An Evaluation of GUI and Kinesthetic Teaching Methods for Constrained-Keyframe Skills

A Thesis
Presented to
The Academic Faculty

by

Andrey Kurenkov

In Partial Fulfillment
of the Requirements for the Degree
Bachelor of Science

School of Computer Science
Georgia Institute of Technology
May 2015

An Evaluation of GUI and Kinesthetic Teaching Methods for Constrained-Keyframe Skills

SUMMARY

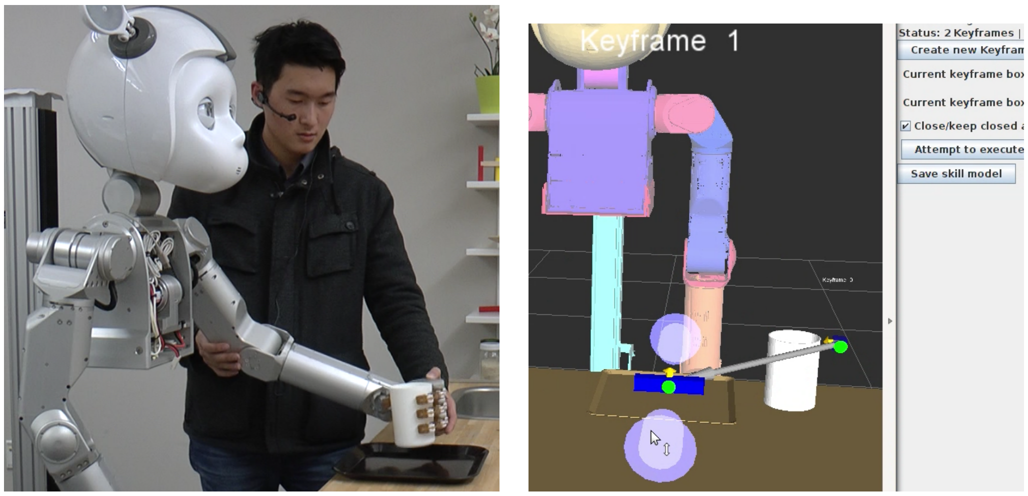
Keyframe-based Learning from Demonstration has been shown to be an effective method for allowing end-users to teach robots skills. I propose a method for using multiple keyframe demonstrations to learn skills as sequences of positional constraints (c-keyframes) which can be planned between for skill execution. I also introduce an interactive GUI which can be used for displaying the learned c-keyframes to the teacher, for altering aspects of the skill after it has been taught, or for specifying a skill directly without providing kinesthetic demonstrations. I compare 3 methods of teaching c-keyframe skills: kinesthetic teaching, GUI teaching, and kinesthetic teaching followed by GUI editing of the learned skill (K-GUI teaching). Based on user evaluation, the K-GUI method of teaching is found to be the most preferred, and the GUI to be the least preferred. Kinesthetic teaching is also shown to result in more robust constraints than GUI teaching, and several use cases of K-GUI teaching are discussed to show how the GUI can be used to improve the results of kinesthetic teaching. The results indicate the benefit of K-GUI teaching over kinesthetic teaching, and suggest that more robust constraints may be needed to more flexibily represent skills and allow for a more intuitive GUI.

Introduction

The goal of Learning from Demonstration (LfD) research is to to enable people with no specialized robotics knowledge to teach robots new skills [1]. Complex humanoid robots are capable of a broad range of skills that could assist humans in both industrial and domestic settings, yet programming these skills requires specialized knowledge and is time consuming. LfD strives to enable robots to learn skills from human demonstrations, and for the learned skills to be usable in environments and contexts not shown during teaching.

A common means of providing demonstrations is kinesthetic teaching, in which the teacher physically guides the robot through a skill. These physical demonstrations can be used to show the skill as a full trajectory, a series of positions (known as keyframes) from the full motion, or a hybrid of both [2]. Several such demonstrations can be given, so that the robot can learn a better model of the skill. Multiple demonstrations can result in a better skill model, since they lessen the impact of bad demonstrations and enable the learning of sets of constraints that describe the skill as generally as possible [3]. Keyframe demonstrations have been shown to be more comfortable for people when giving multiple demonstrations [2]. Alexandrova et al. have also shown that keyframes from a single demonstration can be visualized in an interactive GUI to allow users to directly edit the keyframes of the model [4].

Though splining between keyframes can be used for executing the skill, this approach is not robust to obstructions in the environment not seen during teaching. Though Alexandrova et al. have shown that a visual representation of keyframes can clearly communicate the robot's model of a taught skill, and so allow the teacher to make any needed corrections after kinesthetic teaching, their representation only uses the end effector poses from a single keyframe demonstration and so suffers from this lack of robustness to obstacles [4]. In this work I propose the constrained-keyframes (c-keyframes) skill representation, which can be executed by using motion planning and so allow the skill to be executed despite obstructions in the environment. I describe how c-keyframe skill models can be learned from multiple keyframe demonstrations and how they can visualized in an interactive GUI for editing. I



(a) Kinesthetic Teaching (b) GUI Teaching

Figure 1: Example of teaching to place a cup on a platter in the two teaching modes with our robot platform, Curi.

also compare three teaching methods for c-keyframe skills: kinesthetic teaching, GUI teaching, K-GUI teaching. Comparing the different teaching approaches is useful for evaluating which approach is best suited for LfD, and investigating whether a GUI without kinesthetic interaction is sufficient for teaching skills. I report the results of a study which show that users both prefer and are in some cases more effective at specifying skill constraints using K-GUI teaching, and that though they are able to use the GUI for teaching skills they prefer and are better at using the more intuitive kinesthetic teaching approach.

LfD Survey

The goal of Learning from Demonstration (LfD) research is to to enable people with no specialized robotics knowledge to teach robots new skills [1]. Complex humanoid robots such as Curi **??** are capable of a broad range of skills that could assist humans in both industrial and domestic settings, yet programming these skills requires specialized knowledge and is time consuming even for roboticists. LfD simplifies the task of adding new skills to robots by having them learn these tasks from demonstrations given by humans, which can be done with no programming or robotics knowledge. Furthermore, LfD strives to require as few demonstrations as possible by creating generalized models of tasks that can be used in environments and contexts not seen during learning.

The definition of LfD is purposefully ambiguous with regards to how the task is demonstrated and how it is represented, since multiple approaches to LfD with different methods of demonstration and action representation have evolved since its inception in the 1980s. It is also important to note that LfD strives to have robots learn a robust model of the task that can be adapted to contexts not seen in any teaching demonstrations. Furthermore, LfD techniques generally seek to learn good models of the task even if some demonstrations are flawed. Therefore, it is not sufficient to have the robot learn a single trajectory or set of trajectories that is then executed in exactly the same way thereafter, as is common in industrial robotics [5].

LfD was originally inspired by human learning by imitation, and was in fact called that in a 1996 survey of the topic by Bakker and Kunyoshi [6]. The motivation for learning by imitation is to have a third approach to teaching a robot to perform a certain skill and have this approach be easier than direct programming and more reliable than independent robot learning. Learning by imitation is defined to have three necessary steps: *observing, representing, and reproducing* an action. The ability to adapt to new situations was also cited as being important. The first approach offered by Kuniyoshi and Inaba involves the robot observing a teacher completing an assembly task and creating a hierarchical symbolic representation of the task using pre-defined primitive skills. Observation of the demonstration is done with a computer vision method as well as a special worn glove, which allows the robot to extract the state at any point during the demonstration and learn the sequence of primitive skills that match the more complex demonstrated skill. The approach is validated using a table constructions task, in which the starting locations of different components are varied but the learned skill is robust enough to be executed successfully [7]. The other two early methods for learning by imitation covered in the 1996 survey involve robots directly imitating another agent by observing its actions and repeating them. These approaches are considerably more limited than the planning based approach of Kuniyoshi and Inaba, since they are based on repetition of observed tasks that works with a simple domain such as a maze but would fail in a more complex domain such as assembly where the initial parts may begin in different positions [6].

Atkenson and Schaal used the term Learning from Demonstration for their work, though the motivation behind it is the same as that of learning by imitation. Their work demonstrates an approach to LfD that does not involve learning a model of the task to be able to execute it later, and that instead uses the demonstrations to create a reward function to guide the robot in a policy search [8]. The concepts of policy search based on a reward function comes from the Machine Learning field of Reinforcement Learning, which represents a task as a mapping between states and actions. The appropriate mapping can be found if a reward function exists for transitions between states, since an initial policy can be chosen and iteratively improved until expected reward is maximized. Atkenson and Schall use demonstrations both to create the reward function, by rewarding the robot being in states found in the demonstration, and to use them for creating the initial policy. Unlike the previous approaches, this one is meant for learning continuous manipulation skills such as swinging a pendulum. Their experiments demonstrated several important findings: that merely mimicking the demonstrated human actions is not sufficient for good execution due to modeling error, that reinforcement-learning approaches perform better than direct mimicking but may still not converge if the task is modeled incorrectly, and that incorporating task level learning (which uses parameters from the task's object instead of just the robot state) assisted with learning better than reinforcement learning alone. Many variations on using Reinforcement Learning for Learning from Demonstration exist, with a notable one being the approach of inverse reinforcement learning which derives a reward function using demonstrations and features from the skill [9].

Dillmann et al. suggest a classification approach for LfD (referred to as PbD, for Programming by Demonstration) systems that covers the distinctions betweed LfD approaches discussed above: whether the skill being taught is a high-level sequence of actions or a basic trajectory, the form of demonstrations given, how the skill is internally represented by the robot, and whether the skill execution is based on direct execution from the skill representation or an additional planning step is used the action can be executed[10]. They also criticize existing approaches up to that point for not generalizing, not allowing observation of skills on a non-abstract level, and not having a validation and feedback means. To address this, they propose a complex LfD system with a Machine-Learning based observation system for human actions and support for teacher feedback to improve the learned action. Nicolescu and Mataric likewise criticize existing approaches to LfD for ignoring feedback cues and generalization. Similarly to [7], their proposed solution is based on learning high-level complex actions built from an existing set of primitive skills, linking observed effects at each stage of the skill to effect of primitive actions[11]. Unlike prior approaches, the robot does not just observe the teacher but is rather led through an execution of the skill by the teacher during the demonstration. These two forms of demonstration were later categorized as imitation teaching, in which the a robot just observes a demonstration of the skill through either external observation or sensors on the teacher, and demonstration teaching, in which the robot directly senses the demonstrations through either teleoperation or shadowing. Knowledge about the primitive skill set and feedback cues are used to cut down the set of observations, and a standard longest common subsequence algorithm is used to find a generalized model of the skill from multiple demonstrations [5]. Finally, like [10] this approach follows the initial learning of a generalized model of the skill with the teacher observing the robot performing the learned skill and possibly providing feedback cues that further improve the learned skill model. The benefit of teacher critique of the learned skill has also been argued for and experimentally supported in more recent work [12][13].

The above works demonstrate the broad categories into which LfD systems fit, though there are many different learning approaches and skill representations that fit into these categories in different ways. Examples of more novel LfD systems include one inspired by neural mechanisms [14] and one that uses a more sophisticated representation of skills through gaussian mixture models [15]. A topic of research gaining increasing attention is using these LfD techniques together with systems that allow for the creation of demonstrations in a virtual environment. Demonstration in virtual environments provides several benefits: they vastly simplify tracking of the teachers' actions, all sensor noise is eliminated, it can be done without physically being near the robot, and the ability to simulate the task in the same virtual environment in which it was taught [16]. However, virtual environments present the drawbacks of requiring modeling of the skill's environment and limiting the ability of the user to potentially less intuitive input modalities. Despite these drawbacks, demonstrations may still be easier to give in virtual environments than in the real world with the use of "virtual fixtures," which are additional visual or other feedback signals provided to the teacher to as guidance or clarification of what the robot is learning from the given demonstration. An additional benefit is that the ability to give demonstrations from a distance means demonstrations can potentially be supplied by many more teachers over the internet, and the quantity of such demonstrations will overcome any decreased quality in the demonstrations [17].

Related Work

There has been extensive research done in LfD on learning skill policies as well as approaches for recreating smooth trajectories from either one or multiple demonstrations. A topic that has been researched less extensively is the use of LfD for learning skills as sets of constraints that can be used with classical and motion planning. Constraint extraction has been done for finding appropriate reference frames as well as relevant objects for sequences of skill primitives that are learned from segmenting multiple demonstrations [3]. Demonstrations have also been used to guide motion planning by speeding up constrained planning based on experience graphs [18] and by learning time-dependent task constraints from demonstrations which can be used by a sampling-based planner to match the demonstrations while avoiding novel obstacles [19]. Demonstrations have also been used to learn appropriate constraints for task-level skill models [20], and have been used for learning new task-level concepts that could be used as goal constraints [21].

LfD research also encompasses the question of how users can best provide demonstrations of skills to robots. In [22], user satisfaction with a dialog-based interface for providing demonstrations is evaluated. In [2], a similar interface is used and keyframe demonstrations are proposed and evaluated as an alternative to trajectory demonstrations. Users did not prefer one method significantly over the other, except that keyframes were preferred to trajectories for teaching with multiple demonstrations. An interactive GUI for editing the positions and reference frames of keyframes recorded from a single demonstration is suggested and shown to be helpful for users in [4]. This GUI is also evaluated in the context of fixing the aspects of demonstrations through a "crowd" of users using a cloud-based web interface[23]. A cloud framework has also been suggested for full teleoperation and ability to give demonstrations remotely, which could simplify gathering many demonstrations [?].

There is a need for LfD that enables the execution of actions through motion planning, since that allows skills to be executed independently of how they were demonstrated and so naturally be adapted to new environments and objects. Though several approaches have been proposed for using trajectory demonstrations to extract time-dependent information to guide motion planning, no work has yet proposed an approach for learning a discrete sequence of constraints to plan between for skills that can be taught with keyframe demonstrations. A discrete set of constraints has the benefit of being possible to fine tune without further demonstrations, and for possibly being more adaptable to new objects and environments. In this work I propose the c-keyframes skill representation, which is composed of sequential constraints on end effector positions and can be learned from multiple kinesthetic keyframe demonstrations. Additionally, I show how skills can both be edited and directly specified in an interactive GUI similar to that of [4], but that allows for editing skill constraints rather than end effector poses. The GUI is evaluated for its usefulness in teaching robots without requiring to be physically near them, and as a means for improving upon kinesthetic teaching.

Skill Representation

The c-keyframe skill representation follows from the nature of keyframe demonstrations. Keyframes for end effectors only encode a single pose, whereas c-keyframes have a space of possible poses for the end effector defined with a box-shaped positional constraint and a single associated orientation. The benefit of using positional constraints is that they can be easily learned from multiple kinesthetic demonstrations, can be intuitively visualized in an interactive GUI, and can be directly used with the OMPL motion planning library for skill execution [24]. Because they are made up of a box-shaped position constraint and a single

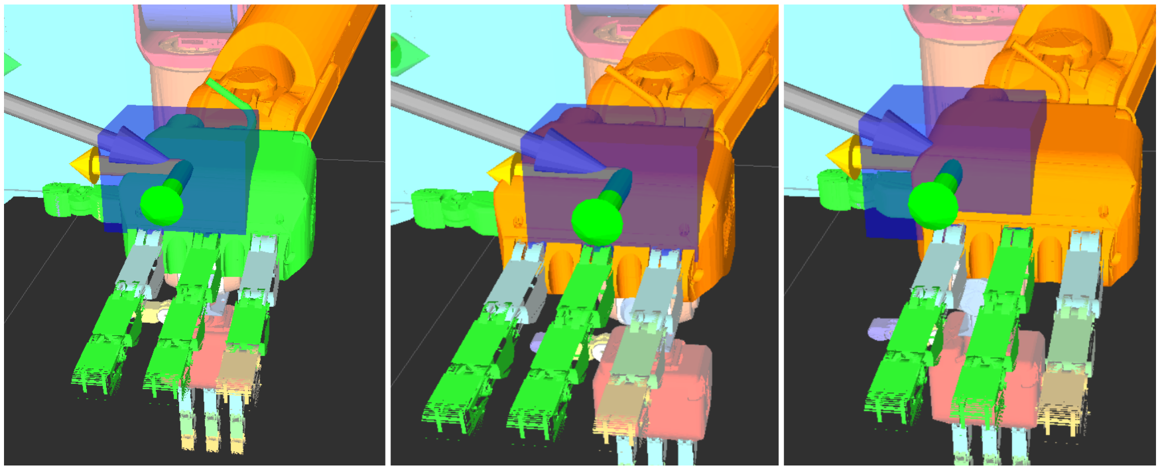


Figure 2: Visualization of a single c-keyframe and 3 possible end effector poses defined by it. A pair of arrows defines the end effector orientation.

orientation, c-keyframes can be easily visualized by a combination of a box and two arrows as shown in Figure 2. ROS markers and rViz were used to create this visualization [25].

Skill Teaching Methods

Kinesthetic

A set of keyframe demonstrations, as in [2], for the same skill can be used to find its c-keyframe representation. To do so, the keyframes from all the demonstrations are first clustered. In this work I used a tuned version of k-means clustering with k being set to the rounded average of the number of keyframes from all the demonstrations, though a more robust approach based on Gaussian Mixture Models could also be used [2]. The k centroids are initialized to the keyframes closest to the average of the set of keyframes with the same sequence number from demonstrations with at least k keyframes. Next, a box constraint from each cluster can be found by extracting the minimum volume box that encloses all the keyframes in that cluster and has the center of those keyframes. This is done by performing Principal Component Analysis on the the keyframes, and using the resulting principal components to define the orientation of the box. The scale of the box along each principal component is set to be large enough to reach the farthest keyframe along that component. An orientation can also be found from each cluster by averaging the orientation of all the keyframes in it. The combination of the box position constraint

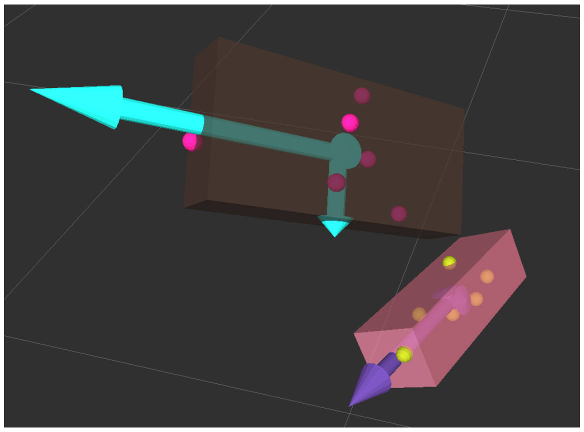


Figure 3: Image showing a set of clustered keyframes. The arrows are the PCA components, and the minimum volume enclosing box is shown.

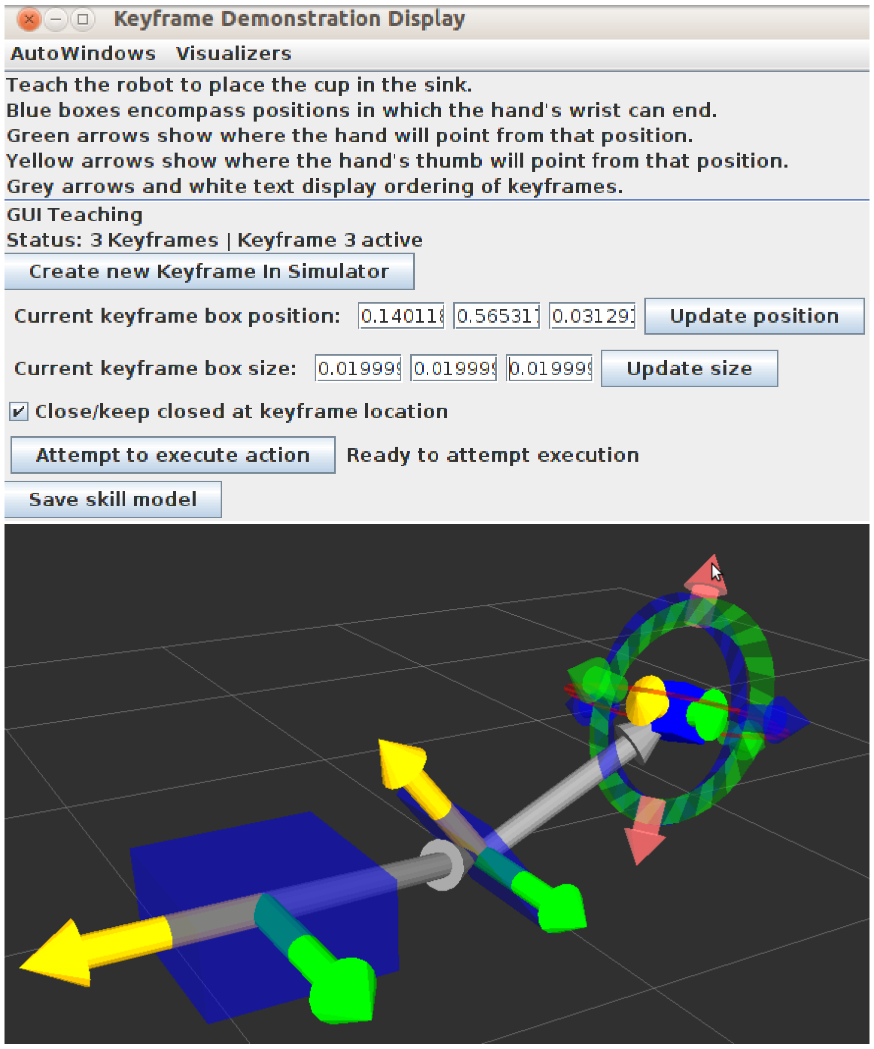
and orientation form a c-keyframe from each cluster, and the order of the c-keyframes is set based on the average keyframe sequence number of the keyframes within each cluster.

GUI

Constrained keyframes can also be directly specified through a GUI, avoiding the need for any demonstrations. As shown in 4, the GUI is implemented using a combination of a Java-based GUI for text input and buttons as well as interactive markers in rViz which display the keyframes and allow them to be moved and rotated. The c-keyframes can also be selected for editing by directly clicking on the box of the keyframe in rViz. The buttons on the Java GUI include the functions for keyframe creation, precise positioning and sizing, setting the hand to close or open, and attempting to execute the skill specified by the current set of c-keyframes. Motion planning is done using the OMPL motion planning framework, which supports setting goals based on positional constraints. Reference frames are not yet specified, since the focus of this study is on specifying constraints, though that aspect could be included in the GUI as in [4].

K-GUI

Based on the two previous methods for creating and editing c-keyframes skills, it is straightforward to first teach a model of a skill with kinesthetic demonstrations and then edit it in the GUI. Unlike in the GUI-only approach, the GUI in this teaching method is used


Keyframe Demonstration Display
AutoWindows
Visualizers

Teach the robot to place the cup in the sink.
Blue boxes encompass positions in which the hand's wrist can end.
Green arrows show where the hand will point from that position.
Yellow arrows show where the hand's thumb will point from that position.
Grey arrows and white text display ordering of keyframes.

GUI Teaching
Status: 3 Keyframes | Keyframe 3 active

Create new Keyframe In Simulator

Current keyframe box position: 0.140118 0.565311 0.031291

Current keyframe box size: 0.019999 0.019999 0.019999

[✓] Close/keep closed at keyframe location

Ready to attempt execution

Figure 4: Interactive GUI for directly specifying constraints. A java based GUI allows the user to create new keyframes and their position and size. It also allows to specify whether the robot's hand should be closed and to preview the skill with simulated execution. Interactive markers in rViz are used to be able to edit the position and rotation of keyframes.

to correct any flawed aspects of the skill learned from demonstrations. The GUI can also be used to expand c-keyframes to cover as much area as possible, and therefore teach the skill as robustly as possible. In every K-GUI teaching scenario, the objects involved in the kinesthetic teaching tasks should exist in the rViz environment as well.

User Study

I conducted a user study with 10 participants, who were undergraduate and graduate students with no experience in robotics at the Georgia Institute of Technology. The purpose of the study was to evaluate which methods of teaching novice teachers prefer and are good at for teaching c-keyframe skills.

Study Protocol

Each study participant was tasked with teaching a single skill using each of the teaching methods. Three skills with appropriate for the c-keyframe representation were chosen: placing a cup anywhere on a platter, pouring liquid from a cup into a bowl, and closing the lid of a box. Before teaching with each method, the participants were guided through a practice task of placing a cup on the edge of the table. It was explicitly explained that in kinesthetic mode they should give a range of demonstrations to show different places the cup can be put down along the edge of the table, and that in the GUI mode the keyframe for placing down the cup should be made large enough to cover the entire edge of the table. The order of teaching methods was counterbalanced, and the instructions given for each skill during practice were kept the same regardless of the order. The order of which skills were taught was kept the same for all participants, since the intention was to compare the teaching methods and it was not expected that the order of skills would affect that. A time limit of 10 minutes was placed on each teaching method, but otherwise it was left up to the participant to decide how many keyframes or demonstrations were appropriate while teaching.

Metrics

Metrics were collected for evaluating the speed, difficulty, user preference, and constraint robustness for each teaching method. To evaluate teaching speed, users were timed during each teaching interval. Perceived difficulty and preference were evaluated using a survey filled out by participants afterwards. People were also asked to respond to the free-form question "Based on your experience, comment on the pros and cons of each mode of teaching for teaching skill constraints." The robustness of the learned skills was evaluated based on how many different virtual environments with different collision objects the resulting skill could be planned with succesfully.

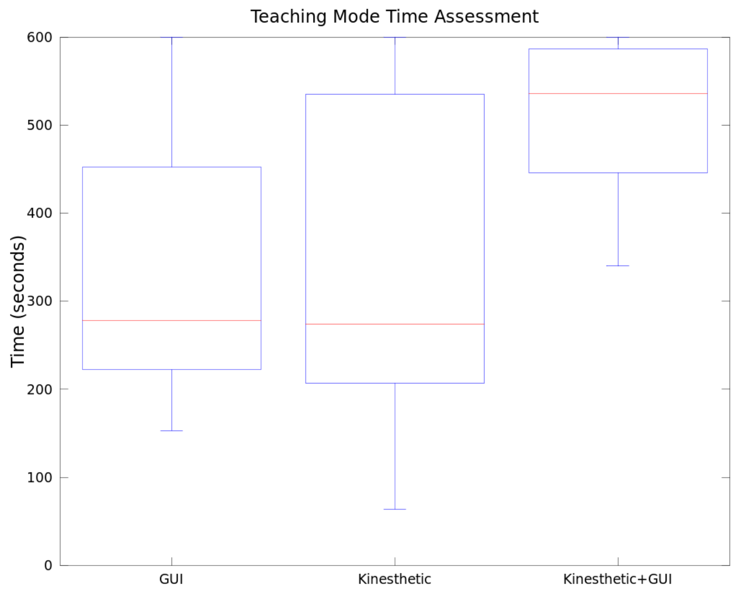


Figure 5: Measured speed of each teaching mode.

Results

Speed

The measured times are shown in Figure 5. Participants were allowed to at most use 10 minutes of teaching time for each skill, and were told to take about a minute to finish teaching when they reached 9 minutes of teaching. Counterbalancing of the order of the teaching methods, and so which which skills were taught with each method, was done in order to account for variable difficulty of teaching skills. A clear and predictable result is that K-GUI teaching takes longer than kinesthetic teaching by itself. On average, the GUI teaching took about the same amount of time to use as kinesthetic teaching. Kinesthetic teaching times varied due to the users choosing to provide different numbers of demonstrations, and GUI teaching times varied due to users choosing to spend different amount of time fine tuning the taught skill.

Difficulty

The collected difficulty evaluations are presented in Figure 6. On average the GUI mode of teaching was evaluated as being the most difficult, and kinesthetic the easiest. Though

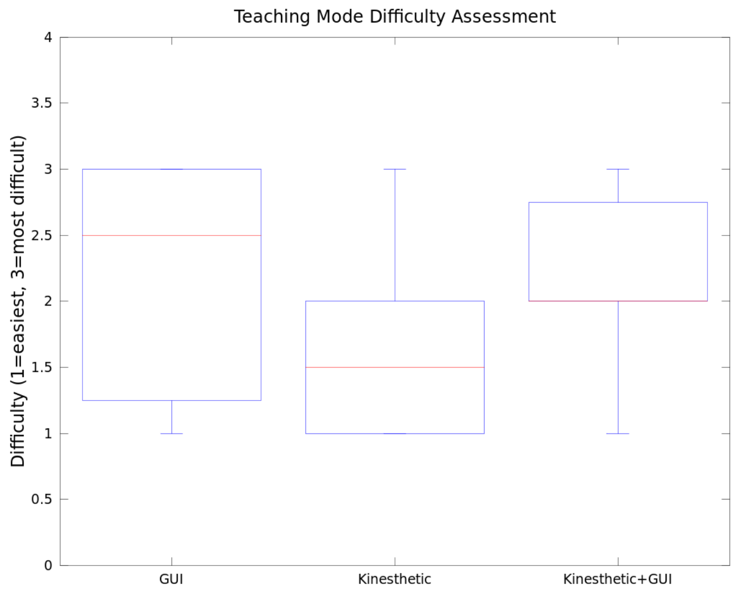


Figure 6: Difficulty evaluations from survey.

this matches with our expectation, due to the low sample size the result is not statistically significance. However, several participants comments also indicates that kinesthetic teaching was easiest. For example: "Kinesthetic seems more intuitive","kinesthetic helps to teach to me as the user exactly what is required for success (this is hard to simulate on the computer)" "Easy to learn,Intuitive." The comments concerning the GUI reflected greater difficulty. For example: "Harder to use software", "little bit hard using GUI to adjust the position or camera", "Just GUI gives us more freedom, but it might not be that intuitive.", "For just GUI, it could be accurate but not realistic or hard to teach compared to kinesthetic with GUI." Users appeared to have most difficulty in the GUI with moving the camera and the keyframes in 3D space.

Preference

Preference was measured through the survey with the question "If you had to teach another task to the robot, which mode of teaching would you choose." The answers to this question are as follows: 0 chose GUI, 3 chose kinesthetic, and 7 chose K-GUI. Thus, K-GUI is chosen significantly more often as most preferred (χ^2, $p < 0.01$ compared to random chance). The GUI mode of teaching was the least preferred mode in accordance with it being the most

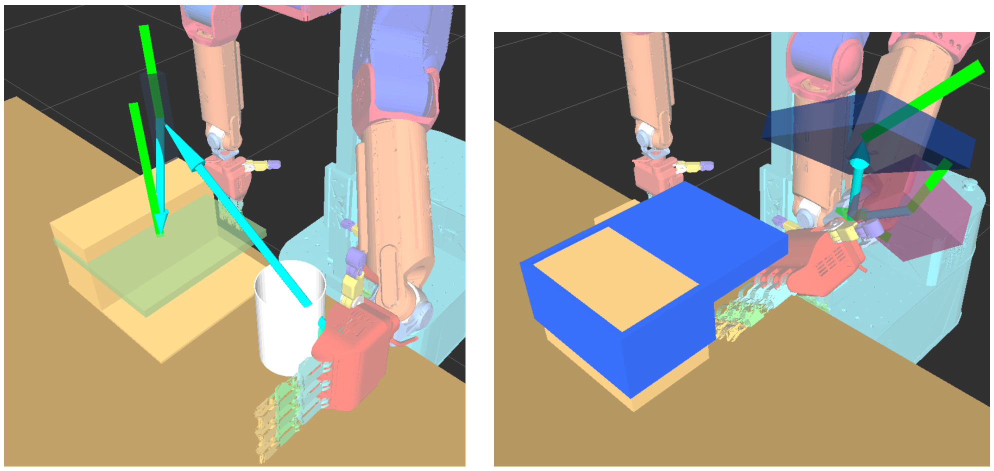

(a) GUI platter model (b) Kin+GUI Box Model

Figure 7: Examples of model evaluation.

difficult, though users commented that the GUI had the benefit of allowing them to be more accurate than kinesthetic teaching. Despite the K-GUI mode being on average slower and more difficult to use than kinesthetic teaching, it was selected as the most preferred approach. The preference for the K-GUI model was explained in several participant comments: "I prefer kinesthetic [combined with GUI] because it is easy to teach at first and then one could amend actions that seems problematic through GUI", "working in the GUI allows to tweak motions from kinesthetic..." These results are in line with what has been shown in [4], where users considered a GUI for editing the keyframes of a single keyframe demonstration to be useful for visualizing exactly what has been learned and being able to edit it. Here I are showing the utility of skill visualization and editing for our more general skill representation.

Constraint Robustness

Figure 7 shows the process by which the skill models that were stored from the user study were evaluated for constraint robustness. This was done by attempting to plan with each model in a total of 15 simulated test environments with different collision objects added to the scene. Grasping of the cup as well as the physics of the box's lid were not simulated, so the goal of planning was only to move the end effector based on the keyframe constraints. It is expected that the success rate of taught skills at correctly executing the action will

Participant	Mode Order	GUI	Kinesthetic	K-GUI
Participant 1	G-KG-K	7/15	13/15	12/15
Participant 2	G-KG-K	9/15	13/15	9/15
Participant 3	K-G-KG	5/15	7/15	0/15
Participant 4	KG-K-G	9/15	6/15	14/15
Participant 5	G-kG-K	7/15	4/15	13/15
Participant 6	KG-G-K	6/15	11/15	14/15
Participant 7	K-KG-G	5/15	10/15	6/15
Participant 8	G-K-KG	11/15	7/15	4/15
Participant 9	K-G-KG	0/15	10/15	5/15
Participant 10	KG-G-K	0/15	11/15	12/15
Average	NA	5.4	9	8.9
Standard Deviation	NA	3.3	3.7	5.1
Total	NA	59/150	92/150	89/150

Table 1: Constraint robustness results from simulated tests of skills in multiple environments. Skills were always taught in the order platter-pour-box.

correlate with this measure. The result of this evaluation process are shown in Table 1.

The GUI method of teaching has the lowest average number of successful planning attempts and the lowest standard deviation for this result of the three, despite allowing for direct sizing of the keyframes. This may be because participants often did not elect to resize the keyframes, which resulted in planning not being possible when obstructed when collision objects are present. As can be seen on Table 2, the GUI results are least successful for the pour task; this may be because resizing the constraints was not as straighforward as the platter task.

Kinesthetic teaching has a slightly higher average numbers of successful planning results compared to K-GUI. Additionally, on average the skill models from K-GUI teaching prior to GUI edits have more planning successes than after editing has been done. However, K-GUI teaching has higher standard deviation in both measures due to several particularly bad skill models from participants 3, 8, and 9 for the box skill. From Table 2, it can be observed

Skill Type	GUI	Kinesthetic	K-GUI
Platter	56.6%	60.9%	88.8%
Pour	18.3%	37.7%	60.0%
Close Box Lid	73.3%	80.0%	36.6%

Table 2: Constraint robustness results by skill type.

that these bad skill models result in K-GUI having a much lower success percentage for the box skill despite it being higher for the other two skills. Due to the small sample size of the study and natural variation in user performance it cannot conclude whether K-GUI has no advantage over kinesthetic teaching or if the bad skill models are outliers and K-GUI otherwise presents a benefit over kinesthetic teaching. However, example use cases from the study support the idea that the GUI is either not used to alter the kinesthetic model or is used to improve it.

Qualitative Results

Several use cases can be used to explain why the GUI had the lowest success rate for constrained planning. Figure 7a presents an example of the platter skill in which users made the placement keyframes as large as possible by using the GUI's resizing feature to correctly specify that the cup can be placed anywhere on the platter. However, participants did not consistently use the GUI's resizing capability as in that example despite being explicitly guided to resize a keyframe for the practice task. Figure 8 illustrates several models where the users either did not resize the keyframes as much as possible or did not resize them at all from the default size. A possible reason for this is that although it was stated that the users should attempt to teach the robot how to do the skill as generally as possible, some users focused more on adjusting the keyframes so planning in simulation

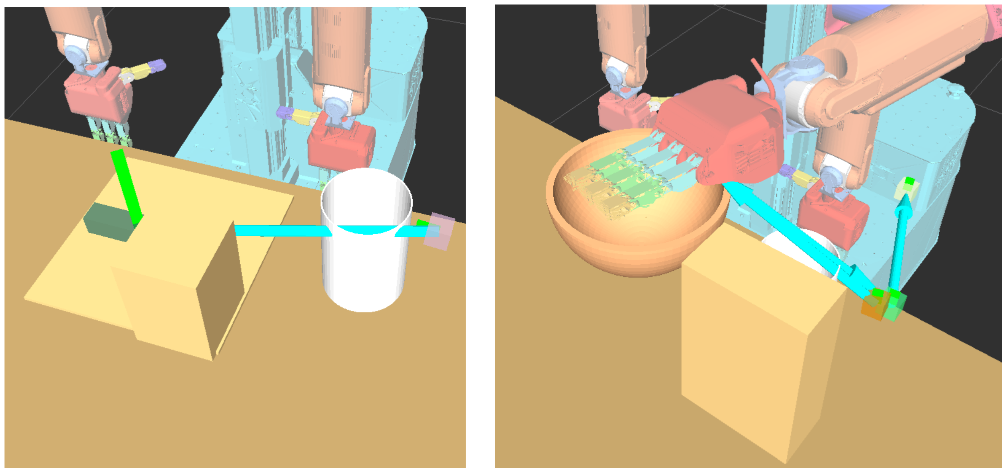
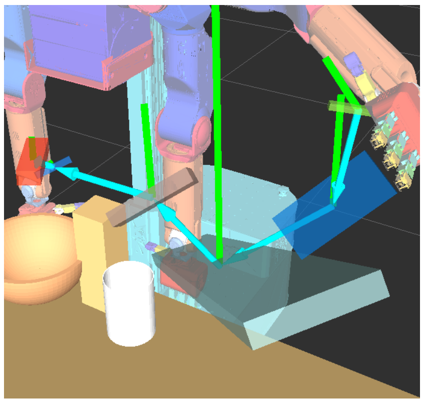
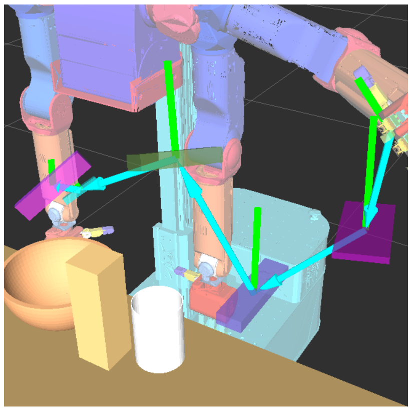
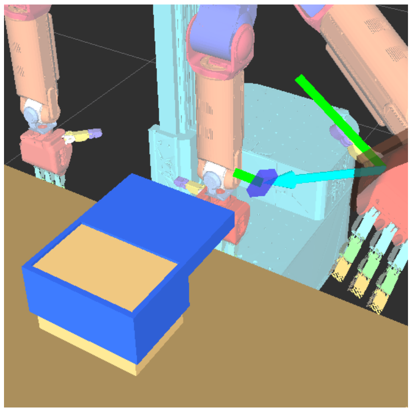

(a) Platter skill (b) Pour skill

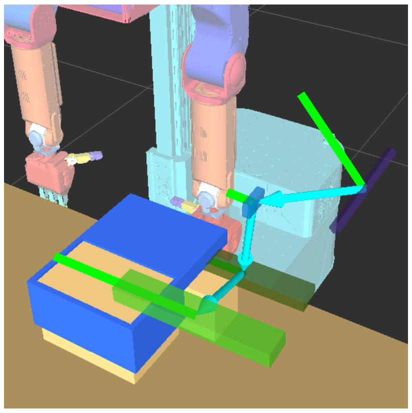
Figure 8: Examples of limited GUI models.

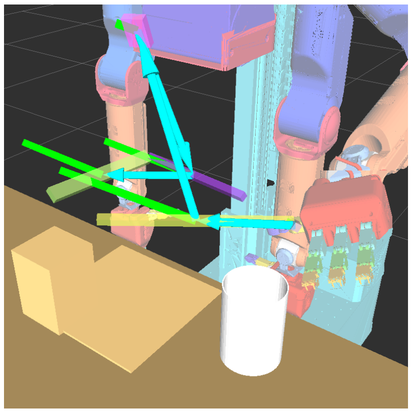
would successfully execute. This suggests that in the future it would be beneficial to use simulated test environments like the ones used in our evaluation during the training phase, to encourage teachers to think about their model's generality.


(a) Before - the cup grasping keyframe is too large.


(b) After - the user has edited that keyframe.


(c) Before - too few keyframes were saved.


(d) After - the user added missing keyframes.


(e) Before - two thin keyframes over the platter.

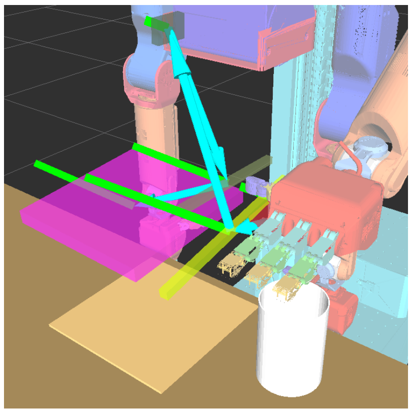

(f) After - the user enlarged a keyframe over the platter.

Figure 9: K-GUI models before and after GUI edits.

K-GUI teaching similarly resulted in participants not consistently resizing the keyframes to reflect their full allowed size. Participants either did not do any significant editing to the kinesthetic models or fixed perceived problems with them. Those who did not edit the skill usually executed the skill in simulation in order to see what the robot learned, but elected not to make more changes after that. However, some participants did use the GUI to improve kinesthetic models as designed, seen in Figure 9. These cases illustrate the capacity to use the GUI to fix problems in kinesthetic skill models, which the quantitative planning success metric does not capture. Additionally, participants' preference of K-GUI teaching even despite ranking it harder and it taking longer suggests is a strong indication that having such a hybrid approach worth exploring.

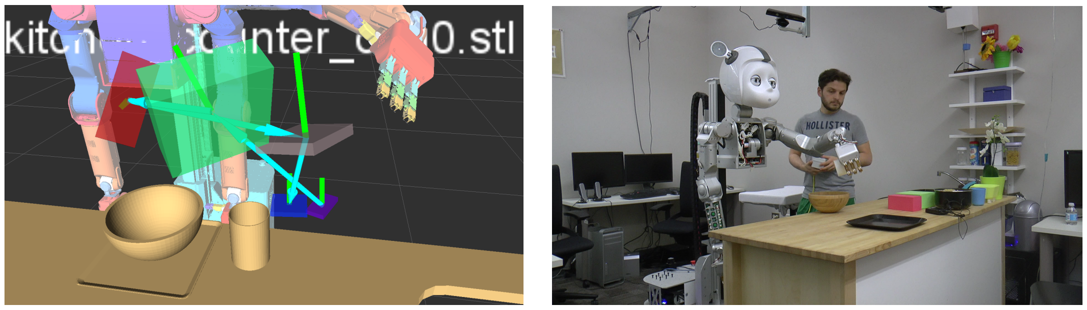
Post Study Evaluation

After the study concluded, additional work was done to test the ability of the saved skill models to be executed on Curi. A new GUI was written to enable the loading of saved models and demonstrations, and the execution of those models in both rViz and on the actual robot. The execution code was implemented by converting the ROS motion plans into a trajectory demonstration format, and using pre-existing code for making Curi execute this trajectory. In order to plan valid trajectories, the positioning of the virtual table object was adjusted to match that of the physical table in the lab and the objects involved in the skill were positioned according to where they were during the study.

When attempting to execute several skill models, many attempted executions that looked like they should work in rViz did not work on the actual robot. In several cases offsets that were added to the height for planning resulted in the hand going above the cup and failing to grasp it, and in other cases the hand moved too close to the table or even hit it. Several aspects of the implementation could be improved in order to make execution on Curi more reliable:

- Perception with the Kinect should be used during demonstration recording to save the exact locations of the end effector relative to the cup

- The virtual model should be tuned to match the real positioning of objects in the lab,



(a) The skill model to execute. (b) A moment from the execution.

Figure 10: Example of executing a taught skill.

and the planning should be modified to account for Curi grasping the cup

- Curi's controller parameters should further be tuned

After encountering multiple issues with executing the saved skill models, a qualitative examination of all the saved skill models was done to record how many models should be usable on the physical robot. The results, recorded in more detail in Appendix 1, indicate that 9/10 kinesthetic skills should be usable, 6/10 GUI skills should be usable, and that 7/10 K-GUI skills should be usable. Since most of the learned skill models are fit for execution, this approach to learning and using constraint-based skill models should work with the robot if the problems enumerated above are addressed. However, many of the executable skill models also have some possibility or producing motion plans that fail, and so additional work can also be done on learning better constraints or optimizing already learned constraints.

Future Work

This work is an initial step in researching methods for novice teachers to teach robots robust constraint-based skill models. Therefore, it can be extended in multiple ways. The primary ways this work should be extended in the future are:

- **More robust end-to-end implementation**

  The primary focus when completing this thesis was to implement the software needed to perform the user study and collect the quantitative and qualitative results from that. Though this resulted in the study's completion, several aspects of the implementation should be revisited if this approach to learning is explored further. The most important aspect is using perception while teachers provide demonstrations. This can allow for recording object locations more precisely as well as potentially additional features for learning constraints. Additionally, the implementation should be extended to use information about grasping changes from demonstrations to solve for the correct grasp state for each c-keyframe, rather than specifying when prior to planning as is done now. Beyond the implementation of learning constraints from demonstrations, the implementation of motion planning needs to be made more robust. Currently, the grasped cup object is not treated as a collision object, although it should be considered a tool attached to the end effector. Though this was allowed for as non-essential for the study, this should be altered to produce consistently valid plans. Additionally, the implementation is limited due to not planning with reference frames other than the robot's body frame. The GUI can easily be modified to allow for modification of c-keyframes reference frames, and the lab's prior work on learning appropriate reference frames is highly relevant to this subject. Lastly, more accurate simulation based on physics and object manipulation can be explored for improving user experience with the GUI.

- **Using the GUI for Active Learning**

  It is somewhat straighforward to explore using the GUI as a means to active learning, by making the model being generated from demonstrations visible during teaching. Beyond direct visualization of the current state of the model, the lab's previous work on verbal and gesture-based active learning can be extended to having "visual questions" within the GUI that express uncertainty about whether the constraints should be more limited or general. One of the main benefits of using a GUI is that all it allows for efficient communication of information about 3D space, which can provide the same questions much faster than other modes of communication. It is promising to explore whether Active Learning would result in better constraints.

- **Skill representation and learning of more robust constraints**

  One limitation of our approach is that each c-keyframe must have a single orientation, which is not appropriate for many skills. To allow for variation in orientation as well as position, each c-keyframe could be extended to have more than one oriented position constraint. This could be learned by performing an additional clustering step for the keyframes belonging to each c-keyframe. A more complex and powerful possibility would be to use a set of semantic constraints such as "above" or "next to" and automatically learn the appropriate set of such constraints from kinesthetic demonstrations. Users could then both specify and edit the learned models by using such semantic geometric constraints, rather than the literal geometric boxes as in our current approach. This would make it much easier to use the GUI by just specifying constraints such as "grasp" or "move above" and modifying several numeric variables rather than using the 3D camera and interactive markers for full 3D positioning. It would be significantly more difficult to learn and visualize the appropriate constraints if such high-level semantic constraints are allowes, but the additional effort and complexity seems to be justified by the current GUI's low ratings in terms of difficulty and preference. There are multiple integrated symbolic and motion planning solutions that can be considered for this research direction [26][27][28].

- **Remote teaching, adaptation and synthesis of many skill models**

  One of the original motivations of this work was the promise of using a GUI in the context of remote teaching. Using the implemented GUI, either keyframe-based demonstrations for learning or entire constrained skill models could be provided remotely. The Robot Management System provides a framework for remote teaching which could be integrated with the current approach to explore remote teaching of constraint-based skills [17]. The promise of being able to collect many demonstrations or skill models could enable research into learning more robust constraints, synthesizing multiple constraint-based skill models, or using a constrained skill model in different environments with GUI input

Conclusion

In this thesis, I have proposed a constraint-based skill representation that can be learned from multiple kinesthetic keyframe demonstrations. I have also showed how skills with this representation can be visualized and edited in an interactive GUI. The results of a user study comparing the speed, difficulty, user preference, and constraint robustness for different teaching modes were presented. The K-GUI teaching mode was found to be the most preferred, and though its constraint robustness is quantitatively similar to that of kinesthetic teaching I discuss use cases in which it allows for improving upon kinesthetic teaching. This work provides a basis for researching future approaches for learning of more robust constraints from demonstrations, as well as justification for the need of a GUI in addition to kinesthetic demonstrations and evidence that the simplest implementation of such a GUI is not sufficiently intuitive for naive teachers to use by itself.

REFERENCES

[1] S. Chernova and A. L. Thomaz, "Robot learning from human teachers," *Synthesis Lectures on Artificial Intelligence and Machine Learning*, vol. 8, no. 3, pp. 1–121, 2014.

[2] B. Akgun, M. Cakmak, K. Jiang, and A. Thomaz, "Keyframe-based learning from demonstration," *International Journal of Social Robotics*, vol. 4, no. 4, pp. 343–355, 2012. [Online]. Available: http://dx.doi.org/10.1007/s12369-012-0160-0

[3] L. Pais, K. Umezawa, Y. Nakamura, and A. Billard, "Learning robot skills through motion segmentation and constraints extraction," in *HRI Workshop on Collaborative Manipulation*, 2013.

[4] S. Alexandrova, M. Cakmak, K. Hsiao, and L. Takayama, "Robot programming by demonstration with interactive action visualizations," in *Proceedings of Robotics: Science and Systems*, Berkeley, USA, July 2014.

[5] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," *Robotics and Autonomous Systems*, vol. 57, no. 5, pp. 469 – 483, 2009. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0921889008001772

[6] P. Bakker and Y. Kuniyoshi, "Robot see, robot do : An overview of robot imitation," in *In AISB96 Workshop on Learning in Robots and Animals*, 1996, pp. 3–11.

[7] Y. Kuniyoshi, M. Inaba, and H. Inoue, "Learning by watching: extracting reusable task knowledge from visual observation of human performance," *Robotics and Automation, IEEE Transactions on*, vol. 10, no. 6, pp. 799–822, Dec 1994.

[8] C. G. Atkeson and S. Schaal, "Robot learning from demonstration," in *Proceedings of the Fourteenth International Conference on Machine Learning*, ser. ICML '97. San

Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1997, pp. 12–20. [Online]. Available: http://dl.acm.org/citation.cfm?id=645526.657285

[9] P. Abbeel and A. Y. Ng, “Apprenticeship learning via inverse reinforcement learning,” in *Proceedings of the Twenty-first International Conference on Machine Learning*, ser. ICML ’04. New York, NY, USA: ACM, 2004, pp. 1–. [Online]. Available: http://doi.acm.org/10.1145/1015330.1015430

[10] R. Dillmann, O. Rogalla, M. Ehrenmann, R. Zllner, and M. Bordegoni, “Learning robot behaviour and skills based on human demonstration and advice: The machine learning paradigm,” 1999.

[11] M. N. Nicolescu and M. J. Mataric, “Natural methods for robot task learning: Instructive demonstrations, generalization and practice,” in *Proceedings of the Second International Joint Conference on Autonomous Agents and Multiagent Systems*, ser. AAMAS ’03. New York, NY, USA: ACM, 2003, pp. 241–248. [Online]. Available: http://doi.acm.org/10.1145/860575.860614

[12] B. Argall, B. Browning, and M. Veloso, “Learning by demonstration with critique from a human teacher,” in *Human-Robot Interaction (HRI), 2007 2nd ACM/IEEE International Conference on*, March 2007, pp. 57–64.

[13] B. D. Argall, B. Browning, and M. M. Veloso, “Teacher feedback to scaffold and refine demonstrated motion primitives on a mobile robot,” *Robot. Auton. Syst.*, vol. 59, no. 3-4, pp. 243–255, Mar. 2011. [Online]. Available: http://dx.doi.org/10.1016/j.robot.2010.11.004

[14] M. Elshaw, C. Weber, A. Zochios, and S. Wermter, “An associator network approach to robot learning by imitation through vision, motor control and language,” in *Neural Networks, 2004. Proceedings. 2004 IEEE International Joint Conference on*, vol. 1, July 2004, pp. –596.

[15] S. Calinon, F. Guenter, and A. Billard, “On learning, representing, and generalizing a task in a humanoid robot,” *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, vol. 37, no. 2, pp. 286–298, April 2007.

[16] J. Aleotti, S. Caselli, and M. Reggiani, “Leveraging on a virtual environment for robot programming by demonstration,” *Robotics and Autonomous Systems*, vol. 47, no. 23, pp. 153 – 161, 2004, robot Learning from Demonstration. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0921889004000454

[17] S. Osentoski, B. Pitzer, C. Crick, G. Jay, S. Dong, D. Grollman, H. Suay, and O. Jenkins, “Remote robotic laboratories for learning from demonstration,” *International Journal of Social Robotics*, vol. 4, no. 4, pp. 449–461, 2012. [Online]. Available: http://dx.doi.org/10.1007/s12369-012-0157-8

[18] *Learning to Plan for Constrained Manipulation from Demonstrations*, 2013 2013.

[19] G. Ye and R. Alterovitz, “Demonstration-guided motion planning,” in *International Symposium on Robotics Research (ISRR)*, vol. 5, 2011.

[20] S. Ekvall and D. Kragic, “Robot learning from demonstration: a task-level planning approach,” *International Journal of Advanced Robotic Systems*, vol. 5, no. 3, pp. 223–234, 2008.

[21] C. Chao, M. Cakmak, and A. Thomaz, “Towards grounding concepts for transfer in goal learning from demonstration,” in *Development and Learning (ICDL), 2011 IEEE International Conference on*, vol. 2, Aug 2011, pp. 1–6.

[22] A. Weiss, J. Igelsbock, S. Calinon, A. Billard, and M. Tscheligi, “Teaching a humanoid: A user study on learning by demonstration with hoap-3,” in *Robot and Human Interactive Communication, 2009. RO-MAN 2009. The 18th IEEE International Symposium on*, Sept 2009, pp. 147–152.

[23] M. Forbes, M. J.-Y. Chung, M. Cakmak, and R. P. Rao, "Robot programming by demonstration with crowdsourced action fixes," in *Second AAAI Conference on Human Computation and Crowdsourcing*, 2014.

[24] I. A. Sucan, M. Moll, and L. E. Kavraki, "The open motion planning library," *Robotics & Automation Magazine, IEEE*, vol. 19, no. 4, pp. 72–82, 2012.

[25] D. Gossow, A. Leeper, D. Hershberger, and M. Ciocarlie, "Interactive markers: 3-d user interfaces for ros applications [ros topics]," *Robotics Automation Magazine, IEEE*, vol. 18, no. 4, pp. 14–15, Dec 2011.

[26] M. Tenorth, A. Perzylo, R. Lafrenz, and M. Beetz, "The roboearth language: Representing and exchanging knowledge about actions, objects, and environments," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, May 2012, pp. 1284–1289.

[27] S. Srivastava, E. Fang, L. Riano, R. Chitnis, S. Russell, and P. Abbeel, "Combined task and motion planning through an extensible planner-independent interface layer," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2014. [Online]. Available: http://robotics.eecs.berkeley.edu/pubs/25.html

[28] L. Kaelbling and T. Lozano-Perez, "Hierarchical task and motion planning in the now," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, May 2011, pp. 1470–1477.

Appendix 1: Detailed Skill Model Evaluation

Results evaluation

Participant 1

Kinesthetic Skill - pour - should work

GUI Skill - platter - should work (small constraints)

KGUI Skill - box - should work

Participant 2

Kinesthetic Skill - pour - should work (small constraints)

GUI Skill - pour - should work (small constraints)

KGUI Skill - box - should work

Participant 3

Kinesthetic Skill - box - should work (some plans likely fail)

GUI Skill - platter - likey will not work (second keyframe too low)

KGUI Skill - pour - should work (some plans may fail - grasping keyframe too large)

Participant 4

Kinesthetic Skill - box - should work

GUI Skill - pour - will not work (set to grab cup from above)

KGUI Skill - platter - should work (some may fail - platter keyframe large)

Participant 5

Kinesthetic Skill - pour - will not work (tilt towards bowl not recorded)

GUI Skill - box - should work (small constraints)

KGUI Skill - platter - likey will not work (keyframes too high)

Participant 6

Kinesthetic Skill - platter - should work (but many plans will fail,

large grasping constraint)

GUI Skill - box - may work (depends on box positioning)

KGUI Skill - pour - will not work (tilt motion not recorded)

Participant 7

Kinesthetic Skill - pour - should work (small constraints)

GUI Skill - platter - should work (cup needs to be correctly placed,

some plans may end off platter, nice constraints)

KGUI Skill - box - should work

Participant 8

Kinesthetic Skill - box - should work (small constraints)

GUI Skill - platter - should work (if keyframe high enough -

cup needs to be constraint object)

KGUI Skill - pour - should work (nice constraints)

Participant 9

Kinesthetic Skill - platter - should work (small constraints)

GUI Skill - pour - should work

KGUI Skill - box - should work (small constraints)

Participant 10

Kinesthetic Skill - box - should work (nice constraints)

GUI Skill - pour - wont work (did not tilt all the way)

KGUI Skill - platter - will not work (all keyframes strangely offset - error?)