scene. Furthermore, the learner records the teacher's feedback on the validity of the detected constrained motions given during teaching. This demonstration data is processed to learn the task specification as a sequence of ordered keyframes encoded as Task Space Regions (TSR) [7] augmented with gripper and constraint state variables.

In our work, we learn three types of keyframes: (1) interaction keyframes that capture the physical interactions between the end-effector and objects during task execution, (2) constraint keyframes that encode the move-in-line constraints of the task, and (3) combined keyframes that simultaneously represent an interaction and a constraint boundary that coincides in Cartesian space. There can be multiple ordered sequences of such keyframes for a given task that will generate correct execution traces when provided to a motion planner. We theorize that the specification with the least number of keyframes, which we call the least restrictive task specification, maximizes the flexibility given to the motion planner and design AKL to learn this specification.

AKL learns keyframes in two stages: (1) the interactive demonstration and (2) offline keyframe learning. The interactive demonstration stage leverages active learning to understand human intent behind demonstrated straight-line motion and provides suggestions to the teacher to improve their mental model of the learner. It consists of an online constraint detector and interactive communication between the robot and the teacher. During the demonstration, the online constraint detector observes the end-effector poses and infers the latent binary state variable that denotes the constrained nature of the current trajectory. When the detector infers the presence of a constrained region, interactive communication begins with the learner's query about the correctness of the detected latent state, followed by the teacher's answer, and ends with a suggestion to the teacher on improving their teaching technique. After the demonstration is complete, the offline keyframe learning algorithm receives the demonstration data and human feedback recorded during the demonstration and performs interaction-based and constraint-based segmentation. Next, the offline algorithm utilizes the human intent captured as feedback to remove incorrect constraints learned from the demonstration data reducing the over-constrained nature of the learned task specification. Finally, keyframes are extracted from the interaction and constraint segments to create the least restrictive task specification for the demonstrated task.

1.3 Performance Evaluation

We evaluated the performance, in terms of success rates, keyframe accuracy, pose accuracy, constraint accuracy, teaching workload, usability, and teaching efficiency, of our proposed framework, Active Keyframe Learning (AKL), against two baseline methods, keyframe demonstration (KD) [3, 53] and articulate constraints learning approach, from [42], augmented with interaction learning (mACL). To gather data, we conducted a 12 participant within-subject study with three multistep tasks: (1) a pick and place task with no move-in-line constraints, (2) an inspection task with a single move-in-line constraint given in the task instructions, and (3) an assembly task with a single move-in-line constraint that is not given in the task instructions given to participants. Each participant performed three tasks for each teaching mode, answered the NASA TLX questionnaire after each task, and rated the ten statements of the System Usability Scale (SUS) [5] on a 7-point Likert scale for each teaching interface. The collected data were analyzed using linear mixed-effects model tests and the Wilcoxon Rank-Sum test.

We found that AKL displayed the highest success rates in learning the least restrictive task specification and significantly improved the keyframe and pose accuracy. Although the linear mixed-effects model analysis results for constraint accuracy were insignificant, fifty percent of task specifications learned using AKL had less than 2.18 cm of constraint length errors, whereas that for mACL spanned 23.65 cm. Additionally, the total number of constraint errors for AKL (19 errors) was much lower than for mACL (46 errors), suggesting an increase in constraint accuracy. We learned that 73.7% of the constraint errors for AKL were due to incorrect human feedback emphasizing the ability to further improve constraint accuracy by increasing the feedback correctness. Furthermore, we found that AKL could prevent 51.2% of constraint errors through human feedback highlighting the positive impact of our proposed interactive demonstration framework on constraint accuracy. Our findings on workload and usability suggested that AKL was more userfriendly and resulted in a significantly lower workload than KD. However, AKL showed increased workload and decreased usability scores compared to mACL. Although the interactive communication aspect of AKL was found to increase workload and reduce usability, the overall teaching efficiency of AKL was significantly higher than KD and mACL, demonstrating the significant benefit of using AKL to teach tasks to robots.

1.4 Contributions and Future Directions

In this thesis, we present Active Keyframe Learning (AKL), a proof-of-concept system for learning the least restrictive task specification, encoded by an ordered sequence of keyframes, that captures the physical interactions and move-in-line constraints of a task from a single demonstration. Our contribution to this thesis is threefold. First, we prove that the specification with the least number of keyframes, which we call the least restrictive task specification, maximizes the flexibility given to a motion planner during task execution. Second, we design an interactive demonstration framework that performs online constraint inference to initiate human-robot communication that learns human intent for the demonstrated constrained motion and provides suggestions to improve teaching. Third, we develop an offline keyframe learning algorithm that performs interaction-based segmentation and constraint-based segmentation augmented with human feedback to learn the least restrictive task specification. We evaluate AKL against two state-of-the-art techniques in keyframe and constraint learning and demonstrate the significant benefit of utilizing AKL to teach tasks to robots.

Currently, this thesis is limited to learning physical interactions between task objects and the robot, essential for the task's success. Thus an extension to AKL would be learning non-physical interactions or complex physical interactions that cannot be inferred through gripper state observations. For example, interactions such as holding a mug under a coffee machine until full or screwing in a nail are complex interactions requiring richer keyframe definitions and robust visual inference.

Another future work direction would be extending the online constraint detector to infer additional hard geometric constraints such as orientation constraints or revolute constraints. As constraint inference utilizes fitting errors of a constraint model, theoretically, it can be extended to other parametrized constraints by having several online constraint detectors in parallel, one for each constraint. However, it will be essential to examine the impact of the parallelization of multiple inference problems on the real-time nature of queries. Furthermore, the interactive communication dialogue must be redesigned to accommodate the additional queries, with particular attention given to the frequency of querying.

Although our work employs learner feedback as suggestions to the teacher to improve their demonstration technique during unconstrained motion, we assume that the learner is unaware of its impact on the teacher's mental model of the learner. However, it would be interesting to learn about this impact and understand how it would affect the demonstrations provided by the teacher to improve interactive communication. Therefore, an interesting future direction for our work is leveraging the teacher's mental model of the learner to inform human-robot communication.

Chapter 2

Related Work

2.1 Trajectory Segmentation and Constraint Learning

Techniques for learning high-level task plans as a sequence or hierarchy of primitives often involve trajectory segmentation to delineate task primitives from a continuous trajectory of the task. Some segmentation approaches learn action segments through task events or task space characteristics. Kyrarini et al. and Huang et al. utilize object and grasp poses to infer actions such as *"grasp action"*, *"release action"* and *"move action"* [32, 38]. Such inferred actions split the task into subtasks creating a task representation as a sequence of executable subtasks or primitives. Hasan et al. employ predefined task space partitions to segment human demonstrations [30]. It adopts a task space representation modeled by rows of objects and gaps between these rows of objects to discretize the action space into two primitive actions, *"moving an object"* and *"going through a gap"*. These actions segment the trajectory and serve as hierarchical high-level plan nodes, where leaf nodes describe low-level executable plans. Although [30, 32, 38] learn interactions with objects and the task space, their task specifications are limited to interaction-based primitives disallowing the representation of motion level geometric constraints such as *"move in a straight line"*. Our work proposes a task specification that describes object interactions, such as grasp and place, and geometric constraints on the motion, such as move-in-line. Similar to [38, 32], we learn object interactions utilizing the demonstrated object and grasp poses.

Gaussian mixture models (GMMs) are helpful statistical tools to capture the variability of motion trajectory features and to estimate a set of segmentation points in a demonstration [13, 39, 40, 52]. Lee et al. explore automatic segmentation of a multidimensional motion trajectory, acquired from a single demonstration, by learning a GMM that encodes local directions and relations of trajectory variables. First, a GMM is fitted to the motion trajectory reduced to a lower-dimensional space by Principle Component Analysis (PCA), and then consecutive Gaussians in the learned GMM with different directions are used to segment the trajectory. Although GMMs are efficient at encoding motion primitives, it is unclear how these models generalize to task variations, such as changes in positions of objects or goals. For instance, applying the learned motion primitive to a task with different initial or final object positions requires some geometric transformation, and it is ambiguous if these transformations will still guarantee the satisfaction of the motion primitive constraints. Our work leverages object interactions inferred from demonstrated grasp poses to ground learned position and geometric constraints to relevant objects in the environment, allowing the learned task plan to adapt to positional variances of objects with some limitations. We will discuss its implementation details, benefits, and limitations in section 3.2.

Hidden Markov Models (HMMs) and statistical model-based change point detection algorithms are commonly used to segment demonstrations based on changes in specified models or latent variables [27, 33, 35, 49]. Iqbal et al. perform online activity segmentation employing an HMM to model activity transitions, with activity labels as hidden states and trajectory frames as observed variables [33]. Gutierrez et al. utilize corrective demonstrations to modify learned task models, represented as finite-state automata (FSA), by training a state transition auto-regressive hidden Markov model (STARHMM), where hidden states index primitives and termination states govern primitive transitions [27]. Fearnhead and Liu developed an online change point detection method that detects points of change of underlying models given a set of candidate models that generate an observation sequence [20, 21, 22]. Research in robotics has leveraged this online Bayesian approach to segment trajectories based on skill or motion models [35, 49]. Konidaris et al. segments trajectories into skill chains by specifying the candidate models of the change point detection method to be the set of basis functions defining the skill abstractions [35]. Similarly, Niekum et al. define the underlying models as a set of articulation models, performing articulate motion segmentation utilizing Changepoint detection using Approximate Model Parameters (CHAMP) [49]. Although these techniques can efficiently infer change points between predefined models online, they are limited to inference over parameterized models and cannot recognize trajectory segments that an underlying model cannot describe. For instance, the articulate motion segmentation work, unable to distinguish between articulated motion and arbitrary unconstrained motion, will fit unconstrained motion trajectories into an articulated motion model. Our work proposes an online constraint detection technique to distinguish between constrained motion segments following an underlying straight-line motion model and arbitrary unmodeled unconstrained motion. We augment this online model with an offline counterpart to improve the changepoint detection accuracy and evaluate performance using a human subject experiment.

Although Inverse Reinforcement Learning (IRL) or Inverse Optimal Control (IOC) has been extensively explored in LfD literature as means of learning tasks from expert demonstrations as reward or cost functions to be optimized [2, 19, 23, 26], majority of these methods have difficulty explicitly modeling hard constraints and local characteristics of a task. Bayesian Nonparametric IRL (BN-IRL), however, learns local properties of a task as simple reward functions by partitioning demonstrations into subtasks [43]. As in conventional IRL, BN-IRL models a demonstration as a Markov Decision Process (MDP), where a demonstration is defined as a time-ordered set of state-action pairs. However, it posits partitioning of the MDP into groups, each with its subgoal and reward function, and introduces a latent assignment variable associating each observed state-action pair to a group. Park et al. presented Constraint-based

BN-IRL (CBN-IRL) as a constraint learning extension to BN-IRL, where, in addition to learning partitions, CBN-IRL can model hard locally-active constraints as constraint-based transition functions [50]. Although efficient at segmenting MDPs into subtasks with local constraints, CBN-IRL cannot guarantee the precise delineation of tasks into constrained and unconstrained subtasks due to the unconstrained nature of the latent assignment variable and the information loss arising from the sampling-based discretization of the continuous state space. Our work utilizes a line fitting model to force segmentation to occur at unconstrained and constrained motion boundaries. Furthermore, we use an offline segmentation phase to mitigate the inaccuracies arising from the filtering-based discretization step in our online segmentation algorithm, details of which will be presented in chapter 3.

Robot manipulation tasks often require modeling of global and local task constraints. Throughout LfD research, numerous types of constraints such as global task space constraints [5, 18], safety constraints [67], position and force constraints [64], conceptual constraints [46, 48], shared constraints across tasks [14], high dimensional parametric constraints [1, 15, 16] and geometric constraints [24, 37, 53, 66] have been examined. Our work focuses on learning locally active trajectory constraints that segment demonstrations into unconstrained and constrained segments. Task space regions (TSR) [7], a widely used geometric constraint representation [53, 41, 66, 44, 42], utilizes a reference transform, an offset transforms, and a bounding matrix to define a volume in rigid-body pose space, SE(3), that represents a constraint. Mohseni-Kabir et al. use TSRs to encode guiding constraints in narrow passages learned by segmenting feasibility sample ratios [41, 44]. To calculate feasibility ratios, the authors found feasible connected samples by sampling the task space surrounding a trajectory and rejecting samples that are in collision with the environment or disconnected from their corresponding demonstration pose. Similarly, Liu et al. learn articulated constraints as TSRs from a continuous visual demonstration [42]. In their work, constrained segments in a motion trajectory are extracted by denoising least-square fitting errors using Total Variation Denoising (TVD) [57] and then segmenting the smooth time-varying signal by fitting a series of step functions. They compare their technique to the well-established work on learning articulated motions from visual demonstrations [51] and report improved average error measurements suggesting accurate articulated constraints learning from a single continuous demonstration. However, the learning accuracy hinges on the values of the predefined parameters, such as the errors thresholding parameter, which requires cross-validation of demonstration samples collected in advance. The accuracy is also dependent on the assumption that the teacher will only demonstrate articulated motion (including straight-line motion) when essential to the task. Furthermore, the complexity of their algorithm is n^2, which implies a quadratic increase in runtimes with an increase in demonstrated trajectory length. Our work utilizes a simple calibration per user to reduce the number of predefined hyperparameters. Additionally, we propose an active learning framework with online constraint detection that leverages human feedback to clarify the validity of demonstrated straight-line motion, effectively eliminating the assumption mentioned above, coupled with an offline keyframe learning algorithm of n complexity to improve the accuracy of constraints learned online.

Akgun et al. introduced the concept of keyframes, an ordered sparse set of sequential poses, to function as a compact task representation that is independent of an agent's kinematics [3]. Additionally, their work explored trajectory to keyframe conversions by defining keyframes as points required to recover the original trajectory from a trajectory generated by a spline technique. Since then, keyframe demonstrations have been extensively utilized, in LfD literature, as means of eliminating noisy undesirable motion [24, 37, 4, 53, 28, 66]. However, task representations composed of sequential end-effector poses cannot capture more complex properties such as task constraints. Consequently, Kurenkov et al. introduced constrained keyframes (c-keyframes), an extension to keyframes, to characterize a space of possible poses for an end-effector [37]. C-Keyframes, defined with orientation and box-shaped positional constraints, were learned from the spatial covariance of keyframe clusters. Similarly, LfD researchers have explored the coupling of keyframes and geometric constraints to create compact, agent-independent task representations [24, 53, 66]. For example, TSRs have been utilized to represent keyframes with geometric constraints [53, 66].

Perez-D'Arpino and Shah learn a multistep manipulation task using a single keyframe demonstration augmented by a library of primitive actions with volumetric, orientation, and move-in-line constraints. Each primitive is encoded using a sequence of TSR keyframes learned by clustering multiple keyframe demonstrations. Although keyframe demonstrations facilitate temporal alignment while allowing generalization, trajectory demonstrations are more intuitive to novice teachers and retain critical speed and timing information. Furthermore, the quality of the task representation learned from keyframe demonstrations hinges on the demonstrator's ability to provide an acceptable set of keyframes. Our work leverages an ordered sequence of TSR-defined keyframes to create agent-independent task specifications. However, instead of user-defined keyframes, we propose a framework that learns keyframes from a single continuous demonstration of a task and empirically compare the performance, subjective usability, and teaching workload of continuous demonstrations and keyframe demonstrations through a human subject experiment.

2.2 Active Learning in LfD

Active learning, a framework that allows robots to elicit human feedback for improved learning, has been utilized in robotics to reduce the time and effort spent teaching new skills [12]. Moreover, this paradigm enables humans to teach robots more naturally, increasing the accessibility of LfD systems to users inexperienced in robotics and machine learning [10]. Cakmak et al. examined the design implications of different interaction modes, and types of queries for active learning settings in human-robot interaction [10, 11, 12]. In [10], Cakmak et al. explore three query modes: (1) queries made every turn, (2) queries made only under certain conditions, and (3) queries made only upon the teacher's request and their benefits compared to passive learning. They found all three active learning modes to have improved performance and human preference than passive learning. However, the optimal strategy between the three active learning modes remained inconclusive and most likely user-dependent. Cakmak and Thomaz then studied the implications of query types (label, demonstration, and feature), their form (closed-form and open-ended), and their physical grounding in active learning systems [12]. They discovered that human teachers preferred physically grounded, closed-form feature queries and presented guidelines on designing questions asked by robot learners. Following their guidelines, we adopt closed-form feature queries based on constraints with three predefined answers “yes”, “no”, and “I don’t know”. To physically ground these queries, we pose them at the time of occurrence during the demonstration and present images illustrating each answer’s meaning. As the robot poses feature queries, we opted for the second interaction mode by querying the teacher only when a constraint is detected.

Active learning has been increasingly explored in robotics and learning from demonstration fields. Cui and Niekum [17] utilize active Bayesian inverse reinforcement learning, where the learning agent leverages human feedback to update its belief over reward functions. First, the learning agent proposes a trajectory that a human teacher then segments into good and bad sections. These labeled trajectory segments, called critiques, inform the learning agent’s belief over reward functions for the task at hand. Sadigh et al. used active learning to infer human preferences for a dynamical system’s behavior in the form of reward functions [58]. They generated two candidate trajectories based on a finite sequence of actions provided by the human and queried the human teacher for their pairwise preference. These modify the hypothesis space of reward functions for human preference according to a maximum volume removal heuristic. Biyik et al. [8] extended this by exploring the use of the maximum information gain criterion to generate more efficient queries, while Biyik and Sadik, [9], implemented a batch active framework to generate multiple pairwise queries simultaneously. Wilde et al. learn user preferences for complex task specifications using an active learning framework [65]. Here, users rank alternate paths that the agent generates, refining the cost function that captures user preferences. The agent continues to iteratively generate alternate paths based on the refined cost function and request feedback, eventually approaching the unique solution to the shortest path problem. While the work mentioned above learns only from human feedback, Shah et al. propose a Bayesian interactive robot training framework that learns from both demonstrations provided and human feedback [61]. The learner first builds a belief over task specifications defined using LTL formulas and then updates its belief based on the teacher's assessment of the label queries. These label queries are the specifications with high uncertainty presented to the teacher as task executions. Like Shah et al. [61], we design our active learning framework for learning from demonstrations and human feedback. However, we propose an online active learning framework where the learner elicits feedback during the demonstration to clarify the teacher's intent on constrained regions, limiting the interaction to a single demonstration.

Constraint learning frameworks have leveraged active learning to learn task constraints [31], semantic constraints [63], and behavioral constraints [47] more efficiently. Hayes and Scassellati explore the use of feature queries to learn a constraint network composed of acceptable skill sequences [31]. The system observes the first demonstration and poses open-ended feature queries, informed by an action-space graph, during the subsequent demonstration to maximize learning gains. Tabrez et al. perform skill repair through an interactive framework that utilizes a semantic hierarchy to select skill repairing semantic constraints [63]. The robot elicits feedback on an executed skill in two stages; first, open-ended feedback on how the skill is failing, and second, feedback on constraint parameter nodes of a tree whose leaf nodes represent a fully parametrized constraint. Confirmation of a leaf node indicates the selection of a constraint. Although [31] and [63] allow novice users a more natural interaction with a robot learning constraints, the impact of the given constraints on the robot's mental model and the learned model remains unknown to the user.

To teach a robot naturally and effectively, a human needs to have an accurate mental model of the robot [54, 55]. Mueller et al. explore the use of Robot Behaviour Counterfactuals (RBCs) and Behavioral Verification Indicators (BVIs) in their active learning framework to learn task-specific behavior restrictions through human feedback [47]. RBCs visualize the robot's expected behavior when task constraints change, whereas BVIs illustrate the new model's success or failure in achieving the task goal. In this framework, skills learned during demonstration can be edited by adding new or editing existing behavioral constraints using visualizations in augmented reality.

The RBCs and BVIs presented during the editing phase allow users to build better mental models of the robot by understanding how each constraint changes the execution of the task. Similarly, our work attempts to improve the teacher's mental model of the learner by presenting images with the queries illustrating the resultant robot motion associated with each answer. Additionally, if the robot detects an unintended incorrect constraint, it gives the user suggestions on demonstrating unconstrained regions without triggering a constraint query. In other words, based on the teacher's feedback, the learner will provide feedback to the teacher on improving the teaching efficiency, further increasing the teacher's understanding of the learner's mental model. To the best of our knowledge, we believe this is the first work that combines online constraint detection, online human feedback for active learning, and online suggestions for teaching improvement in the context of human-robot interaction. Here, online refers to the occurrence during the demonstration.

Chapter 3

Method

3.1 Problem Formulation

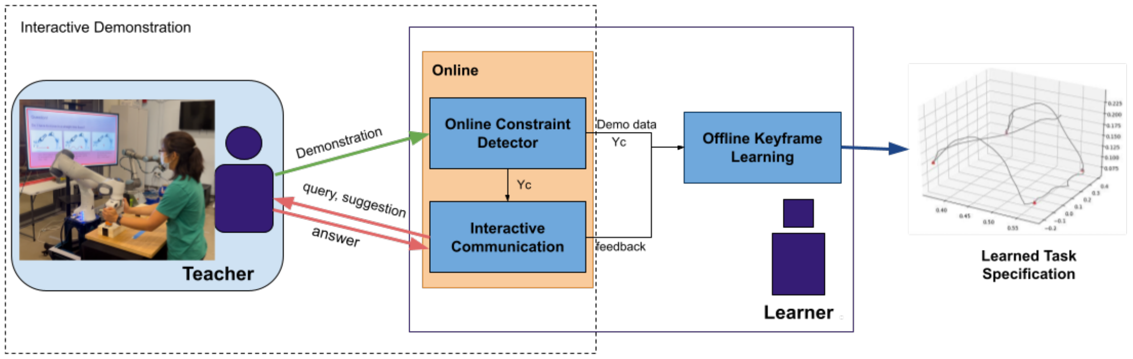

Interactive Demonstration
Teacher
Demonstration
query, suggestion
answer
Online
Online Constraint Detector
Yc
Interactive Communication
Demo data
Yc
feedback
Offline Keyframe Learning
Learner
Learned Task Specification

Figure 3-1: Our proposed active keyframe learning framework that learns the task specification as an ordered sequence of interaction and constraint keyframes from a single demonstration of the task.

Our work focuses on learning a task represented as the least restrictive ordered sequence of keyframes through a single interactive demonstration. Here, the least restrictive ordered sequence of keyframes refers to the task specification that results in a successful execution trace while allowing the most flexibility to the motion planner. To teach a task, the teacher provides a kinesthetic demonstration of the task which the learner observes as $\{X_{ee}, X_g, \{X_w\}\}$, the trajectories of the end effector's pose, the gripper's state, and the positions of the objects in the scene. Here, X_{ee} and X_g are the trajectories of the end-effector's pose and the gripper's state, while $\{X_w\}_{w=0:N_{obj}}$ denotes the set of trajectories of the objects where N_{obj} is the total number of objects present. We define trajectories as $X \in R^{N \times M}$, where M is the number of frames recorded during the demonstration. For a trajectory of poses, $N = 6$ denotes the x, y, and z Cartesian axes and the rotations about those axes, while for a trajectory of gripper states, $N = 1$ represents the state of the gripper G, a boolean variable indicating if the gripper is open ($G = 0$) or close ($G = 1$). In addition to observing the end-effector and object poses, the learner can interrupt the teacher during their demonstration of the task to pose closed-form queries or provide suggestions, as shown in figure 3-1. After each query, the teacher must first pause the demonstration, provide an answer, and then continue with the task. Answers are recorded as $Fb \in R^{B \times N_{Fb}}$, where $B = 2$ denotes the answer-time step pairs and N_{Fb} is the total number of answers given by the teacher. In this setting, we assume that the teacher answers all queries and that all answers provided are correct. On the other hand, teachers have the liberty to accept or decline suggestions given, and we assume that the learner does not know if the suggestion has been accepted or declined.

Given the inputs $\{X_{ee}, X_g, \{X_w\}\}$ and Fb, the system learns a task specification for the demonstrated task encoded as a a set of keyframes that can be executed sequentially to accomplish the task. In our work, keyframes are expressed using Task Space Regions (TSRs) [7]. A TSR consists of three components 1) T^o_w: the transform from the origin to the frame w, 2) T^w_e: end-effector, e, offset in frame of w and, 3) B^w: 6×2 matrix of bounds . Here, w is the frame of an object in the scene. For convenience, let a TSR be denoted by $Y_t = \{T^o_w, T^w_e, B^w\}$. A keyframe $KF_i = \{Y^i_t, G^i, \{C^i_j\}\}$ is a TSR Y_t coupled with the state of the gripper G and an associated set of constraints $\{C_j\}_{j=1:N_c}$, where N_c is the total number of constraints present in the trajectory segment $[KF_{i-1}, KF_i]$ and can differ between segments. This allows one to specify a task using a set of sequential keyframes as $\{KF_i,\}_{i=1:z}$, where z is the total number of keyframes and is inferred from the demonstration provided. These keyframes are learned by inferring the physical interactions between the end-effector and objects in the task (to learn G), and the geometric constraints (to learn

$\{C_j\}$). We limit the scope of this work to the move-in-line constraint introduced in [53] implying $N_c = 1$ and the indirect and direct physical interactions as defined in section 3.2.

In summary, given the inputs $\{X_{ee}, X_g, \{X_w\}\}$ and Fb we learn the least restrictive task specification $\{KF_i, \}_{i=1:z}$ by inferring physcial interactions and move-in-line constraints from a single interactive demonstration of the task.

3.2 Task Specification

A set of sequential keyframes can encode a multistep task to create a task specification that is agent independent [53]. However, multiple such specifications may exist for a given task, each providing varying levels of flexibility during motion planning. Our work aims to learn the least restrictive task specification, described as the specification resulting in a successful execution trace while allowing the most flexibility to the motion planner. In this section, we will first describe the different types of keyframes, interaction keyframes, constraint keyframes, and combined keyframes, the selection of reference object w, and then discuss what it means to learn the least restrictive task specification.

3.2.1 Interaction Keyframes

Capturing the physical interactions between the robot and objects in the environment is essential to learning an accurate task specification. We categorize physical interactions into two types: (1) Direct interactions and (2) Indirect interactions, as shown in figure 3-2. Direct interactions are instances where the end-effector comes into contact with an object, w_d. For example, the robot grasping an object can be classified as a direct interaction. Indirect interactions occur when an object, w_d, grasped by the end-effector directly interacts with another object, w_in, in the environment. An example of an indirect interaction would be the action of placing an object on a surface. Both direct and indirect interactions are represented by a single keyframe KF_i denoting the position, the object, and the gripper state of the interaction. Although complex

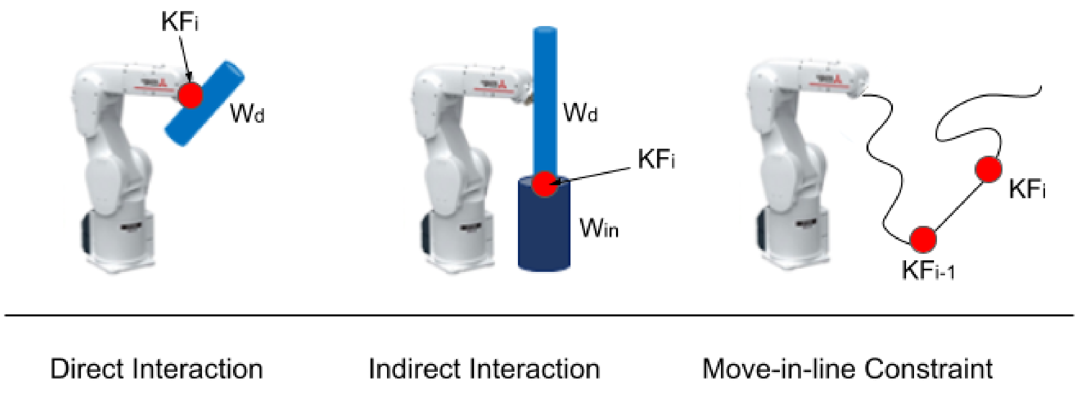


Figure 3-2: Interaction and constraint keyframes learned. Here, keyframe poses are displayed as red circles. Encoding interactions require a single keyframe, whereas, encoding a move-in-line constraint requires two.

tasks may consist of non-physical interactions, the scope of this work is limited to direct and indirect physical interactions.

3.2.2 Constraint Keyframes

Locally-active geometric constraints, such as a straight line, can be encoded using two consecutive keyframes $[KF_{i-1}, KF_i]$ as shown in figure 3-2. Here, KF_{i-1} and KF_i denote the start and end of the constraint and $\{C^i_j\}$ indicates the type of constraint. In this work, we solely focus on move-in-line constraints where the end-effector will move in a straight line in Cartesian space, from KF_{i-1} to KF_i implying $KF_{i-1} = \{Y^{i-1}_t, G^{i-1}, C^{i-1} = 0\}$ and $KF_i = \{Y^i_t, G^i, C^i = 1\}$.

3.2.3 Combined Keyframes

As interactions and constraint boundaries are captured by different variables in a keyframe G and C_j when an interaction and a constraint boundary coincide in 3D space, we represent both using a single keyframe. These keyframes that simultaneously describe a constraint and interaction are called combined keyframes. For instance, sliding an object on the table requires the robot to grasp the object, slide the object on the table and release the object. In this task, the grasp and release interactions coincide with the start and end move-in-line constraint boundaries, respectively. Therefore, in lieu of using two interaction keyframes and two constraint keyframes, the task specification can be reduced to two combined keyframes, $KF_1 = \{Y_t^1, G^1 = 1, C^1 = 0\}$ and $KF_2 = \{Y_t^2, G^2 = 0, C^y = 1\}$, each simultaneously describing an interaction and a constraint boundary.

3.2.4 Selecting the Reference Object w

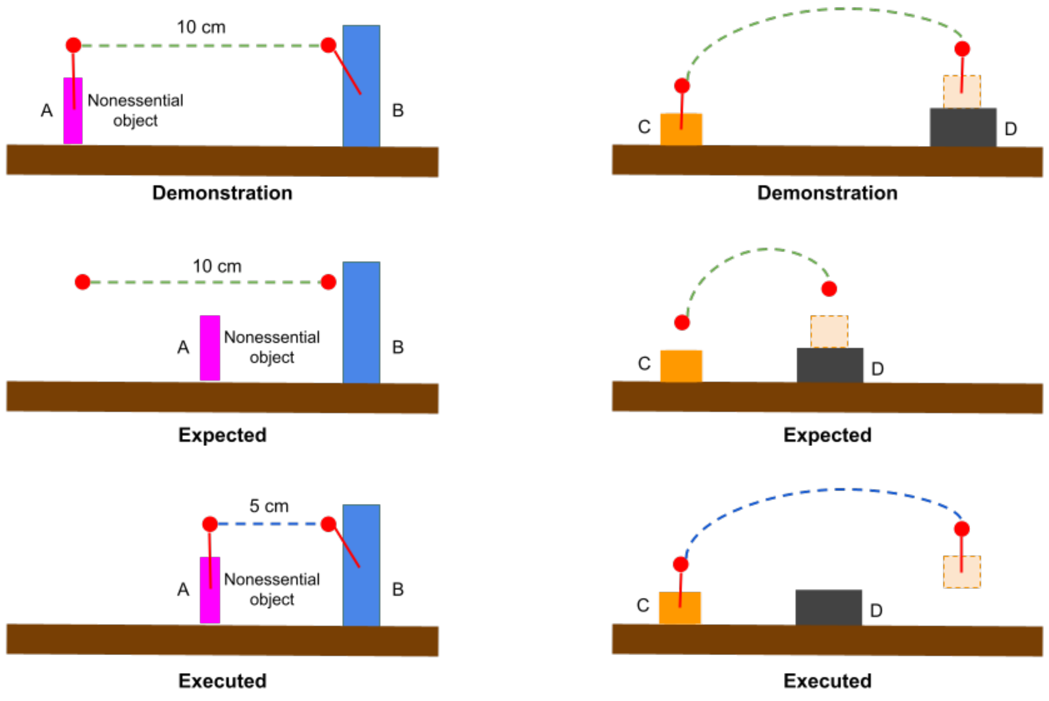


Figure 3-3: Limitations in current work for reference object w selection strategies. The task on the left is to move 10cm in a straight line when reaching object B, the task on the right is to place object C on D. The keyframes are shown in red with a red line indicating the reference object w. Green dashed lines indicate correct trajectories, while blue indicate incorrect trajectories. Due to the limitations in the existing strategy for selecting w, the learned keyframes do not adapt to object position variations in a task.

A keyframe records the end-effector pose with respect to an object w to ensure robustness against variations in object positions. In [53], the object w is defined as

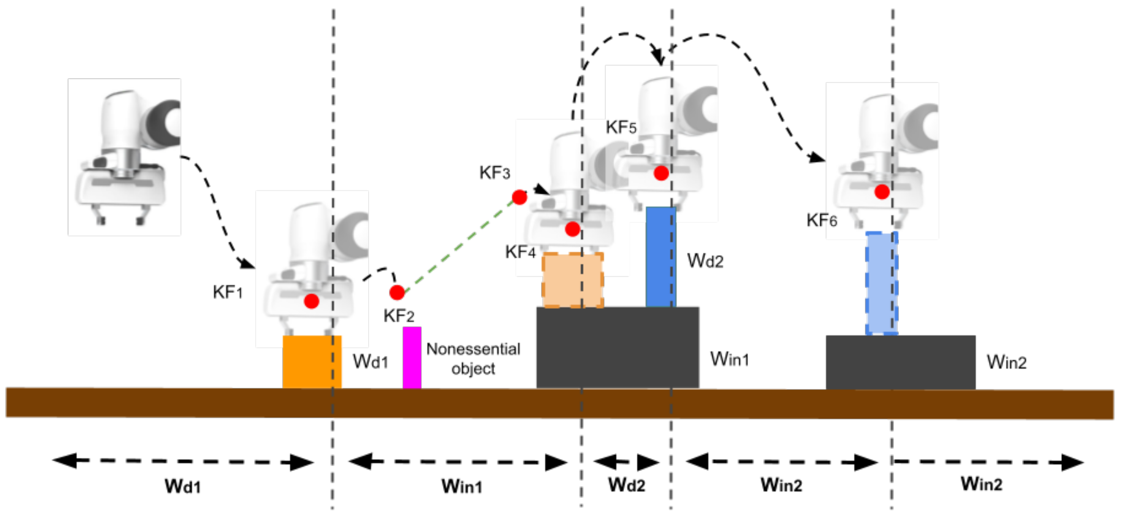


Figure 3-4: Reference object w selection for a task where first, the orange block W_{d1} is placed on the large gray block W_{in1} and then the blue block W_{d2} is placed on the small gray block W_{in2}. The green dashed line indicates a move-in-line constraint. Reference object w for the keyframes are selected to be the key object in the next interaction.

the object nearest to the end-effector. However, this will result in the inability to adapt to position changes in certain tasks involving direct and indirect interactions. To illustrate, consider a direct interaction task requiring a 10 cm straight-line motion to object B, as shown in figure 3-3. As the nearest object definition will select w of the first constraint keyframe as object A, when object A is only 5cm away from B, the move-in-line constraint will be incorrectly executed as a 5cm straight-line. Another example is the indirect interaction task of placing object C on object D, as shown in figure 3-3. Here, object C is the nearest object to the end-effector, which will result in an interaction keyframe that cannot adapt to changes in the position of object D. Another disadvantage of the nearest object definition for w is the generation of keyframes dependent on nonessential objects near the end-effector. For example, in figure 3-3, although object A is unnecessary for the task, the keyframe generated depends on this nonessential object's position. For these reasons, we define object w as the key object in the next interaction, as shown in figure 3-4. Here, the key object of an interaction refers to w_d in direct interactions and w_{in} in indirect interactions.

This definition enables task specifications to adapt to variations in object positions and prevents the keyframe dependence on nonessential objects, as shown in figure 3-4

3.2.5 Least Restrictive Task Specification

Task specifications learned as an ordered sequence of keyframes can be used as input to motion planners to generate execution traces for a given robot to perform the tasks [53]. Learning these task specifications independent of the low-level motion primitives allows systems to leverage motion planners that optimize execution traces for their unique objectives, such as power efficiency or time efficiency. This benefit of learning agent independent task specifications relies on the flexibility given to the motion planner. In other words, the more flexibility given by a specification allows the motion planner to generate a more efficient execution trace. We define flexibility given by a specification as the number of execution traces that satisfies the specification.

Our work aims to learn the task specification, encoded by interaction, constraint, and combined keyframes, that maximizes the flexibility given to a motion planner. We refer to this specification as the least restrictive task specification for convenience. As multiple specifications with varying levels of flexibility can exist for a single task, we theorize that the least restrictive task specification is the specification with the least number of keyframes.

Definition 3.2.1. *For a motion planner, MP, and robot,R, with a constant initializing pose, given a specification as a set of ordered keyframes, S with $|S| = n$, where $\{n | n \in Z, n \geq 1\}$ is the number of keyframes in S, the flexibility given by S is defined as the number of acceptable execution traces $|T_S|$, where T_S is the set of execution traces that satisfy S.*

Lemma 3.2.1. *If a keyframe is added to specification S to create a new specification S' such that $S \subset S'$ and $|S'| = n + 1$, where $|S| = n$, then, the flexibility given by S', $|T_{S'}| \leq$ the flexibility given by S, $|T_S|$.*

Proof. Let $S = \{KF_i,\}_{i=1:n}$, $|S| = n$ and $S' = S \cup \{KF_{new}\}$, $|S'| = n + 1$. Here, KF_{new} is the new keyframe added to specification S to create the new specification

S' and can be an interaction, constraint or combined keyframe. As specifications are ordered sets, KF_{new} can be inserted in any position giving $S' = \{KF_j, \}_{j=1:n+1}$. Now, $T_S = \{\text{execution traces that satisfy } \{KF\}_{i=1:n}\}$, and
$T_{S'} = \{\text{execution traces that satisfy } \{KF\}_{i=1:n} \text{ and } \{KF_{new}\}\}$.
As $T_{S'} \subseteq T_S$, $|T_{S'}| \leq |T_S|$ which implies the flexibility given by S', $|T_{S'}| \leq$ the flexibility given by S, $|T_S|$. □

Lemma 3.2.2. *If $x \in Z$ number of keyframes are added to specification S to create a new specification S'' such that $S \subset S''$ and $|S''| = n + x$, where $|S| = n$, then, the flexibility given by S'', $|T_{S''}| \leq$ the flexibility given by S, $|T_S|$.*

Proof. We proceed using induction.
Base Case: Consider the case of $x = 1$. As $S \subset S''$, $|S| = n$ and $|S''| = n + 1$, from lemma 3.2.1 we get $|T_{S''}| \leq |T_S|$, so the result holds.
Inductive Hypothesis: Suppose, for some $x \in Z$, if we add x keyframes to S to create S' where $S \subset S'$, $|S| = n$, and $|S'| = n + x$ then $|T_{S'}| \leq |T_S|$.
Inductive Step: Consider the case of $x + 1$, where we add a single new keyframe to S' to get S'' where $S' \subset S''$ and $|S''| = n+x+1$. From lemma 3.2.1 we have $|T_{S''}| \leq |T_{S'}|$. From the inductive hypothesis we have $|T_{S'}| \leq |T_S|$, implying $|T_{S''}| \leq |T_S|$. Therefore, we prove that if the statement holds for x, it also holds for $x + 1$.
Hence by induction, the flexibility given by S'', $|T_{S''}| \leq$ the flexibility given by S, $|T_S|$ □

Definition 3.2.2. *For a motion planner, MP, and robot, R, with a constant initializing pose, given a set of task specifications $TS_V = \{S_i\}_{i=1:N_{ts}}$ that can be used by MP to generate successful execution traces for a single task V, where N_{ts} is the total number of specifications that represent V, the least restrictive task specification S_{LR} is defined as the specification $S_i \in TS_V$ that maximizes flexibility given to MP, $|T_{S_i}|$. That is,*

$$S_{LR} = \underset{S_i \in TS_V}{\operatorname{argmax}} |T_{S_i}|, \tag{3.1}$$

where $|T_{S_i}|$ is the flexibility give by S_i, as defined in definition 3.2.1

Theorem 3.2.3. *Given a set of task specifications $TS_V = \{S_i\}_{i=1:N_{ts}}$ for a task V, the least restrictive task specification S_{LR}, as defined in 3.2.2, is the specification $S_c \in TS_V$ with the least number of keyframes. That is, $\forall S_i \in TS_V \backslash \{S_c\}$, if $|S_c| \leq |S_i|$, where $S_c \in TS_V$, then $S_{LR} = S_c$.*

Proof. For a motion planner to generate successful execution traces, the set of keyframes given must capture all the interactions and constraints essential for the successful execution of a task. Let the specification that consists of only the interaction, constraint and combined keyframes crucial for the success of a task be $S_c = \{KF_j\}_{j=1:N_c}$, where N_c is the total number of crucial keyframes. As S_c generates acceptable execution traces, $S_c \in TS_V$. $\forall S_i \in TS_V$, S_i generates acceptable execution traces implying that $S_c = \{KF_j\}_{j=1:N_c} \subset S_i$. Therefore, $\forall S_i \in TS_V$, $S_c \subset S_i$ and $|S_c| \leq |S_i|$ which can be written as $|S_i| = |S_c| + x_i$, for some $x_i \in Z$. Applying lemma 3.2.2 we get $\forall S_i \in TS_V$, $|T_{S_i}| \leq |T_{S_c}|$. Therefore, $S_{LR} = \text{argmax}_{S_i \in TS_V} |T_{S_i}| = S_c$, where $S_c \in TS_V$ and $\forall S_i \in TS_V \setminus \{S_c\}$, $|S_c| \leq |S_i|$. □

Theorem 3.2.3 proves that the least restrictive task specification is the specification with the least number of keyframes that results in successful execution traces for the task. Our work aims to learn the least restrictive task specification from a single demonstration of the task by learning the interaction, constraint, and combined keyframes essential for the success of a task, S_c. Here, a set of keyframes can be trivially verified to be essential for the task's success by ascertaining that removing any keyframe in the set will result in an erroneous task specification. In our experiments, we evaluate all learned tasks compared to the least restrictive task specifications for the given tasks and report our findings in chapter 4.

3.3 The Active Keyframe Learning Framework

We propose an active keyframe learning framework that learns, from a single demonstration, the physical interactions and move-in-line constraints of a task as the least restrictive task specification encoded by sequentially ordered keyframes. However, learning move-in-line constraints from a single demonstration of a task can result in over-constrained task specifications [53]. For instance, a teacher with inexperience in robotics may move the robot in straight lines in unconstrained regions of the task, unintentionally teaching an over-constrained task. To this end, we learn the move-in-line constraints in two stages: (1) the interactive demonstration and (2) offline keyframe learning, as shown in figure 3-1. The interactive demonstration stage leverages active learning to understand human intent behind demonstrated straight-line motion and provides suggestions to the teacher to improve their mental model of the learner. Next, the human intent captured as feedback from the teacher is utilized in the offline keyframe learning stage to remove incorrect constraints learned from the demonstration data reducing the over-constrained nature of the learned task specification. Physical interactions between the robot and the objects are learned solely in the offline keyframe learning stage using the recorded demonstration data. This section describes implementation details of the interactive demonstration and offline keyframe learning stages in our active keyframe learning framework that learns the least restrictive task specification for a demonstrated task.

3.3.1 Interactive Demonstration

The interactive demonstration framework consists of an online constraint detector and interactive communication, as shown in figure 3-1. During teaching, the online constraint detector observes the end-effector poses as a sequence of frames $X_{ee} = \{x^i_{ee}\}_{i=1:M_t}$, where M_t is the number of frames recorded until time t, and infers the latent binary state variable, $Y_c[t]$, denoting the constrained nature of demonstrated motion at time step t. Here, $Y_c[t] = 0$ indicated unconstrained motion and $Y_c[t] = 1$ indicates constrained motion, at time step t. Informed by $Y_c[t]$, interactive communication begins with the learner's query, followed by the teacher's answer, and ends with a suggestion from the learner. This dialog occurs each time $Y_c[t]$ toggles from 0 to 1. Once the interactive demonstration ends, the teacher's answers Fb, the inferred state variable $\{Y_c\}$, and the demonstration data recorded $\{X_{ee}, X_g, \{X_w\}\}$ serve as input to the offline learning phase.

3.3.1.1 Online Constraint Detector

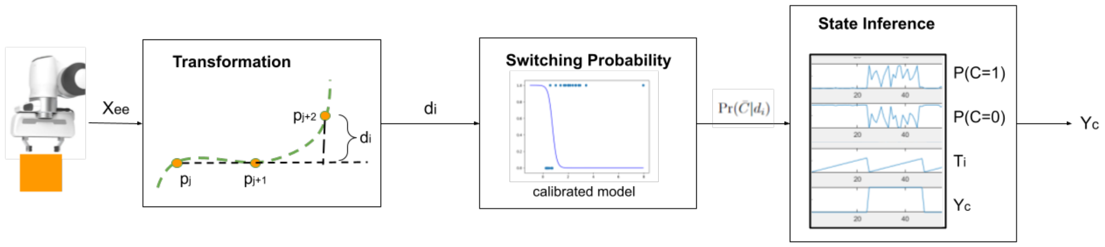


Figure 3-5: Architecture of the online constraint detector

The online constraint detector infers the latent state variable $Y_c[t]$, that denotes the constrained nature of the demonstrated motion at time t, from the end-effector poses $X_{ee} = \{x^i_{ee}\}_{i=1:M_t}$ observed until time t, as shown in figure 3-5. First, the end effector poses are transformed to $\{d_i\}_{i=1:M''_t}$ that represents the distance error of x_{ee} to constrained line motion, followed by model fitting of d_i to get the switching probability $\Pr(\bar{C}|d_i)$ and finally, the inference of of the latent state $Y_c[t]$. This process happens online, updating $Y_c[t]$ with each new end-effector pose observation x_{ee}.

Transformation. During the demonstration, the position of the end-effector is recorded as a sequence of frames $X_{ee} = \{x^i_{ee}\}_{i=1:M_t}$, where M_t is the number of frames recorded until time t. First, these frames are filtered online using the position vector of the frame x^i_{ee}, $\boldsymbol{p}_i = (x_i, y_i, z_i)$, such that, $\|\boldsymbol{p}_{j+1} - \boldsymbol{p}_j\| = D_F$. Here, D_F is a predefined distance value and $\boldsymbol{p}_j$ is the position vector of the frame x^j_F in the new filtered sequence of frames $X_F = \{x^j_F\}_{j=1:M'_t}$. Next, the filtered trajectory X_F is converted to the distance errors $\{d_i\}_{i=1:M''_t}$ by computing the perpendicular distance from $\boldsymbol{p}_{j+2}$ to the the straight line fitted to $\boldsymbol{p}_{j+1}$ and $\boldsymbol{p}_j$, using the equation,

$$d_i = \sqrt{\|\boldsymbol{p}_{j+2} - \boldsymbol{p}_{j+1}\|^2 - \left[\frac{(\boldsymbol{p}_{j+2} - \boldsymbol{p}_{j+1}) \cdot (\boldsymbol{p}_{j+1} - \boldsymbol{p}_j)}{\|\boldsymbol{p}_{j+1} - \boldsymbol{p}_j\|}\right]^2}. \tag{3.2}$$

Switching Probability. As the scope of this work is limited to the straight line constraint, there are only two states the end-effector can be in at instance i: (1) constrained ($Y_c^i = 1$) and (2) unconstrained ($Y_c^i = 0$). We employ a Logistic Regression model, with distance error $\{d_i\}$ as the independent variable, to model constrained and unconstrained state of motion, Y_c^i. The parameters of this model, unique to a user, are learned by fitting a logistic regression model to labeled distance errors computed using labeled demonstration data of a straight line and a circle. This labeled demonstration data is recorded during a simple calibration step, required once per user, where the user moves the end-effector first in a straight line and next in a circle.

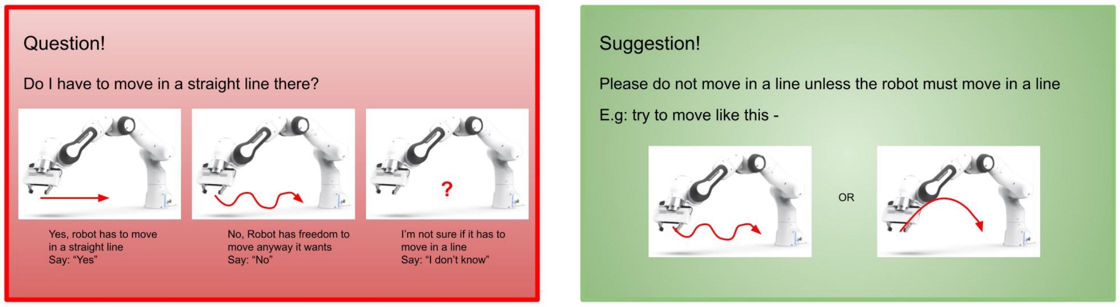
During the demonstration of a task, given the logistic regression model for constrained and unconstrained motion, the probability of the constraint state switching given d_i and $Y_c^{i-1} = C$, $\Pr(Y_c^i = \bar{C} | d_i, Y_c^{i-1} = C)$, is the classification probability of $Y_c^i = \bar{C}$ given d_i, $\Pr(\bar{C}|d_i)$, calculated as,

$$\Pr(\bar{C}|d_i) = \begin{cases} \frac{1}{1+\exp(-\beta_0 - \beta_i d_i)} & C = 1 \\ 1 - \frac{1}{1+\exp(-\beta_0 - \beta_i d_i)} & C = 0 \end{cases}, \tag{3.3}$$

where β_0 and β_1 are the parameters of the logistic regression model. $\Pr(\bar{C}|d_i)$ informs $Y_c[t]$ as explained below.

State Inference. Due to instrument and human motion noise, simply thresholding the switching probability values can lead to incorrect high frequency switching of state values. Therefore, we drew inspiration from Khoramshahi et Billard [34] and introduced an energy tank that governs state switching. The energy of the tank, T_i, is defined as $T_i = T_{i-1} + \Pr(\bar{C}|d_i)^2 - T_d$, where T_d is the constant dissipated energy. When $T_i \geq T_s$, where T_s is the threshold that triggers a state switch, the state of the system is switched ($Y_c^i = 1 - Y_c^{i-1}$), and the energy of the tank is reinitialized to zero, i.e. $T_0 = 0$. When $T_i < T_s$, $Y_c^i = Y_c^{i-1}$. This behavior can be summarized by the following equations; For tank energy T_i,

$$T_i = \begin{cases} T_{i-1} + \Pr(\bar{C}|d_i)^2 - T_d & T_i < T_s \\ 0 & otherwise \end{cases}, \tag{3.4}$$


Question!

Do I have to move in a straight line there?

Yes, robot has to move in a straight line
Say: "Yes"

No, Robot has freedom to move anyway it wants
Say: "No"

I'm not sure if it has to move in a line
Say: "I don't know"

Suggestion!

Please do not move in a line unless the robot must move in a line

E.g: try to move like this -

OR

Figure 3-6: Question posed to the user when in constrained region (left) and the suggestion made to the user when their answers are *"No"* or *"I don't know"* (right).

and for the constraint state Y_c^i,

$$Y_c^i = \begin{cases} Y_c^i = Y_c^{i-1} & T_i < T_s \\ Y_c^i = 1 - Y_c^{i-1} & otherwise \end{cases}, \tag{3.5}$$

where T_d is the constant dissipated energy and T_s is the threshold energy that triggers a state switch. This infers the latent constraint state variable, Y_c, the output of the online constraint detector used to guide the interactive communication between the human and robot.

Although this online technique allows us to identify the presence of constrained motion, the boundaries of the true constrained segment do not coincide with the boundaries of the predicted constrained segment. This is due to the delay introduced by frame filtering, signal transforming, and state switching steps. To extract accurate constraint keyframes, additional processing must be conducted offline, as explained in section 3.3.2.

3.3.1.2 Interactive Communication

Interactive communication takes the form of a dialogue between the teacher and learner designed to capture the human intent for the demonstrated straight-line motion and improve the teacher's mental model of the learner. The dialogue follows the format [(learner - query), (teacher - answer), (learner - suggestion)], where a pair in parenthesis represent (communicator, type of communication). According to Cakmak et al., the type, form, mode, and physical grounding of queries are essential aspects to consider when designing interactive systems for human-robot interaction [10, 11, 12]. Following their guidelines, we adopt closed-form physically grounded feature queries under the *queries made only under certain conditions* mode. The suggestion is provided based on the teacher's answer and aims to improve the teacher's understanding of the learner.

Query type. As we are interested in leveraging communication to learn the human intent for a straight-line motion, only a single query inquiring about the validity of the demonstrated straight-line motion is required. We design this query as a feature-based query, where the feature is the inferred constraint state of motion $Y_c[t]$ at time t, to be *"Do I have to move in a straight line there?"*.

Query form. Following guidelines presented in [12], we design a closed-form query by predefining the answers available to the teacher as *"Yes"*, *"No"*, and *"I don't know"*. These answers are presented to the teacher each time the query is posed.

Query mode. We adopt the *queries made only under certain conditions* mode by defining the condition to be when $Y_c[t-1] = 0$ and $Y_c[t] = 1$. In other words, the learner will pose the question when the inferred constraint state swiches from not constrained to constrained.

Physical grounding. To physically ground these queries, we pose them at the time of occurrence and present images on a screen, illustrating each answer's meaning to the teacher, as shown in figure 3-6.

Suggestion. The suggestion on demonstrating unconstrained motion to the learner is only given if the teacher's answer is *"No"* or *"I don't know"*. Furthermore, the suggestion, given as *"Please do not move in a line unless the robot must move in a line."*, is displayed on the screen with example images of good unconstrained motion, as shown in figure 3-6. The teacher can choose to accept or decline the suggestion given, and we assume that the learner does not have access to the teacher's decision.

Impact of communication on learner's state. When the teacher provides the answer *"No"* indicating that the constrained motion demonstrated is not a constraint of the task, the learner updates the state $Y_c[t] = 1$ to $Y_c[t] = 0$. The answers *"Yes"*, and *"I don't know"* have no impact on the learner. As the learner does not have access to the teacher's decision on accepting or declining the suggestion, the suggestion does not affect the learner's state.

3.3.2 Offline Keyframe Learning

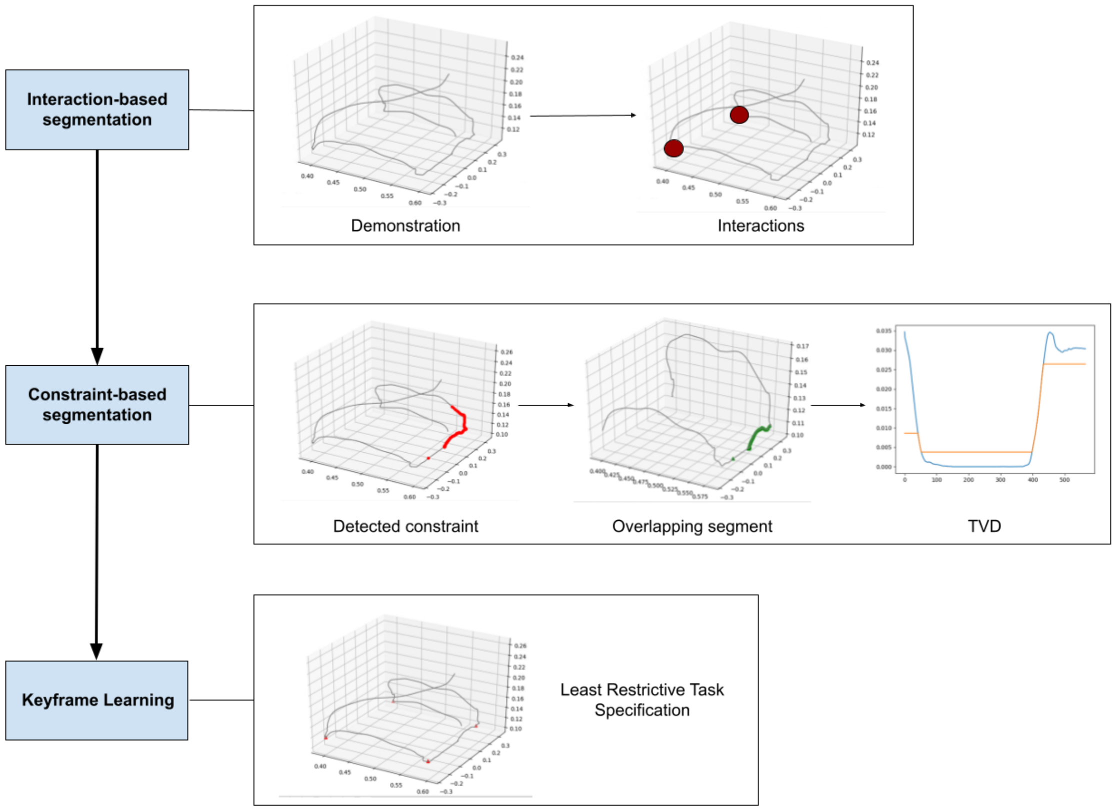

Interaction-based segmentation
Demonstration
Interactions
Constraint-based segmentation
Detected constraint
Overlapping segment
TVD
Keyframe Learning
Least Restrictive Task Specification

Figure 3-7: Overview of Offline Keyframe Learning

This section describes the method proposed to learn a sequence of keyframes from demonstration data, Offline Keyframe Learning (OKL), as shown in Algorithm 1 and

```
Algorithm 1: Offline Keyframe Learning (OKL)
   Input: $X_{ee}$ - A list of demonstration frames
          $X_g$ - A list of gripper states
          $\{X_w\}$ - A list of object poses
          $Fb$ - Human feedback given
          $\{Y_c\}$ - Inferred constraint states
   Output: $S_{LR}$ - Sequence of keyframes that specify the task
 1 $S_{LR} \leftarrow 0$
 2 $S \leftarrow$SegmentTrajectory$(X_{ee}, X_g)$
 3 $Co \leftarrow$ ModifyDetectedConstraints$(\{Y_c\}, Fb)$
 4 for $i$ = 0 to S.size do
 5     $Cr \leftarrow$ getConstrainedRegions$(S(i), Co)$
 6     for $j$ = 0 to Cr.size do
 7         $l \leftarrow$ getTrueLine$(S(i), Cr_j)$
 8         $L \leftarrow$ FitLineModel$(l)$
 9         $e \leftarrow$ FittingError$(S(i), L)$
10         $e_s \leftarrow$ TVD$(e)$
11         $\sigma \leftarrow$ getThreshold$(e_s, l)$
12         $S_{LR} \leftarrow S_{LR} +$ ConstraintKF$(S(i), \sigma)$
13     end
14 end
15 $S_{LR} \leftarrow$ AddInteractionKFs$(X_{ee}, X_g)$
16 $S_{LR} \leftarrow$ CombineKF$(S_{LR,S})$
17 return S$_{LR}$
```

figure 3-7. OKL requires the demonstration frames of the end-effector, gripper state, and the objects in the scene $\{X_{ee}, X_g\{X_w\}\}$, the answers to the queries posed by the robot Fb, and the constraint state inferred online $\{Y_c\}$. Keyframes are learned in three steps: first, line 2 segments the trajectory according to interactions; second, lines 3 - 11 segment the trajectory segments from line 2 based on constraints; and third, lines 12 - 16 learns keyframes from the trajectory segments. The output of OKL is the task specification as an ordered sequence of keyframes that can be given as input to a motion planner to execute the task.

Interaction based segmentation. First, the continuous trajectory recorded, X_{ee}, is segmented based on the physical interactions in the task. In this work, interactions are limited to those that involve a change in the gripper state. For instance, picking an object will cause the gripper state G to switch from 0 to 1, whereas placing an object will change G from 1 to 0. Therefore, to perform segmentation based on interactions, *SegmentTrajectory* in line 2 segments the trajectory by the frames corresponding to a gripper state change.

Constraint based segmentation. To learn constraint keyframes, the constraints detected online are modified to reflect the feedback received. This is achieved by *ModifyDetectedConstraints* by removing the constraint segments corresponding to the feedback *"No"*, in line 3. Next, the constrained regions $Cr = \{Cr_j\}$ in each interaction-based segment is listed in line 5. As explained in section 3.3.1.1, these constrained regions do not perfectly align with the true constraints $Ct = \{Ct_i\}$. Therefore, before fitting the line model L, the overlapping segment l between the true constraint Ct_i and detected constraint Cr_j must be found. Ct_i, Cr_i, and l constitute of a set of sequential poses $\{\boldsymbol{p}^t_k\}_{k=0:N_t}$, $\{\boldsymbol{p}^r_k\}_{k=0:N_r}$, and $\{\boldsymbol{p}^l_k\}_{k=0:N_l}$ respectively. Here, N_t, N_r, and N_l are the number of frames in Ct_i, Cr_i, and l. As Cr_j begins after Ct_i, the starting pose of l can be defined as $\boldsymbol{p}^l_0 = \boldsymbol{p}^r_0$. Furthermore, as the misalignment between Ct_i and Cr_j is caused due to delay introduced primarily by the distance filter, $\boldsymbol{p}^l_{N_l} = \boldsymbol{p}^r_m$, such that, $\|\boldsymbol{p}^r_{N_r} - \boldsymbol{p}^r_m\| \geq D_F$ and $\|\boldsymbol{p}^r_{N_r} - \boldsymbol{p}^r_{m+1}\| < D_F$. Therefore, the function *getTrueLine*, in line 7, returns $l = \{\boldsymbol{p^r}_i\}_{i=0:m}$. Next, a straight line is fitted to l by *FitLineModel*, in line 8, using two least square fittings, to obtain the equation of the straight line constraint L. Then, the fitting errors e of all the poses in $S(i)$ to the line model L is calculated by *FittingError* in line 9.

Total Variation Denoising (TVD) [57] is a filtering method that minimizes the total variation of a signal as means of smoothing the signal while preserving its sharp edges. The calculated fitting errors are denoised using TVD, in line 10, to generate a smoother time-varying signal, e_s. This signal can be thresholded to extract the true constraints. Here, the threshold value σ is set by *getThreshold*, line 11, to be $\sigma = \max(e_l) + \epsilon$, where e_l are the smooth fitting errors that correspond to l, and ϵ is a small positive number.

Keyframe Learning. Following the thresholding of e_s, the constraint keyframes, the boundary frames of the true constraint segment, are added to the list of keyframes S_{LR} by *ConstraintKF* in line 12. Next, *AddInteractionKFs* in line 15 updates the list of keyframes KF with the interaction keyframes learned using X_{ee} and X_g as the frames corresponding to the issued gripper commands. *CombineKF*, in line 16, ensures that interaction and constraint keyframes that coincide are combined into a single keyframe and assigns the reference objects w of keyframes using the interaction segments S. The finalized list of keyframes S_{LR} can be executed sequentially to perform the multistep manipulation task captured by the demonstration data.

3.4 Summary

Active Keyframe Learning (AKL) learns the least restrictive task specification from the observed demonstration end-effector poses, gripper states, and object positions augmented by human feedback elicited on the validity of online detected constrained regions of the demonstrated trajectory. It utilizes an interactive demonstration that consists of an online constraint detector and interactive communication between the robot and teacher to learn the human intent behind demonstrated constraint motion. The learned human intent is utilized in the offline keyframe learning algorithm to refine the keyframes learned from the interaction-based and constraint-based segmentation steps, thereby reducing the overconstrained natured of the learned keyframes. In this manner, AKL allows us to learn the least restrictive task specification as an ordered sequence of keyframes that capture the physical interactions and move-in-line constraints from a single demonstration. In the next chapter, we evaluate the performance of AKL against two state-of-the-art techniques in keyframe and constraint learning. We design a within-subjects study to obtain demonstrations for three tasks under each teaching method and analyze the keyframe accuracy, pose accuracy, constraint accuracy, teaching workload, system usability, and teaching efficiency to assess the benefits of AKL as a framework to teach robots tasks.

Chapter 4

Experimental Setup and Evaluation

This chapter presents the experimental setup and evaluation. It first describes the within-subjects study designed to evaluate our technique's performance, efficiency, workload, usability, and runtimes against two baseline methods on three tasks. We then discuss our findings under the results section. The source code for the experiments is available at: `https://github.com/thavishi/keyframe_detection`.

4.1 Baseline Methods

Our technique is evaluated against two baseline methods: Keyframe demonstrations (KD) [3] utilized in [53] and articulated constraints learning approach from [42]. The first baseline method, KD, explicitly allows users to define the keyframes during a demonstration. It is essential to observe that the loss of information due to a user-defined keyframe representation of a continuous trajectory makes learning the task's constraints challenging. Despite this inherent constraint learning limitation, we selected KD as a baseline due to its extensive use in keyframe learning and trajectory segmentation research. In our experiments, participants were instructed to define the least number of keyframes they think are required for successful robot learning. These keyframes are augmented with the participant's gripper commands to create the set of sequential keyframes used for comparison.

The articulated constraints learning approach, briefly introduced in chapter 2, was selected as our second baseline method based on its capability to learn articulated constraints from a single demonstration. In our experiments, we utilize this method to perform trajectory segmentation based on line constraints and list the boundary frames of the segments as constraint keyframes. In order to make the specification comparable, we augment the learned keyframes with interaction keyframes inferred by a participant's gripper commands. However, as constraints and interactions are learned in two independent processes, keyframes that simultaneously represent a constraint and an interaction will be inferred as two different keyframes. In our experiment, the sliding window size for this method was 5 cm, equal to our predefined distance value D_F. The TVD regular parameter was 0.6 as in [42]. Additionally, we used labeled demonstration samples to infer the error threshold value σ to be 2.01. We will refer to this modified articulated constraints learning baseline as mACL for convenience.

4.2 Measures

The following measures are utilized to quantify participant performance. For convenience, we will refer to the sequence of keyframes learned by our technique or a baseline method as learned keyframes, and the actual least restrictive task specification for each task, defined as in section 3.2, as the ground truth.:

Keyframe Accuracy: The accuracy of the keyframes, learned quantified by the Intersection-over-Union (IoU) measure computed between the learned keyframes and the ground truth.

Pose Accuracy: The correctness of the poses of the keyframes learned, measured as the IoU measure computed between learned poses and the ground truth.

Constraint Accuracy: The accuracy of the constraints, calculated as the IoU measure computed between the length of the learned constraints and the length of the true constraints.

Teaching Workload: The subjective workload on a participant using the system, measured by the NASA-TLX workload scale [29].

Teaching Efficiency: The efficiency of teaching a task, measured by dividing keyframe accuracy by teaching workload.
Method Success Rate: The rate of success of a method when used to teach tasks, computed as the number of successful trials divided by the total number of trials. In our work, a trial refers to a single demonstration of a task and is considered successful if the learned task specification is equal to the ground truth.

4.3 Hypotheses

AKL is designed to learn the least restrictive task specification by assigning keyframes to the boundaries of interaction and constraint segments, combining any interaction and constraint keyframes that coincide in Cartesian space, and utilizing human feedback to reduce constraint errors. In contrast, KD utilizes user-defined keyframes that are highly dependent on the user's knowledge of what constitutes the least restrictive task specification. On the other hand, mACL cannot combine coinciding interaction and constraint keyframes and does not account for the possibility of unintentionally over-constrained demonstrations. Thus, we hypothesize:
Hypothesis 1: Keyframe accuracy will be higher in our technique than in both the baselines.

The poses of the learned keyframes allow us to examine the accuracy of the task specification learned unaffected by the learned interaction and constraint labels. As AKL learns the least restrictive task specification, we hypothesize:
Hypothesis 2: Pose accuracy will be higher in our technique than in both the baselines.

KD learns only user-defined keyframes and cannot learn move-in-line constraints, and therefore, is not included in our constraint accuracy comparisons. Although AKL and mACL use least square model fitting and TVD to learn constraints offline, AKL

utilizes the latent constraint states inferred online augmented with human feedback to reduce constraint errors during offline learning. Furthermore, as constraint segmentation is performed on interaction segments, the boundaries of the learned constraints are more accurate. Thus we hypothesize:
Hypothesis 3: Constraint accuracy will be higher in our technique than in the mACL baseline.

Teaching workload allows us to examine the subjective workload when using the system to teach a robot. As AKL employs a natural and intuitive teaching interface, the interactive demonstration framework, where users provide a continuous demonstration while conversing with the robot, we hypothesize:
Hypothesis 4: Teaching workload will be lower in our technique than in both the baselines.

As AKL learns the least restrictive task specification and has an intuitive teaching interface, we hypothesize:
Hypothesis 5: Teaching efficiency will be more significant in our technique than in both the baselines.

4.4 Experimental Design

The experiment was designed to collect kinesthetic demonstrations of three manipulation tasks: (1) a pick and place task having no constraints, (2) an inspection task having an explicit constraint, and (3) an assembly task having an implicit constraint, as shown in Fig. 4-1. Here, explicit constraints refer to the constraints that are explicitly written in the descriptions of a task provided to a participant, whereas implicit constraints are not explicitly written in the task description despite being required for the correct execution of the task. The pick and place task requires the participant to move two objects from one shelf to another. Its least restrictive task specification consists of four interaction keyframes, one for every direct (grasp ob-

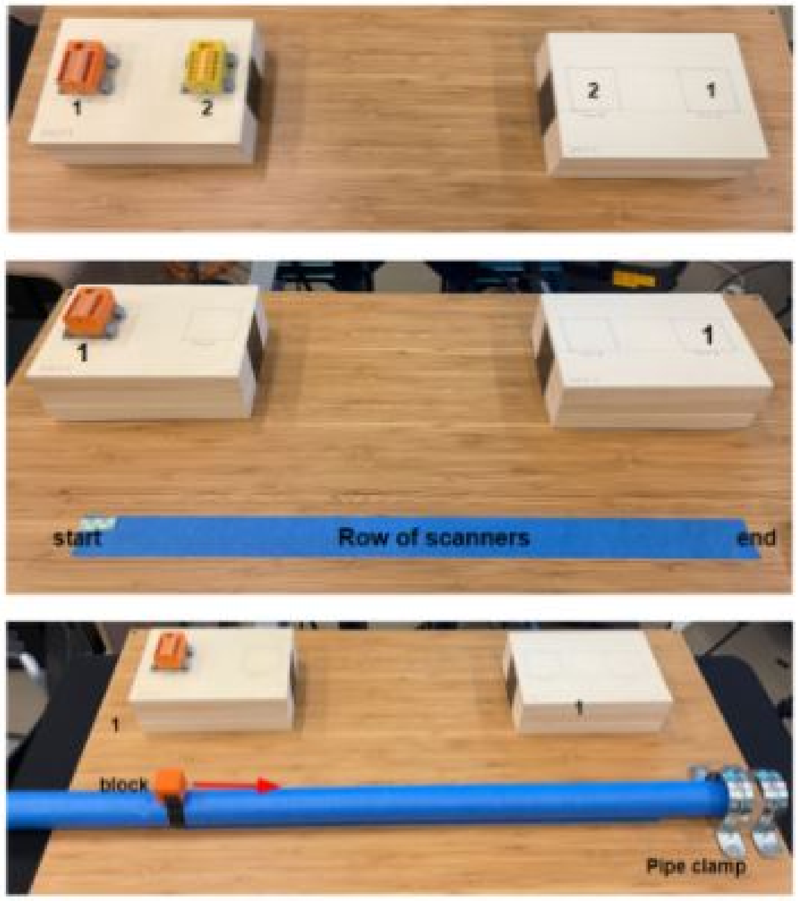


Figure 4-1: Task setup for the three tasks, pick and place task (top), inspection task (middle), and assembly task (bottom).

ject) and indirect (place object) interaction. The inspection task involves moving an object over a row of scanners and then placing the object on the second shelf. Its least restrictive task specification consists of two interaction keyframes for the grasp and placing of the object and two constraint keyframes representing the straight-line motion over the row of scanners. This move-in-line constraint is explicitly mentioned in the task description as *"move in a line over the row of scanners"*. The assembly task requires participants to move an object from one shelf to the other and then slide the pipe through the clamp. The pipe sliding step generates a straight-line motion which must be encoded in the task specification as a move-in-line constraint. However, the instruction provided to participants, *"slide the pipe through the clam"* does not explicitly mention that it requires straight-line motion. The assembly task has four keyframes in its least restrictive task specification, two interaction keyframes for the pick and place of the object and two combined keyframes that simultaneously represent the boundaries of the straight-line motion and the grasp and release of the pipe.

We utilized a Franka Emika Panda robot arm coupled with an Ubuntu 16.04 computer running Robot Operating System (ROS) Indigo to collect kinesthetic demonstrations. Task descriptions and robot queries were displayed on a graphical user interface (GUI) while answers were given using speech (*"yes"*, *"no"*, and *"I don't know"*). Voice commands were also used to to convey the start (*"start"*) and end (*"stop"*) of demonstrations, gripper commands (*"open"* and *"close"*), and user-defined keyframe poses (*"record"*). In our experiment, the distance value D_F parameter and the TVD regular parameter were defined as 5cm and 0.6. The algorithm inferred all other parameters of our technique from the participants' calibration and demonstration data.

The experiment began with a demographic questionnaire followed by a training session designed to allow participants to familiarize themselves with the kinesthetic teaching setup. The participants then entered the calibration phase, where they were asked to move the robot along a predefined line and predefined circle. Upon completing the calibration phase, participants began the primary phase, performing nine tasks (three modes, AKL, KD, and ACL, with three tasks per mode). The order of teaching modes was balanced between subjects using a Latin square design. We utilized a single Latin square with three rows to balance the 12 participants to analyze ordering effects. After performing each task, participants responded to the NASA-TLX questionnaire [29], and after each teaching mode, they rated the ten statements of the System Usability Scale (SUS) [6] on a 7-point Likert scale as in [25]. Participants concluded the experiment by responding to an exit questionnaire allowing open-ended comments on the experiment and their user experience for the three different teaching modes.

4.5 Analysis

4.5.1 Linear mixed-effects model analysis

The task specifications learned and the subjective workload reported were analyzed using MATLAB's native linear mixed-effects modeling function (fitlme). The linear mixed-effects models designed to investigate the hypotheses were formulated as follows:

$DV \sim Age + Sex + RobotEx + JoystickCEx + Mode + Task + (1|Participant) + (1|Participant{:}Order)$

DV represents the dependent variable being analyzed under each hypothesis and is keyframe accuracy, pose accuracy, constraint accuracy, teaching workload, and teaching efficiency for hypotheses 1 - 5, respectively. The independent variables included in the model are as follows:

- *Age*: The age of a participant in years.
- *Sex*: A participant's sex as reported in the demographic questionnaire. Female was considered the reference category in the models, while Male was an indicator variable. Our experiment consisted of six female and six male participants.
- *RobotEx*: A participant's experience interacting with robots, represented as a binary variable, where no experience was zero and any experience handling robots was one. Five participants had experience working with robots.
- *JoystickCEx*: A participant's experience using a joystick controller as reported on a seven-point Likert scale.
- *Mode*: The method used to collect demonstrations. KD and mACL were indicator variables, while AKL was the reference category.
- *Task*: The task being demonstrated. The pick and place task was the model's

reference variable, while the inspection and assembly tasks were indicator variables.

- *Participant*: The ID number assigned to each participant.
- *Order*: The chronological order of the teaching modes.

The random effects were modeled using *Participant* as a grouping variable and *Order* as a nested grouping variable within *Participant*. The remaining independent variables were fixed effects in our linear mixed-effects models.

4.5.2 System Usability Scale analysis

The System Usability Scale is designed to obtain a participant's subjective rating of the overall system usability. It consists of ten statements targeting various aspects of system usability, such as system consistency, complexity, and ease of use. Each statement is rated with a 7-point Likert-type scale, and an overall usability score ranging from 0 (low usability) to 100 (high usability) is computed as in [25]. These Likert items and the overall usability scores were analyzed using MATLAB's Wilcoxon Rank-Sum test (ranksum), and the results will be discussed in the following section.

4.6 Results

In this section, we will first compare the method success rates for each task and then examine the results of the linear mixed-effects model analyses. For convenience, the abbreviations AKL, KD, and mACL refer to our proposed method, and the baseline methods, keyframe demonstration, and modified articulated constraint learning, respectively. Additionally, the pick and place, inspection, and assembly tasks will be abbreviated as Task1, Task2, and Task3. The significance of our findings is based on the p-values of the estimated mixed-effects model coefficients for the fixed effects, and a significant coefficient implies that a change in the value of that effect significantly impacts the model's dependent variable. In our discussion, 'significant' refers to $p < 0.05$ while 'highly significant' refers to $p < 0.001$.

Table 4.1: Task specification learning success rates as percentages. Case(a) presents success rates as defined in section 4.2 and Case(b) reports success rates adjusted for algorithmic limitations. AKL achieved higher success rates for all three tasks compared to the baselines under both cases.

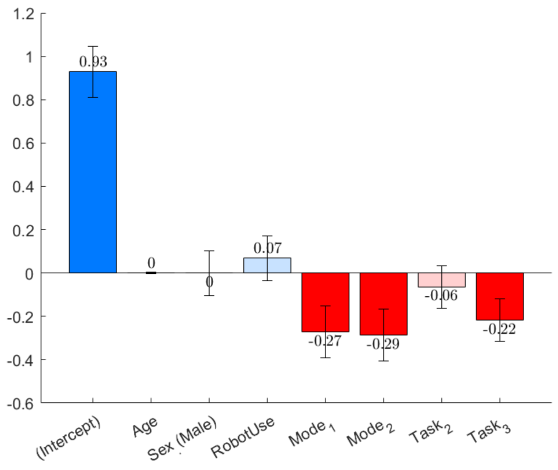
	KD		mACL		AKL
	Case(a)	Case(b)	Case(a)	Case(b)	
Task1	50.0	50.0	33.3	33.3	66.7
Task2	0.0	33.3	41.7	41.7	75.0
Task3	0.0	41.7	0	16.7	58.3
Overall	13.9	41.7	25.0	33.3	66.7

4.6.1 Task specification learning success rates

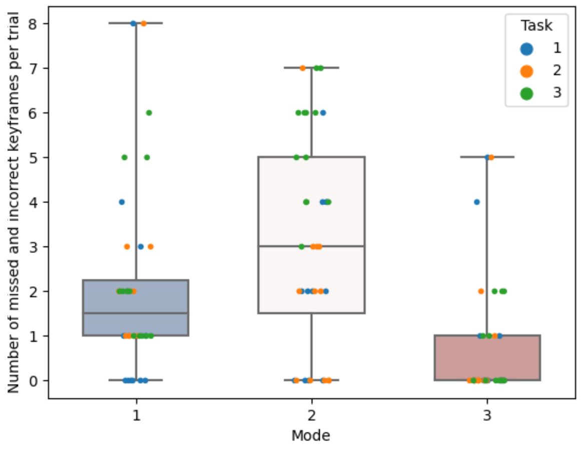
Table 4.1 presents the method success rates, as percentages, for each task under two cases: (a) where the success of a trial is defined as in section 4.2, and (b) where the limitations of the algorithms are considered when determining the success of a trial.

Under case (a), AKL produced higher success rates than both KD and mACL for all three tasks, suggesting that more participants could teach the robot the correct task specification using AKL than the baseline methods. KD reports zero successful trials for Task2 and Task3 as this baseline method cannot infer the constraints. Similarly, as mACL treats interaction keyframe learning and constraint keyframe learning as two separate processes in Task3, it generates four separate keyframes instead of the two keyframes representing both an interaction and a constraint boundary. This results in zero successful trials for Task3 under mACL.

Under case (b), we compare the AKL success rates to the baseline success rates computed for the best case of mitigating the inherent limitations of the techniques. For KD, the inability to learn constraints can be mitigated by augmenting the algorithm with a technique that adds constraint labels to the learned keyframes. To obtain the best case results, we assume that all added constraint labels are correct, resulting in 33.3 and 41.7 percent success rates for Task2 and Task3. mACL is augmented with a process that combines interaction and constraint keyframes to mitigate



(a)



(b)

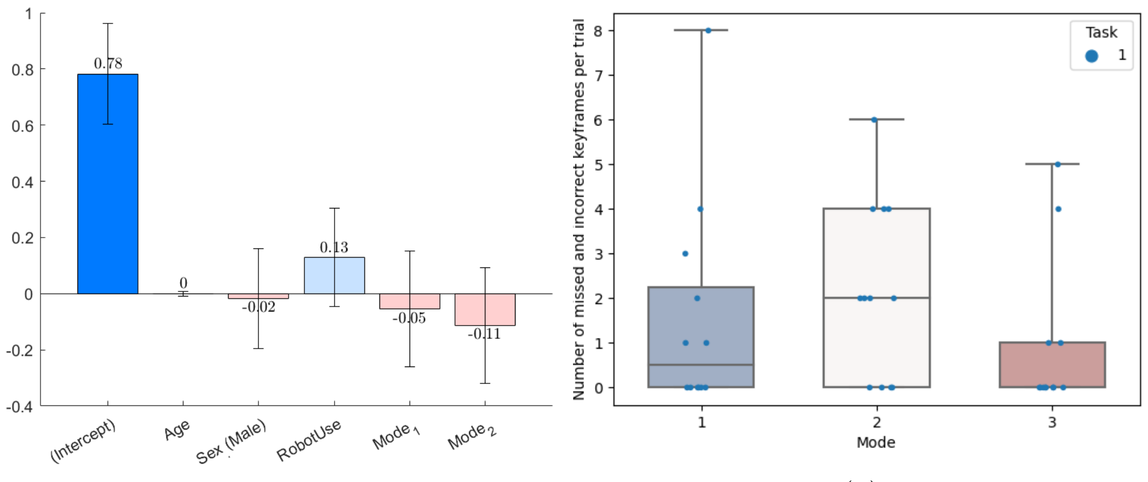
Figure 4-2: Results for keyframe accuracy analysis. Figure 4-2a presents the estimated coefficients of the linear mixed-effect model for keyframe accuracy, with 95% confidence intervals. Blue and red bars represent improved and reduced performance. Dark colored bars indicate highly significant effects ($p < 0.05$), light colored bars indicate non-significant effects. Figure 4-2b illustrates the distributions of the number of missed and incorrect keyframes per trial for each mode.

its limitation in Task3. By assuming that all combinations performed by this process are correct, we obtain the best case success rate of 16.7 percent. Nevertheless, the success rates reported by AKL are higher than both baselines in all three tasks for case (b).

4.6.2 Hypothesis 1: Keyframe Accuracy

The linear mixed-effects model analysis for keyframe accuracy, shown in figure 4-2a results in highly significant negative coefficients for KD (Mode_1) and mACL (Mode_2) factors. As negative coefficients correspond to poorer keyframe accuracy, these results suggest that, compared to the reference category AKL, KD and mACL display lower keyframe accuracy, supporting Hypothesis 1.

Figure 4-2b illustrates the distribution of the total missed and incorrect keyframes per trial for each mode of teaching, KD, mACL, and AKL (Mode 1, 2, and 3, respectively). Similar to the linear mixed-effects model results, KD and mACL have a higher median than AKL, corresponding to a lower average keyframe accuracy. It is



(a) (b)

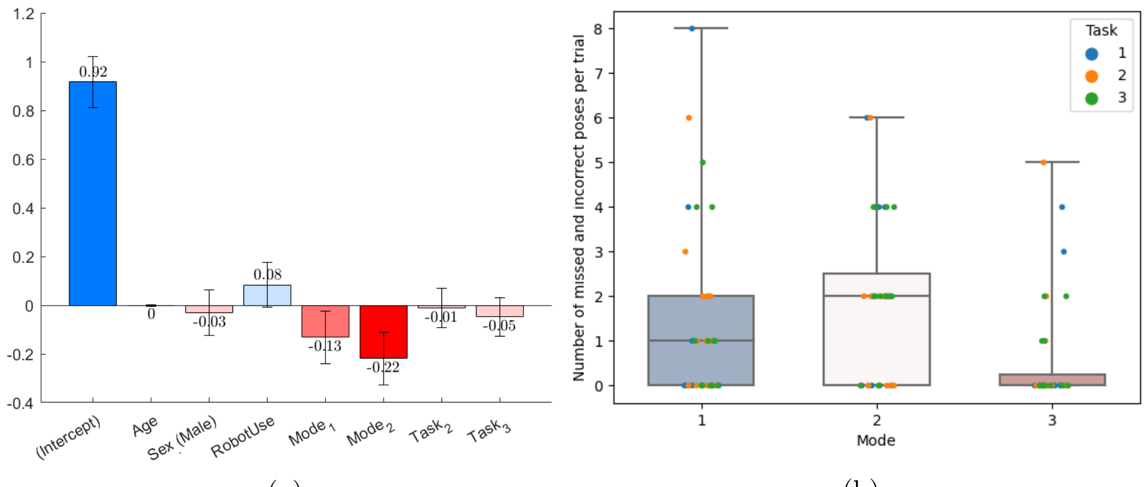
Figure 4-3: Results for keyframe accuracy analysis on Task1 data. Figure 4-3a presents the estimated coefficients of the linear mixed-effect model for keyframe accuracy for Task1, with 95% confidence intervals. Blue bars represent improved accuracy and red bars represent reduced accuracy. Dark colored bars indicate highly significant effects ($p < 0.01$), light colored bars indicate non-significant effects. Figure 4-3b illustrates the distributions of the number of missed and incorrect keyframes per trial for Task1 data for each mode.

interesting to note that the limitations of KD and mACL discussed in section 4.6.1 are evident by the absence of Task2 and Task3 data points at the zero level for KD and the absence of Task3 data points at the zero level for mACL. Therefore the highly significant decrease in keyframe accuracy for KD and mACL can be partially explained by the inherent limitations of the two techniques.

In order to examine the keyframe accuracy unaffected by the limitations of KD and mACL, we conducted a linear mixed-effect model analysis on Task1 data, shown in figure 4-3a. Although the results are not significant, the KD and mACL factors have negative coefficients indicating a decrease in keyframe accuracy compared to AKL. Furthermore, figure 4-3b suggests similar findings by having higher missed and incorrect keyframe per trial medians for KD and mACL than AKL.

4.6.3 Hypothesis 2: Pose Accuracy

The results of the linear mixed-effects model analysis on the effects of the teaching mode on the pose accuracy, as shown in figure 4-4a, indicate a significant negative



(a) (b)

Figure 4-4: Results for pose accuracy analysis. Figure 4-4a presents the estimated coefficients of the linear mixed-effect model for pose accuracy, with 95% confidence intervals. Blue bars represent improved accuracy and red bars represent reduced accuracy. Dark colored bars indicate highly significant effects ($p < 0.01$), medium colored bars indicate significant effects ($p < 0.05$), and light colored bars indicate non-significant effects. Figure 4-2b illustrates the distributions of the number of missed and incorrect poses per trial for each mode.

effect from KD (Mode_1) factor and a highly significant adverse effect from mACL (Mode_2) factor, when compared to the reference category AKL. As negative coefficients in this model correspond to a decrease in pose accuracy, hypothesis 2 is supported.

The distributions of the missed and incorrect keyframe poses per trial for each teaching mode, KD, mACL, and AKL, indicate a similar correlation between the teaching mode and pose accuracy. Here, as keyframe poses are unaffected by constraint labels, the constraint learning limitation of KD does not affect its distribution. Furthermore, the specification learned by mACL for Task3 was permitted six keyframes to allow for meaningful comparisons. In other words, the interaction and constraint keyframe combination limitation of mACL does not affect the number of missed and incorrect keyframe poses recorded. As illustrated in figure 4-4b, KD and mACL have higher missed and incorrect keyframe poses per trial medians and larger interquartile ranges than AKL. This suggests that AKL learned task specifications with a higher pose accuracy than that learned using KD and mACL.

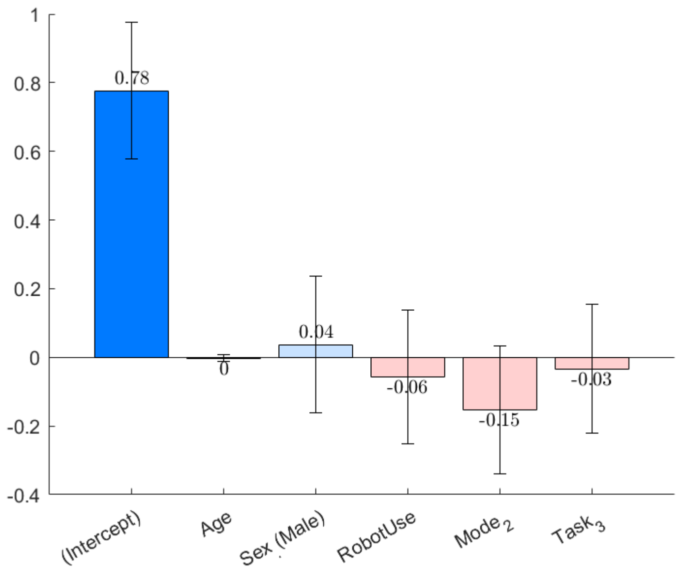


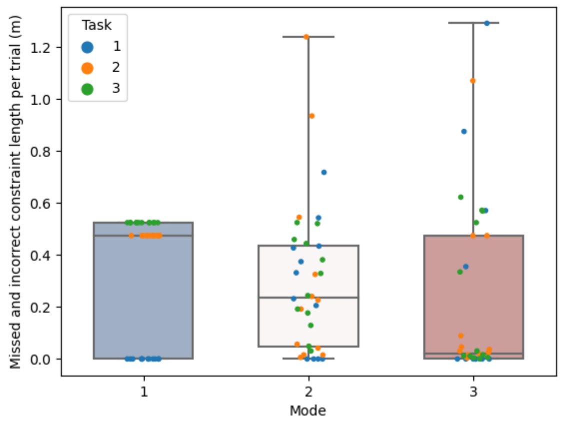
Figure 4-5: The estimated coefficients of the linear mixed-effect model for constraint accuracy, with 95% confidence intervals. Blue bars represent improved accuracy and red bars represent reduced accuracy. Dark colored bars indicate highly significant effects ($p < 0.01$) and light colored bars indicate non-significant effects.

4.6.4 Hypothesis 3: Constraint Accuracy

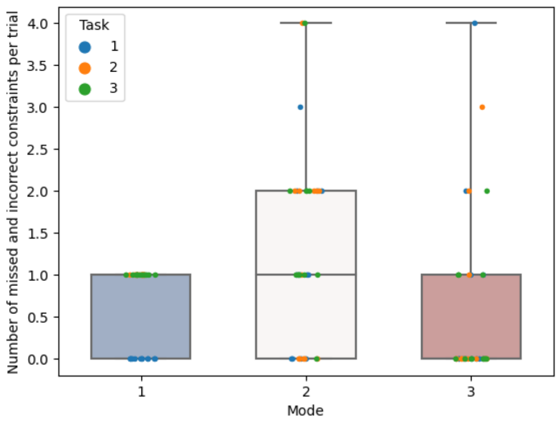
Table 4.2: Number of constraint errors for AKL and the impact of participant's feedback on these constraint errors. 14/19 of the constraint errors are due to incorrect feedback. 20 incorrectly detected constraints were corrected by participant's feedback.

Total number of constraint errors	19
Constraint errors due to incorrect feedback	14
Constraint errors avoided due to correct feedback	20

As Task1 is an unconstrained task, the intersection over union measure related to the length of the learned constraints is always zero. Therefore, we considered only Task2 and Task3 data in our linear mixed-effects model analysis for constraint accuracy. For hypothesis 3 to be supported, the coefficient of the mACL (Mode_1) factor in our linear mixed-effects model, illustrated in figure 4-5, must indicate an adverse effect on constraint accuracy. However, although the mACL coefficient is negative, indicating lower constraint accuracy than AKL, the result was insignificant



(a)



(b)

Figure 4-6: Distributions of constraint errors. Figure 4-6a illustrates the distributions of the missed and incorrect constraint lengths per trial for each mode. Figure 4-6b shows the distributions of the number of missed and incorrect constraints per trial for each mode.

(p-value = 0.10).

Figure 4-6 illustrates the distributions of the total missed and incorrect constraint length per trial and the number of missed and incorrect constraints detected per trial (left and right, respectively). The results for KD show zero incorrect constraints for Task1 and one missed constraint per trial for Task2 and Task3, reflecting the constraint learning limitation of the baseline. This implies that the data points for KD's missed constraint lengths give the length of the actual constraints of Task2 and Task3. In the constraint length error graph, although AKL appears to have a wider interquartile range for the total missed and incorrect constraint lengths per trial, the median (2.18 cm) is much lower than the mACL median (23.65 cm). In other words, the best fifty percent of task specifications learned using AKL had less than 2.18 cm of constraint length errors, whereas that for mACL were spread over 23.65cm. The incorrect constraint lengths include the misalignments between the actual constraint and learned constraint due to variance in the demonstrations provided. For instance, under AKL in Task2, the start of the constrained region had a standard deviation of 1.60cm. Therefore, examining the number of missed and incorrect constraints per trial will further insight into our constraint accuracy comparison. AKL reports a lower interquartile range and median for the number of constraint errors per trial, suggesting higher constraint accuracy than mACL. Although AKL and mACL display equal ranges in this graph, the total number of constraint errors for mACL (46 errors) is much higher than for AKL (19 errors). It is interesting to note that 73.7% (14 errors) of the constraint errors for AKL were due to incorrect feedback given to the queries posed by the robot, as given in Table 4.2. Furthermore, human feedback provided during demonstrations rectified 20 erroneously detected constraint regions suggesting that AKL could prevent 51.2% of constraint errors by leveraging human feedback. These findings suggest that human feedback aids in reducing constraint errors, and improving human feedback can further improve the constraint accuracy of AKL.

4.6.5 Hypothesis 4: Teaching Workload

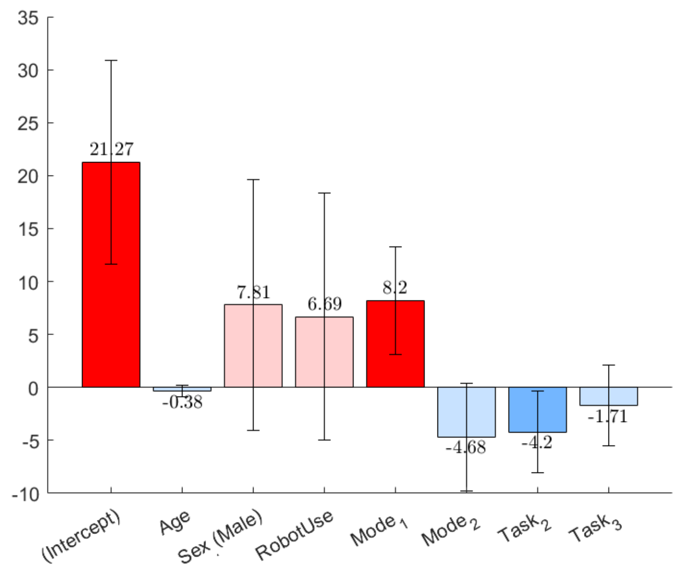


Figure 4-7: The estimated coefficients of the linear mixed-effect model for teaching workload, with 95% confidence intervals. Blue bars represent reduced workload and red bars represent increased workload. Dark colored bars indicate highly significant effects ($p < 0.01$), medium colored bars indicate significant effects ($p < 0.05$), and light colored bars indicate non-significant effects.

In order for hypothesis 4 to be supported, the coefficients of KD (Mode$_1$) and mACL (Mode$_2$) in our linear mixed-effects model for teaching workload, illustrated in figure 4-7, should be positive as positive coefficients correspond to an increase in

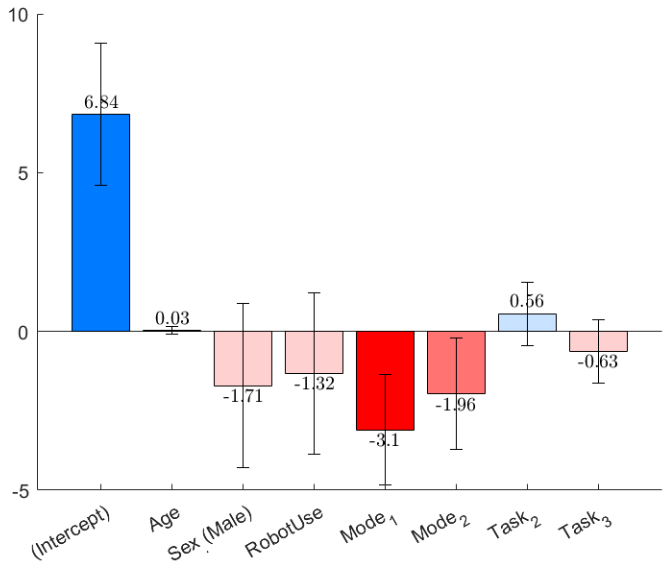


Figure 4-8: The estimated coefficients of the linear mixed-effect model for teaching efficiency, with 95% confidence intervals. Blue bars represent improved efficiency and red bars represent reduced efficiency. Dark colored bars indicate highly significant effects ($p < 0.01$), medium colored bars indicate significant effects ($p < 0.05$), and light colored bars indicate non-significant effects.

teaching workload. As hypothesized, the KD factor resulted in a highly significant increase in teaching workload. However, the coefficient of the mACL factor was found to be negative, indicating a decrease in workload, leaving our hypothesis unsupported. These results suggest that although switching from user-defined keyframes to a continuous demonstration framework reduces the workload, users find the clarification questions asked during a demonstration to increase the workload of the teaching task.

4.6.6 System Usability Scale

Participants reported a lower overall usability score for KD (M = 67.5, p-value = 0.21) and a significantly higher score for mACL (M = 87.1, p-value = 0.03, p < 0.05) as compared to AKL (M = 77.1). Similar to our findings for hypothesis 4, these scores suggest that participants find an uninterrupted continuous teaching framework more user-friendly than one augmented with clarification questions posed by the student robot.

4.6.7 Hypothesis 5: Teaching Efficiency

For KD (Mode$_1$), the effect on the teaching efficiency measure is highly significant in the direction of poorer efficiency, as indicated by the negative coefficient ($p < 0.01$) of the KD factor shown in figure 4-8. This outcome is not surprising given the highly significant reduction in performance for keyframe accuracy and teaching workload displayed in the analyses for hypotheses 1 and 4. On the other hand, mACL (Mode$_2$) presented a reduced teaching workload compared to the reference category AKL. However, due to the highly significant adverse effect on keyframe accuracy, mACL reported a significant reduction in teaching efficiency. These results suggest that AKL displayed a higher teaching efficiency than KD ($p < 0.05$) and mACL ($p < 0.01$), supporting hypothesis 5.

4.6.8 Runtimes

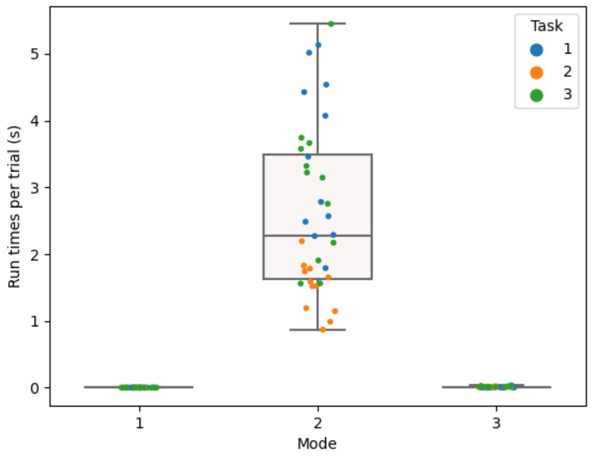


Figure 4-9: Distributions of runtimes per trial for three modes.

The number of least square fittings run for mACL increases linearly with the length of the demonstrated trajectory. Furthermore, for each least square fitting, the fitting errors of all trajectory frames are computed to obtain the inners ratio implying a complexity of n^2. On the other hand, the number of least square fittings run for AKL is equal to the number of constraint regions C_r detected, independent of the trajectory length, and the fitting errors are calculated only for the respective trajectory segments. This implies that AKL has a linear complexity of n, suggesting lower runtimes than mACL. As expected, the runtimes for mACL (Mode_2) are much higher than for AKL (Mode_3), as illustrated in figure 4-9. In fact, there is no overlap between the distribution for the mACL runtimes and the AKL runtime distribution. Furthermore, the average runtime per trial for mACL (2.64 s) is approximately 200 times the average runtime per trial for AKL (0.011 s). AKL appears to have a similar runtime distribution as KD (Mode_1) due to the scale used in figure 4-9. However, the runtimes for KD are negligible as users directly provide the keyframes as demonstration data suggesting higher runtimes for AKL than for KD. The average runtime per trial for AKL (0.011 s) and KD (0.001s) support this.

4.7 Discussion

We designed a within-subject study with three tasks and two baselines, KD and mACL, to evaluate five hypotheses on the keyframe accuracy, pose accuracy, constraint accuracy, teaching workload, and teaching efficiency of AKL and to compare the success rates, runtimes, and usability of AKL to KD and mACL.

The **success rates** for learning the least restrictive task specification for AKL (overall: 66.7%) were higher than that for KD (overall for case (a): 13.9%, overall for case (b): 41.7%) and mACL (overall for case (a): 25.0%, overall for case (b): 33.3%) under both cases as depicted in Table 4.1. Here, case (a) refers to the success rates for the baselines as defined in section 4.2, and case (b), the rates after accounting for the limitations of the baselines. The high success rates for AKL suggest that participants were more successful at teaching the least restrictive task specification using AKL than KD or mACL.

The linear mixed-effects model analysis supported **Hypothesis 1 for keyframe accuracy** suggesting a highly significant increase in AKL's keyframe accuracy than KD ($p < 0.01$) and mACL ($p < 0.01$), as shown in figure 4-2a. A comparison of the distributions of the total missed and incorrect keyframes per trial for the three methods further support this claim (medians for AKL, KD, and mACL are 0, 1.5, and 3 keyframes, respectively), as shown in figure 4-2b. Additionally, we compared the keyframe accuracy of the three modes for Task1 data to examine the effect of the inherent limitations of the baselines on our results. Although not significant, the results from the linear mixed-effect model results indicated a keyframe accuracy gain under AKL than KD and mACL, agreeing with the comparison of the distributions of the total missed and incorrect keyframes per trial for Task1 data (medians for AKL, KD, and mACL are 0, 0.5, and 2 keyframes, respectively), as illustrated in figure 4-3.

Hypothesis 2 for pose accuracy was supported by the linear mixed-effects model analysis, suggesting that the pose accuracy for AKL was significantly higher than for KD ($p < 0.05$) and mACL ($p < 0.01$), as shown in figure 4-4a. A comparison of the distributions of the total missed and incorrect poses per trial for the three modes further reinforce this claim (medians for AKL, KD, and mACL are 0, 1, and 2 poses, respectively), as shown in figure 4-4b.

Although the linear mixed-effects model analysis for **Hypothesis 3 for constraint accuracy** indicated an increase in constraint accuracy for AKL compared to mACL, the results were insignificant, as depicted in figure 4-5. However, the comparisons of distributions of the missed and incorrect constraint length per trial and the distributions of the total missed and incorrect constraints per trial for the three modes, illustrated in figure 4-6, suggest improved constraint accuracy for AKL compared to mACL. For instance, fifty percent of task specifications learned using AKL had less than 2.18 cm of constraint length errors, whereas that for mACL were spread over 23.65cm, and the total number of constraint errors for AKL (19 errors) was much lower than for mACL (46 errors). Furthermore, we discovered that 73.7% of the constraint errors for AKL were due to incorrect human feedback and that AKL was able to prevent 51.2% of constraint errors through human feedback. These findings suggested the importance of leveraging human feedback to reduce constraint errors and that increasing the accuracy of the given feedback will directly improve the constraint accuracy for AKL.

The linear mixed-effects model analysis on the NASA-TLX workload scores partially supported **Hypothesis 4 for teaching workload** by indicating a highly significant increase in workload for KD ($p < 0.01$) and an insignificant decrease in workload for mACL ($p > 0.05$), compared to AKL, as shown in figure 4-7. These results suggest that switching from user-defined keyframes to a continuous demonstration framework significantly reduced the workload, whereas interactive communication introduced to continuous demonstrations increased the teaching workload.

Similar to teaching workload observations, the **System Usability Scale scores** reported were lower for KD (M = 67.5, p-value =0.21) and a significantly higher for mACL (M = 87.1, p-value = 0.03, p < 0.05) compared to AKL (M = 77.1). These scores suggest that participants found AKL usability to be lower than mACL and higher than KD.

The linear mixed-effects model analysis supported **Hypothesis 5 for teaching efficiency** suggesting a significant increase in AKL's teaching efficiency than KD ($p < 0.01$) and mACL ($p < 0.05$), as shown in figure 4-8. These results suggest that although AKL displayed increased teaching workload and lower usability scores, AKL improves the efficiency of teaching tasks to a robot compared to KD and mACL.

We compared the **runtime** of AKL to the baselines, as shown in figure 4-9 and, as expected, found that although AKL is slower than KD, as KD does not require keyframe learning, AKL (average runtime of 0.011s per trial) is 200 times faster than mACL (average runtime of 2.64s per trial).

Chapter 5

Conclusion and Future Work

Although recent advances in robotics enable the automation of manual tasks in manufacturing, integrating robots into a factory remains time and resource intensive, as it requires conventional robot programming and robot experts. Even small changes to the task will require reprogramming and can result in high reintegration costs. In order to increase the feasibility of robot integration into industrial processes, the programming of robots must be easily accessible to domain experts with little to no experience in robotics. This requires the design of an intuitive approach to teaching robots that allow non-experts to teach tasks in a time-efficient manner without robot programming.

This thesis presents Active Keyframe Learning (AKL), a proof-of-concept system for learning physical interactions and move-in-line constraints of a task from a single demonstration. We learn the task as an agent-independent ordered sequence of interaction and constraint keyframes that can be given as input to a collision avoidance enabled motion planner for task execution. As multiple such specifications can exist for a task, we learn the least restrictive task specification to maximize the flexibility given to the motion planner during task execution. We prove that the least restrictive task specification, in the set of keyframe-encoded specifications correctly describing the task, is the specification with the least number of keyframes.

As learning from a single demonstration can often lead to over-constrained task specifications, we design an interactive demonstration framework that performs online move-in-line constraint inference and allows human-robot communication to learn human intent for the demonstrated constrained motion. We perform online constraint inference by first transforming demonstration data to straight-line fitting errors, then computing a switching probability utilizing a logistic regression model trained on constrained and unconstrained motion classification, and finally, inferring the latent constraint state by passing the switching probability through an energy tank. The inferred latent state informs the interactive communication dialogue where the robot verifies the validity of the inferred latent state and provides suggestions to the teacher to improve their demonstration technique.

To learn the interaction and constraint keyframes from a single demonstration, we develop an offline keyframe learning algorithm that performs interaction-based and constraint-based segmentation. We utilize the latent constraint states and human feedback from the interactive demonstration to remove the unnecessary constraint segments and learn the least restrictive task specification defined by the keyframes at the boundaries of the interaction and constraint segments.

We evaluated the performance of our proposed framework, Active Keyframe Learning (AKL), against two baseline methods, keyframe demonstration (KD) [3, 53] and articulate constraints learning approach, from [42], augmented with interaction learning (mACL). We designed a within-subject study with three multistep tasks to evaluate the success rates, keyframe accuracy, pose accuracy, constraint accuracy, teaching workload, usability, and teaching efficiency of AKL compared to the two baselines. We discovered that AKL displayed the highest success rates in learning the least restrictive task specification and significantly improved the keyframe and pose accuracy. Although the results for the hypothesis for constraint accuracy were insignificant, fifty percent of task specifications learned using AKL had less than 2.18 cm of constraint length errors, whereas that for mACL spanned 23.65cm. Additionally, the total number of constraint errors for AKL (19 errors) was much lower than for mACL (46 errors), suggesting an increase in constraint accuracy. We learned that 73.7 % of the constraint errors for AKL were due to incorrect human feedback emphasizing the ability to further improve constraint accuracy by increasing the feedback correctness.

Furthermore, we found that AKL could prevent 51.2% of constraint errors through human feedback highlighting the positive impact of our proposed interactive demonstration framework on constraint accuracy. Our findings on workload and usability suggested that AKL was more userfriendly and resulted in a significantly lower workload than KD. However, AKL showed increased workload and decreased usability scores compared to mACL. Although the interactive communication aspect of AKL was found to increase workload and reduce usability, the overall teaching efficiency of AKL was significantly higher than KD and mACL, demonstrating the significant benefit of using AKL to teach tasks to robots.

Currently, this thesis is limited to learning physical interactions between task objects and the robot, essential for the task's success. Thus an extension to AKL would be learning non-physical interactions or complex physical interactions that cannot be inferred through gripper state observations. For example, interactions such as holding a mug under a coffee machine until full or screwing in a nail are complex interactions requiring richer keyframe definitions and robust visual inference.

Another future work direction would be extending the online constraint detector to infer additional hard geometric constraints such as orientation constraints or revolute constraints. As constraint inference utilizes fitting errors of a constraint model, theoretically, it can be extended to other parametrized constraints by having several online constraint detectors in parallel, one for each constraint. The detectors will have unique transformations to fitting errors based on the parameterized constraint model and an individual calibration step for the logistic regression model. However, it will be essential to examine the impact of the parallelization of multiple inference problems on the real-time nature of queries. Furthermore, the interactive communication dialogue must be redesigned to accommodate the additional queries, with particular attention given to the frequency of querying.

Although our work employs learner feedback as suggestions to the teacher to improve their demonstration technique during unconstrained motion, we assume that the learner is unaware of its impact on the teacher's mental model of the learner. However, it would be interesting to learn about this impact and understand how it would affect the demonstrations provided by the teacher to improve interactive communication. For instance, if the teacher understands and accepts the suggestion, they will only move in a straight line when it is essential for the task's success. Learning this improvement in the teacher's mental model of the learner can be utilized to reduce the frequency of querying for the teacher's intent. Therefore, an interesting future direction for our work is leveraging the teacher's mental model of the learner to inform human-robot communication.

Bibliography

[1]

[2] Pieter Abbeel and Andrew Y. Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the Twenty-First International Conference on Machine Learning*, ICML '04, page 1, New York, NY, USA, 2004. Association for Computing Machinery.

[3] Baris Akgün, M. Cakmak, Karl Jiang, and A. Thomaz. Keyframe-based learning from demonstration. *International Journal of Social Robotics*, 4:343–355, 2012.

[4] Baris Akgun and Andrea Thomaz. Simultaneously learning actions and goals from demonstration. *Auton. Robots*, 40(2):211–227, feb 2016.

[5] Leopoldo Armesto, Jorren Bosga, Vladimir Ivan, and Sethu Vijayakumar. Efficient learning of constraints and generic null space policies. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1520–1526, 2017.

[6] Aaron Bangor, Philip T. Kortum, and James T. Miller. An empirical evaluation of the system usability scale. *International Journal of Human–Computer Interaction*, 24(6):574–594, 2008.

[7] Dmitry Berenson, Siddhartha Srinivasa, and James Kuffner. Task space regions: A framework for pose-constrained manipulation planning. *The International Journal of Robotics Research*, 30(12):1435–1460, 2011.

[8] Erdem B, Malayandi Palan, Nicholas C. Landolfi, Dylan P. Losey, and Dorsa Sadigh. Asking easy questions: A user-friendly approach to active reward learning. In Leslie Pack Kaelbling, Danica Kragic, and Komei Sugiura, editors, *Proceedings of the Conference on Robot Learning*, volume 100 of *Proceedings of Machine Learning Research*, pages 1177–1190. PMLR, 30 Oct–01 Nov 2020.

[9] Erdem Biyik and Dorsa Sadigh. Batch active preference-based learning of reward functions. *CoRR*, abs/1810.04303, 2018.

[10] Maya Cakmak, Crystal Chao, and Andrea L. Thomaz. Designing interactions for robot active learners. *IEEE Transactions on Autonomous Mental Development*, 2(2):108–118, 2010.

[11] Maya Cakmak and Andrea Thomaz. Active learning with mixed query types in learning from demonstration. 01 2011.

[12] Maya Cakmak and Andrea L. Thomaz. Designing robot learners that ask good questions. In *2012 7th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 17–24, 2012.

[13] Sylvain Calinon and Aude Billard. A probabilistic programming by demonstration framework handling constraints in joint space and task space. In *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 367–372, 2008.

[14] Glen Chou, Dmitry Berenson, and Necmiye Ozay. Learning constraints from demonstrations with grid and parametric representations. *The International Journal of Robotics Research*, 40(10-11):1255–1283, 2021.

[15] Glen Chou, Necmiye Ozay, and Dmitry Berenson. Learning constraints from locally-optimal demonstrations under cost function uncertainty. *IEEE Robotics and Automation Letters*, PP:1–1, 02 2020.

[16] Glen Chou, Necmiye Ozay, and Dmitry Berenson. Uncertainty-aware constraint learning for adaptive safe motion planning from demonstrations. In *CoRL*, 2020.

[17] Yuchen Cui and Scott Niekum. Active reward learning from critiques. In *ICRA*, pages 6907–6914, 2018.

[18] Peter Englert, Ngo Anh Vien, and Marc Toussaint. Inverse kkt: Learning cost functions of manipulation tasks from demonstrations. *The International Journal of Robotics Research*, 36(13-14):1474–1488, 2017.

[19] C. Eteke, D. Kebüde, and B. Akgün. Reward learning from very few demonstrations. *IEEE Transactions on Robotics*, pages 1–12, 2020.

[20] Paul Fearnhead. Exact and efficient bayesian inference for multiple changepoint problems. *Statistics and Computing*, 16(2):203–213, June 2006.

[21] Paul Fearnhead and Zhen Liu. On-line inference for multiple changepoint problems. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 69(4):589–605, 2007.

[22] Paul Fearnhead and Zhen Liu. Efficient bayesian analysis of multiple changepoint models with dependence across segments. *Statistics and Computing*, 21, 10 2009.

[23] Chelsea Finn, Sergey Levine, and Pieter Abbeel. Guided cost learning: Deep inverse optimal control via policy optimization. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, ICML'16, page 49–58. JMLR.org, 2016.

[24] Tesca Fitzgerald, Ashok K. Goel, and Andrea Lockerd Thomaz. Representing skill demonstrations for adaptation and transfer. In *AAAI Fall Symposia*, 2014.

[25] Samir Yitzhak Gadre, Eric Rosen, Gary Chien, Elizabeth Phillips, Stefanie Tellex, and George Konidaris. End-user robot programming using mixed reality. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 2707–2713, 2019.

[26] Shixiang Gu, Ethan Holly, Timothy Lillicrap, and Sergey Levine. Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3389–3396, 2017.

[27] R. A. Gutierrez, V. Chu, A. L. Thomaz, and S. Niekum. Incremental task modification via corrective demonstrations. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1126–1133, 2018.

[28] Reymundo A. Gutierrez, Elaine Schaertl Short, Scott Niekum, and Andrea L. Thomaz. Towards online learning from corrective demonstrations, 2018.

[29] Sandra G. Hart and Lowell E. Staveland. Development of nasa-tlx (task load index): Results of empirical and theoretical research. In Peter A. Hancock and Najmedin Meshkati, editors, *Human Mental Workload*, volume 52 of *Advances in Psychology*, pages 139–183. North-Holland, 1988.

[30] M. Hasan, M. Warburton, W. C. Agboh, M. R. Dogar, M. Leonetti, H. Wang, F. Mushtaq, M. Mon-Williams, and A. G. Cohn. Human-like planning for reaching in cluttered environments. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7784–7790, 2020.

[31] Bradley Hayes and Brian Scassellati. Discovering task constraints through observation and active learning. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4442–4449, 2014.

[32] J. Huang, D. Fox, and M. Cakmak. Synthesizing robot manipulation programs from a single observed human demonstration. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4585–4592, 2019.

[33] Tariq Iqbal, Shen Li, Christopher Fourie, Bradley Hayes, and Julie A. Shah. Fast online segmentation of activities from partial trajectories. In *International Conference on Robotics and Automation, ICRA 2019, Montreal, QC, Canada, May 20-24, 2019*, pages 5019–5025. IEEE, 2019.

[34] Mahdi Khoramshahi and Aude Billard. A dynamical system approach for detection and reaction to human guidance in physical human–robot interaction. *Auton. Robots*, 44(8):1411–1429, nov 2020.

[35] George Konidaris, Scott Kuindersma, Roderic Grupen, and Andrew Barto. Robot learning from demonstration by constructing skill trees. *The International Journal of Robotics Research*, 31(3):360–375, 2012.

[36] Sanjay Krishnan, Animesh Garg, Sachin Patil, Colin Lea, Gregory Hager, Pieter Abbeel, and Ken Goldberg. Transition state clustering: Unsupervised surgical trajectory segmentation for robot learning. *The International Journal of Robotics Research*, 36(13-14):1595–1618, 2017.

[37] Andrey Kurenkov, Baris Akgun, and Andrea L. Thomaz. An evaluation of gui and kinesthetic teaching methods for constrained-keyframe skills. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3608–3613, 2015.

[38] Maria Kyrarini, Muhammad Abdul Haseeb, Danijela Ristic-Durrant, and Axel Graeser. Robot learning of industrial assembly task via human demonstrations. *Autonomous Robots*, 43, 01 2019.

[39] Sang Hyoung Lee, Il Hong Suh, Sylvain Calinon, and Rolf Johansson. Learning basis skills by autonomous segmentation of humanoid motion trajectories. In *2012 12th IEEE-RAS International Conference on Humanoid Robots (Humanoids 2012)*, pages 112–119, 2012.

[40] Sang Hyoung Lee, Il Hong Suh, Sylvain Calinon, and Rolf Johansson. Autonomous framework for segmenting robot trajectories of manipulation task. *Autonomous Robots*, 38:107–141, 2015.

[41] Changshuo Li and Dmitry Berenson. Learning object orientation constraints and guiding constraints for narrow passages from one demonstration. In Dana Kulić, Yoshihiko Nakamura, Oussama Khatib, and Gentiane Venture, editors, *2016 International Symposium on Experimental Robotics*, pages 197–210, Cham, 2017. Springer International Publishing.

[42] Yizhou Liu, Fusheng Zha, Lining Sun, Jingxuan Li, ManTian Li, and Xin Wang. Learning articulated constraints from a one-shot demonstration for robot manipulation planning. *IEEE Access*, PP:1–1, 11 2019.

[43] Bernard Michini and Jonathan P. How. Bayesian nonparametric inverse reinforcement learning. In Peter A. Flach, Tijl De Bie, and Nello Cristianini, editors, *Machine Learning and Knowledge Discovery in Databases*, pages 148–163, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[44] Anahita Mohseni Kabir, Changshuo Li, Victoria Wu, Daniel Miller, Benjamin Hylak, Sonia Chernova, Dmitry Berenson, Candace Sidner, and Charles Rich. Simultaneous learning of hierarchy and primitives for complex robot tasks. *Autonomous Robots*, 43, 04 2019.

[45] Anahita Mohseni-Kabir, Charles Rich, Sonia Chernova, Candace L. Sidner, and Daniel Miller. Interactive hierarchical task learning from a single demonstration. In *Proceedings of the Tenth Annual ACM/IEEE International Conference on Human-Robot Interaction*, HRI '15, page 205–212, New York, NY, USA, 2015. Association for Computing Machinery.

[46] C. Mueller, J. Venicx, and B. Hayes. Robust robot learning from demonstration and skill repair using conceptual constraints. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6029–6036, 2018.

[47] Carl Mueller, Aaquib Tabrez, and Bradley Hayes. Interactive constrained learning from demonstration using visual robot behavior counterfactuals. In *Proceedings of the Accessibility of Robot Programming and Work of the Future Workshop at RSS 2021*, 2021.

[48] Carl L. Mueller and Bradley Hayes. Safe and robust robot learning from demonstration through conceptual constraints. In *Companion of the 2020 ACM/IEEE International Conference on Human-Robot Interaction*, HRI '20, page 588–590, New York, NY, USA, 2020. Association for Computing Machinery.

[49] S. Niekum, S. Osentoski, C. G. Atkeson, and A. G. Barto. Online bayesian changepoint detection for articulated motion models. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1468–1475, 2015.

[50] Daehyung Park, Michael Noseworthy, Rohan Paul, Subhro Roy, and Nicholas Roy. Inferring task goals and constraints using bayesian nonparametric inverse reinforcement learning. In Leslie Pack Kaelbling, Danica Kragic, and Komei Sugiura, editors, *Proceedings of the Conference on Robot Learning*, volume 100 of *Proceedings of Machine Learning Research*, pages 1005–1014. PMLR, 30 Oct–01 Nov 2020.

[51] Sudeep Pillai, Matthew R. Walter, and Seth J. Teller. Learning articulated motions from visual demonstration. *CoRR*, abs/1502.01659, 2015.

[52] Claudia Pérez-D'Arpino and Julie A. Shah. Fast target prediction of human reaching motion for cooperative human-robot manipulation tasks using time series classification. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6175–6182, 2015.

[53] Claudia Pérez-D'Arpino and Julie A. Shah. C-learn: Learning geometric constraints from demonstrations for multi-step manipulation in shared autonomy. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4058–4065, 2017.

[54] Preeti Ramaraj. Robots that help humans build better mental models of robots. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(18):15730–15731, May 2021.

[55] Preeti Ramaraj, Charles L. Ortiz, and Shiwali Mohan. Unpacking human teachers' intentions for natural interactive task learning. In *2021 30th IEEE International Conference on Robot Human Interactive Communication (RO-MAN)*, pages 1173–1180, 2021.

[56] Harish Ravichandar, Athanasios S. Polydoros, Sonia Chernova, and Aude Billard. Recent advances in robot learning from demonstration. *Annual Review of Control, Robotics, and Autonomous Systems*, 3(1):297–330, 2020.

[57] Leonid I. Rudin, Stanley Osher, and Emad Fatemi. Nonlinear total variation based noise removal algorithms. *Physica D: Nonlinear Phenomena*, 60(1):259–268, 1992.

[58] Dorsa Sadigh, Anca Dragan, Shankar Sastry, and Sanjit Seshia. Active preference-based learning of reward functions. 07 2017.

[59] Lindsay Sanneman, Christopher Fourie, and Julie A. Shah. The state of industrial robotics: Emerging technologies, challenges, and key research directions, 2020.

[60] Ankit Shah, Pritish Kamath, Julie A Shah, and Shen Li. Bayesian inference of temporal task specifications from demonstrations. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[61] Ankit Shah and Julie Shah. Interactive robot training for non-markov tasks. *CoRR*, abs/2003.02232, 2020.

[62] Yonadav Shavit, Nadia Figueroa, Sina Mirrazavi, and Aude Billard. Learning augmented joint-space task-oriented dynamical systems: A linear parameter varying and synergetic control approach. *IEEE Robotics and Automation Letters*, PP:1–1, 05 2018.

[63] Aaquib Tabrez, Jack Kawell, and Bradley Hayes. Asking the right questions: Facilitating semantic constraint specification for robot skill learning and repair. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6217–6224, 2021.

[64] Ana Lucia Pais Ureche, Keisuke Umezawa, Yoshihiko Nakamura, and Aude Billard. Task parameterization using continuous constraints extracted from human demonstrations. *IEEE Transactions on Robotics*, 31(6):1458–1471, 2015.

[65] Nils Wilde, Dana Kulić, and Stephen L. Smith. Learning user preferences in robot motion planning through interaction. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 619–626, 2018.

[66] Jinqiang Yuan, Chee-Meng Chew, and Velusamy Subramaniam. Learning geometric constraints of actions from demonstrations for manipulation task planning. In *2018 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pages 636–641, 2018.

[67] Weichao Zhou and Wenchao Li. Safety-aware apprenticeship learning. In Hana Chockler and Georg Weissenbacher, editors, *Computer Aided Verification*, pages 662–680, Cham, 2018. Springer International Publishing.