Robust Robot Learning from Demonstration and Skill Repair Using Conceptual Constraints

Carl Mueller*, Jeff Venicx*, and Bradley Hayes
{carl.mueller, jeff.venicx, bradley.hayes}@colorado.edu

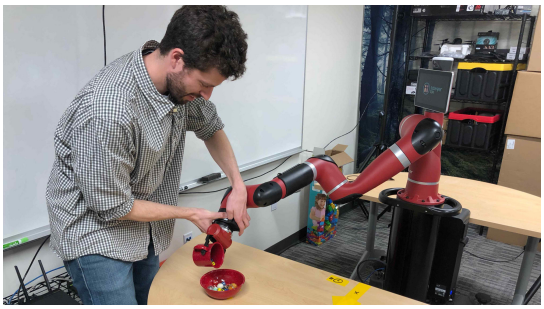
***Abstract*— Learning from demonstration (LfD) has enabled robots to rapidly gain new skills and capabilities by leveraging examples provided by novice human operators. While effective, this training mechanism presents the potential for sub-optimal demonstrations to negatively impact performance due to unintentional operator error. In this work we introduce Concept Constrained Learning from Demonstration (CC-LfD), a novel algorithm for robust skill learning and skill repair that incorporates annotations of conceptually-grounded constraints (in the form of planning predicates) during live demonstrations into the LfD process. Through our evaluation, we show that CC-LfD can be used to quickly repair skills with as little as a single annotated demonstration without the need to identify and remove low-quality demonstrations. We also provide evidence for potential applications to transfer learning, whereby constraints can be used to adapt demonstrations from a related task to achieve proficiency with few new demonstrations required.**

I. Introduction

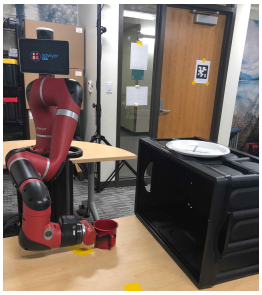
Robot skill Learning from Demonstration (LfD) is a broad label that encompasses techniques enabling robots to acquire skills and behaviors through human guidance. Within an LfD framework, a human teacher typically provides 'ground truth' trajectories or goal states that convey the 'what' and/or 'how' of a skill. Throughout the robotics literature, the terms *learning from demonstration* [1], [2], *programming by demonstration* [3], *apprenticeship learning* [4], and *imitation learning* [5] all tend to refer to this same high level concept.

Learning by example is crucial for facilitating the widespread adoption of robots into society and industry. Beyond endowing robots with the ability to acquire skills that do not have easily crafted objective functions, LfD provides robots with the capability to learn from lay users. By foregoing the need for expert-level knowledge about dynamics, control theory, and programming, automation can be made accessible to a greater diversity of end-users.

The central challenges in learning from demonstration stem from the need to simultaneously extract information about both the relevant feature space (e.g., configuration space, end-effector space, distance between objects, etc.) and allowable feature (co)variances at each step of the skill being taught. By restricting the learning algorithm to learning solely from the trajectories themselves, acquiring robust models of skills can require prohibitively large families of demonstrations, including the need to include experiential data about rare or potentially dangerous situations. In this work we characterize the process of increasing the robustness of a skill as a *repair* task, whether through enabling it to succeed under conditions that the existing behavioral policy does not or by enabling more efficient solutions to an already successful skill.



(a) In the pouring task, the robot must pick a cup off of the table and pour its contents into a bowl without spilling.



(b) The placement task requires a robot to avoid colliding with an object not modeled by its motion planner, testing its ability to adhere to tight motion tolerances.

Fig. 1: Two representative tasks used for evaluating skill repair from demonstration with CC-LfD.

In response to the need for robust skill learning under human guidance, we introduce *Concept Constrained Learning from Demonstration* (CC-LfD): a method for learning and repairing skill policies with minimal additional demonstrations. Central to our method's success is the hypothesis that physical trajectory demonstrations alone are a relatively low-bandwidth signal as compared to the fusion of trajectories with abstract concepts in the form of planning predicates [6]. By introducing a method to propagate constraints from constraint-annotated demonstrations into the entirety of a skill's training set, we are able to achieve rapid skill repair and robust skill learning from demonstration, even if the initial training data or skill model contains errors. Through our

* These authors contributed equally to this work.
Department of Computer Science, University of Colorado Boulder, 1111 Engineering Drive, Boulder, CO USA

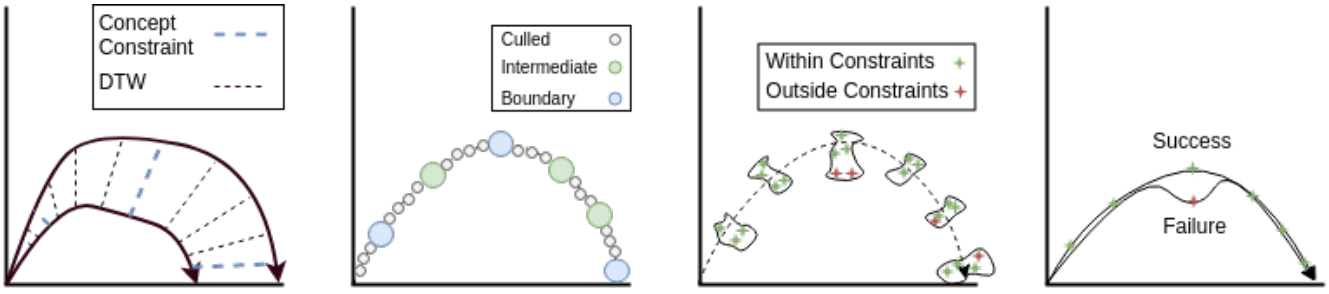


Fig. 2: Illustration of concept constrained skill learning from demonstration. (Left) Two trajectories with Dynamic Time Warping (DTW) applied and illustrated constraint boundaries. (Center-Left) Graphical representation of boundary and intermediate keyframes alongside culled keyframes that violate the variational distance threshold. (Center-Right) Keyframe models with sampled points labeled according to conformity with required constraints. (Right) Final trajectories connecting sampled points, contrasting constraint-aware and unconstrained results.

experiments we show that our method facilitates robust skill learning from demonstration, providing a dramatic reduction in the training data required for skill repair as compared to introducing additional high-quality trajectories.

The three primary contributions of this work are:

- Concept Constrained Learning from Demonstration (CC-LfD), an algorithm for few-shot robust skill learning from demonstration
- An application of CC-LfD to skill repair, enabling CC-LfD to make existing skills more robust to failure with minor additional effort.
- An experimental validation of CC-LfD for skill repair and transfer learning implemented on a manufacturing robot.

II. Background and Related Work

Many modalities have been explored with respect to robot skill learning from novice users, with three of the most popular being graphical, imitation-based, and kinesthetic interfaces. With graphical user interfaces, humans may be tasked with specifying (or editing) a sequence of keyframes for a robot to use when computing a plan [7], [8], [9]. GUIs afford users the potential to visualize the skill captured by the robot's model, potentially enhancing the 'debugging' phase of skill learning from demonstration beyond what is possible without an abstracted interface. In imitation learning [10], [11], a robot is tasked with recovering skills from human demonstrations. This process is generally made more difficult by the correspondence problem [12], where the robot must also deduce the proper mapping between the human's joints and its own such that it can successfully replicate the demonstrated skill. Kinesthetic teaching [13] is a process by which a human physically guides a robot through skill execution, bypassing the correspondence problem and removing many of the difficulties inherent to performing imitation learning or learning through the tele-operation of high degree-of-freedom platforms.

As the robot is receiving 'ground truth' data for executing the learned skill, a crucial decision must be made with respect to how demonstrations are encoded and thus generalized. At the *narrow generalization* end of the spectrum, one may consider recording the entire trajectory and replaying the desired behavior verbatim from a known demonstration. Such an approach is analogous to performing inverse reinforcement learning [14] to learn a policy defined only for states that have been seen before. At the *broad generalization* end of the spectrum, one may consider only recording a set of potential goal states and relying on a motion planner to determine the appropriate trajectory for the robot to follow. Of course, neither of these approaches is likely to capture the intent of the demonstration in a robust manner. An intermediate solution to this problem is to encode a trajectory into a series of keyframes (or, in industrial robotics parlance, *via-points*) [15], [16], [17], providing a connected graph of waypoints for a motion planner to traverse through. As the inter-keyframe distance expands, more of the skill representation is pushed into the motion planner (a search process), thereby potentially providing flexibility during execution (e.g., allowing for obstacle avoidance) [16], [18]. Accordingly, the distillation of keyframes and the interpretation of trajectories between them encompass a key challenge inherent to skill learning from demonstration.

A secondary challenge within learning from demonstration is the verification that the robot's learned model accurately reflects the instructor's intent. Often, it is difficult to know when sufficient training data has been received to cover the diversity of scenarios the skill is expected to perform within. As such, a number of interactive skill repair methods have been proposed to address an identified deficiency in the robot's learned model. Jain et al. [19] propose a co-active learning method that leverages an iterative process for trajectory improvements, wherein a human need only provide slight improvements with successive examples. Apart from methods directly requesting more trajectory demonstrations, earlier work in human-in-the-loop skill learning investigated questions (label, demonstration, and feature queries) that a robot could pose to a human instructor to repair and more quickly learn skills [20]. More recent work has investigated this concept further, showing that human augmentation of a

robot's objective function can be better accomplished with targeted feature queries than comparison-based queries [21].

Algorithm 1: Trajectory Preprocessing

Input: Collection of recorded trajectories C
Output: Labeled and Aligned Trajectories L

```
L, trajectoryLength ← alignWithDTW(C);
for i in range(trajectoryLength) do
    activeConstraints ← {};
    for T in L do
        activeConstraints ← activeConstraints ∪
          getConstraints(L[i]);
    for T in L do
        applyConstraints(L[i], activeConstraints);
/* All demonstration trajectories now have same
   constraint sequence.                          */
return L
```

Relatedly, work from Chao et al. [22] has shown that robots with human-grounded concepts are better positioned to transfer existing knowledge and reason about goal states within unfamiliar tasks. Our work posits that grounded concepts, instantiated by planning predicates (e.g., "is_upright(cup)"), provide a rich, interpretable path for objective function augmentation. Through combinations of predicates associated with segments of a trajectory, one can impose meaningful constraints within non-trivial combinations of features, maintaining intelligibility through a medium of concepts familiar to a human.

Our approach addresses an important technical gap in robot skill learning, providing a contribution to both the keyframing and trajectory planning aspects of LfD, enabling the application of conceptual constraints to keyframes and trajectories (e.g., "keep the cup upright and don't place your end-effector over the computer") during relevant parts of a skill. Recent results on learning behavioral constraints from demonstration within the human-robot interaction community have largely focused on task ordering constraints [23], [24], [25], whereas our approach enables the integration of low-level motion constraints (e.g., "keep the end-effector parallel to the table") as well as high-level conceptual constraints (e.g., "don't let the cup of water and the laptop be near each other"). We accomplish this by integrating constraint enforcement into the trajectory planning (keyframe sampling) process.

III. Methods

Here, we introduce CC-LfD (Concept Constrained Learning from Demonstration), a method of learning from demonstration that leverages both trajectory data and constraints annotated during demonstrations to rapidly produce robust skill representations. Our approach enables few-shot skill learning and repair, as it enables the propagation of constraints from annotated trajectories through existing unannotated training data. It is important to note that the approach to skill repair is to rebuild the learned model incorporating the constraint-annotated (repair) demonstrations, rather than augment an existing model. By leveraging logical combinations of planning predicates, taking the form of boolean state classifiers (e.g., *on_table(cup)*) as constraints, our method appropriately biases waypoint sampling from learned keyframes and effectively models important changepoints that may otherwise go undetected during the keyframe clustering step of trajectory-based learning from demonstration. In this section, we detail both the training and execution phase of CC-LfD by first providing a general intuition for each, followed by more precise technical detail.

A. Skill Training

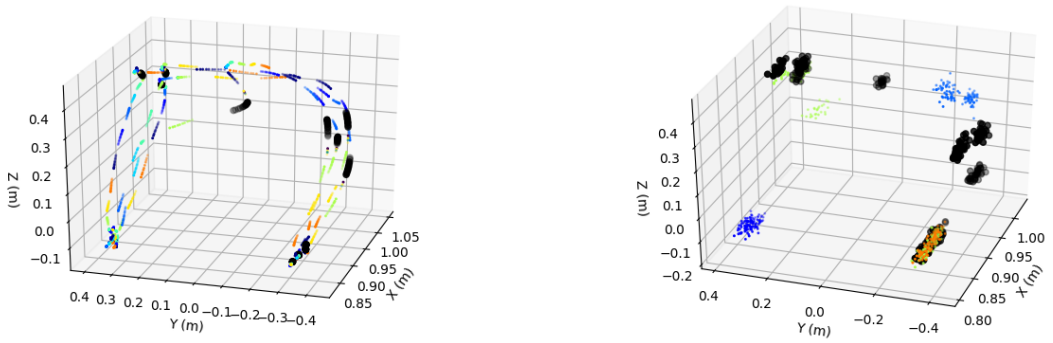
In the training phase, we seek to produce skill models that can later be used by a robot to quickly sample successful trajectories during live execution. To accomplish this our method requires a collection of trajectories $C = \{T_0, T_1, \ldots, T_{|C|}\}$. A trajectory is defined as a sequence of frames (or observations) $T = \{f_0, f_1, \ldots, f_{|T|-1}\}$, each containing a time-stamp (t) and a vector of world state data (s) such that $f = (t, s)$. The world state vector consists of the robots joint configuration, end-effector XYZ position, and end-effector role-pitch-yaw orientation. This world state could include other information from the environment as well, but we restrict the world state to only include data pertaining to the kinematics of the robotic arm itself.

As training trajectories will not generally maintain temporal alignment across frames [26], [27], [28] (e.g., $f_{15} \in T_0$ may not represent the same point in skill execution as $f_{15} \in T_1$), we perform trajectory alignment as a preprocessing step (Fig. 2-left, Algorithm 1) using Dynamic Time Warping (DTW). In contrast to traditional trajectory alignment that strictly minimizes a running distance metric across trajectory

Algorithm 2: Model Formation

Input: Labeled and Aligned Trajectories L, Variational Distance Threshold α
Output: Keyframe Graph G

```
kf_groups ← clusterObservationsIntoKeyframe(L);
kfs ← buildGaussianKernelDensityModels(kf_groups);
for kf ∈ kfs do
    p ← generateSamplePointsFromModels(kf);
    p ← discardConstraintViolations(p,
      getConstraints(kf));
    kf ← buildGaussianKernelDensityModel(p);
G ← buildDirectedGraph(kfs, L);
/* Build directed graph from keyframes using
   ordering information in L                     */
p ← generateSamplePointsFromModels(G);
G ← cullAdjacentOverlappingKeyframes(G,p,α);
/* Remove intermediate keyframes that have a
   variational distance < α with a preceeding
   neighbor                                      */
return G
```



(a) Grouping of trajectory points by keyframe. Black points indicate inclusion within constraint transitions, while colored points represent inclusion within intermediate keyframes.

(b) Keyframe clustering showing sparseness after the culling process. Three boundary keyframes are represented in black, while intermediate keyframes are represented in color. Overlapping clusters occur in these figures due to projection from 6D pose-space into 3D.

Fig. 3: Visualization of trajectories being clustered into keyframes before and after keyframe culling and augmentation.

states, our approach prioritizes the alignment and ordering preservation of constraint boundaries, serving to produce more consistent and informed results. We define a constraint boundary as the frame $f_i \in T_n$ where the currently applied set of constraints changes. This could mean a new constraint is added or removed at that frame.

Given a collection of aligned trajectories, we produce a directed graph where each vertex represents a keyframe: a distribution over state space representing a waypoint within a skill. Given initiation and goal vertices, the sequence of states sampled from the directed path between them produces a series of states for a motion planner to use during execution. In our implementation, we model keyframes using Gaussian Kernel Density Estimation [29].

B. Constraint Application During Demonstrations

During trajectory demonstration, a user can indicate the application of constraints by *annotating* a frame. This is accomplished by communicating a boolean expression (e.g., "is_upright(cup) AND NOT near_laptop(cup)") where each variable consists of a predefined concept at the desired frame of the trajectory. Upon the demonstration's conclusion, each annotated boolean expression of constraints is evaluated from its origination frame onward, with the constraint assumed to apply to each subsequent frame until the expression no longer holds true. This means that when the behavior of the demonstration violates the constraint, the system turns that constraint to false. As an example, a constraint *is_upright(cup)* applied at frame f_a would propagate to all proceeding frames until it is found to be violated by the demonstration at f_b (where $b > a$). The result of constraint annotation and its post-processing is the production of intervals where boolean expressions of concepts must be true. This method of constraint annotation might restrict the types of usable concept constraints (e.g. more abstract constraints) but for the purposes of this paper, the methodology suffices.

C. Trajectory Alignment and Constraint Propagation

Demonstration trajectories are aligned against a chosen reference demonstration using standard DTW [30] with a Euclidean distance cost function. We perform an iterative alignment process where each trajectory is rebuilt based on the warping path provided by the DTW algorithm. This results in repeated points for certain trajectories that may align with multiple points in another trajectory. This iterative process repeats the DTW alignment procedure on the extended trajectories until those trajectories have been extended to an equivalent length (same number of samples). This enables simple uniform sampling of points during the clustering process without having to maintain index alignments across all trajectories. Once the collection of trajectories are aligned, constraints are combined across trajectories as a boolean expression consisting of the logical *AND* of all constraints (both applied and propagated) occurring at that frame index. Thus, for a boolean expression of constraints $b_{m,n}$ occurring at frame f_n of trajectory T_m, we apply

$$b_{m,n} = \bigwedge b_{p,n} \forall p \in \{0, 1, .., |C| - 1\}$$

D. Keyframe Clustering

Traditional keyframe clustering is performed through an unsupervised process that, given a constellation of points in state space, produces clusters that are well separated. As our approach introduces annotated constraint expressions, we utilize this additional information during the clustering step. Our method enforces the creation of *boundary keyframes* consisting of trajectory data from a fixed-size

Algorithm 3: Skill Reconstruction

Input: Keyframe Graph Vertex Sequence V, Samples per keyframe n
Output: Motion Plan M

```
waypoints ← [];
for v ∈ V do
    p ← sampleValidPointsFromKeyframe(v,n);
    p ← discardConstraintViolations(p,getConstraints(v).);
    if p == ∅ then return ERROR;
    waypoints.add(maxLikelihoodPointFromSet(p,v));
M ← createMotionPlan(waypoints);
return M
```

window around frames that lie on a *constraint transition* boundary. As mentioned previously, constraint transitions occur when the applied boolean constraint expressions for consecutive frames differ.

To ensure potentially valuable motion data is not discarded, intermediate keyframes are produced by clustering frames at uniform intervals between boundary keyframes. These uniform intervals are chosen such that sample points within a keyframe are aligned with each other according to DTW alignment process. To prevent overfitting the demonstrations, and thus creating a brittle skill representation, CC-LfD models each keyframe cluster as a probability distribution over state space that we recover from its member frames. In our reference implementation, we model the distribution of states within each keyframe using Gaussian Kernel Density Estimation, choosing a bandwidth parameter value that maintains the majority of probability mass in close proximity to the observed frame. Increasing the kernel bandwidth will expand the distribution model for the keyframe, causing a corresponding increase in sample variation, simultaneously increasing the flexibility of the keyframe and the likelihood of poor skill reconstruction during execution.

To avoid keyframe overlap (which can result in backtracking behavior during skill execution), we perform a culling step after keyframe clustering and modeling is complete, deleting intermediate keyframes whose distributions are too close to an immediately preceeding neighbor in the keyframe graph (Fig. 3, Algorithm 2). This overlap is classified by a prior minimum threshold of variational distance ($\delta(k_i, k_{i-1}) > \alpha$) imposed between the two distributions over an equally sized set of points sampled from each. For a set of n points sampled from Keyframe 0 ($P_{k_0} \sim k_0$) and n points sampled from Keyframe 1 ($P_{k_1} \sim k_1$), the variational distance between keyframe distributions is:

$$\delta(k_0, k_1) = \sum_{p \in P_{k_0} \cup P_{k_1}} |k_0(p) - k_1(p)|$$

E. Constraint-based Keyframe Augmentation

Once the graph of keyframes has been established, we perform a postprocessing step to better fit each learned keyframe distribution to the data and constraints that apply to them. For each keyframe distribution, n points are sampled. If the sampled point satisfies the constraints that apply to the keyframe, it is accepted and added to the keyframe's training set. Otherwise, the point is rejected. (Algorithm 2) As n is increased, two benefits are realized: 1) the model bandwidth parameter can be reduced, lessening the likelihood of outliers, and 2) the distribution will more accurately reflect the projection of the demonstrated trajectory frames onto the manifold where the constraint expression is true, increasing the sampling efficiency of the distribution during runtime.

F. Skill Execution

Within CC-LfD, skill execution is accomplished by sampling waypoints from an ordered sequence of keyframes and subsequently constructing motion plans between them. As the CC-LfD skill model encodes a directed graph of keyframes, any path from an initiation vertex to a goal vertex will produce a valid sequence of keyframes. Once a sequence has been obtained, grounded waypoints from each keyframe are obtained by sampling from each keyframe's distribution (Fig. 2 Center-Right)

As skill execution is likely to occur in an environment that does not precisely match the training environment, such as one with a different configuration of obstacles, points sampled from keyframes may themselves be invalid (e.g., in collision with an obstacle), be infeasible to use in a motion plan (e.g., no valid solution), or violate the required set of constraints for the keyframe. To overcome this limitation, CC-LfD samples multiple states from the keyframe's distribution, rejecting samples that violate one of the aforementioned validity criteria. The sample point with the highest likelihood is retained as a waypoint for the final motion plan (Algorithm 3).

In the event that all of the sampled states from an *intermediate* keyframe are rejected, our approach skips the keyframe and moves on to the next in sequence. This affords flexibility during execution at the expense of potentially sacrificing motion cues learned from the training data, opting to push more burden into the motion planner (as it must now plan across a greater distance) while providing more freedom to the final motion plan (still honoring the imposed constraints). Should all of the sampled states from a *boundary* keyframe be rejected, the skill cannot be executed as no feasible plan could be found that is guaranteed to honor the provided constraints.

IV. Evaluation and Results

A. Robot Platform

We evaluate CC-LfD using the Rethink Robotics Sawyer robot. Sawyer is a 7-degree of freedom robotic arm, with a working envelope of 1260 millimeters and a maximum payload of 4 kilograms. Our CC-LfD reference implementation, including both skill learning and robot control software, is

implemented as a node within Robot Operating System [31] utilizing the MoveIt! motion planning framework [32]. [1]

B. Skill Repair from Poor Initial Demonstrations

We demonstrate the utility of CC-LfD through an evaluation involving skill repair, a domain in which the goal is to make a brittle or ineffective skill model more effective and robust to varying environmental conditions. Poor demonstrations may contain usable signal to help dictate the skill or they might purposefully provide negative signal i.e. what not to do [33]. Thus there is motivation to show that CC-LfD capably operates over poorly demonstrated skills. This evaluation tests the ability of a learning method to absorb positive aspects of sub-optimal or noisy demonstrations while rejecting aspects harmful to the model, requiring a minimum of additional information.

To demonstrate skill repair with CC-LfD, we utilize two tasks common across skill learning from demonstration literature: a compound pick-and-pour task and a precision placement task. The first task involves picking up a cup, moving it over a target region (another cup), and pouring its contents into the target vessel without spilling along the way. This task was chosen due to the complexity of having a constraint that must only be enforced for part of the trajectory (e.g., keep the cup upright) and is violated in other parts.

The second task involves picking up a cup, maneuvering around an obstacle not modeled by the motion planner, and resting the cup atop the obstacle. This task was chosen because of its ability to illustrate adherence to motion constraints for part of the task, requiring certain keyframes to obey spatial rules captured by demonstrations without the benefit of analytical models to assist (e.g., the motion planner's collision avoidance).

Both tasks are evaluated for success according to their objective behavior which is defined by the intent of the task (pouring contents of cup into target, placing cup on resting place) and the expected constraints on the task itself (e.g. cup must remain upright, cup must not collide with hidden obstacle).

For each task, three low-quality demonstrations are provided as a baseline of poor performance to which additional demonstrations must be added (these demonstrations may not be excluded from model training) to repair the learned skill model. Generally, a low-quality demonstration is one that explicitly violates the chosen constraints for a given task. As an example, in the cup pouring task, a poor demonstration might tip the cup too far before it is over a target, prematurely spilling contents. Other low-quality demonstrations may add harmful variance in the keyframes of a learned skill that is likely to degrade execution quality in naïve LFD solutions. We test two approaches to skill repair:

- **CC-LfD:** Train a CC-LfD model with a single constraint-annotated demonstration and the initial low-quality demonstration dataset.

[1]CC-LfD reference implementation is available at: https://github.com/cairo-robotics/.

(a) *Upright* (b) *Threshold* (c) *Violated*

Fig. 4: Examples of the Upright constraint's reference pose, tilt tolerance, and violation.

- **Naïve LfD:** Train a skill model with the addition of successful high-quality demonstrations to the initial low-quality dataset, but with constraints omitted.

In other words, we add high-quality training data to each skill's initial low-quality training set, build a new skill model, and test the model by evaluating its skill executions for success. In the CC-LfD condition we add a single trajectory annotated with boolean constraint expressions during demonstration by the human, while in naïve LfD condition we add a number of high-quality demonstration trajectories without constraint information.

C. Concept Constraints

The evaluation system employs two concepts to showcase the effectiveness of CC-LfD: an object being 'upright' and a minimum end-effector height.

The upright constraint dictates that an object must be upright according to a predefined upright orientation, reference axis, and a threshold angle of deviation that is object-specific. The upright orientation uses a quaternion representation of the end-effector of Sawyer. In other words, we use a specific grasping orientation to represent the 'uprightness' of the cup rather than the orientation of the cup itself. The environment reference axis defines the axis against which rotation deviations are measured. This axis is generally the z-axis relative to the frame of reference of Sawyer. Axis-angle rotations around the reference axis have no bearing on the 'uprightness' of the object. The threshold angle is the limit within which an object is upright compared with its current angle of deviation. In the pouring task and placement task, the upright constraint is used to ensure that the cup is not tilted past its upright threshold angle (Fig. 4).

D. Results

As the robot can only manipulate its 7-DoF arm, an 'upright' constraint applied to a keyframe forces configurations to be sampled where the object is upright within the robot's grasp, while a 'minimum height' concept forces configurations to be sampled where the end effector pose is a minimum height above the table underneath it. The CC-LfD framework supports any type of concept that can be encoded as a boolean classifier over state space.

Pouring Results					
Type	**Poor Demonstrations**	**Repairs**	**Attempts**	**Correct**	**Incorrect**
Baseline	3	0	10	0	10
7 Repairs	3	7	10	4	6
14 Repairs	3	14	10	6	4
21 Repairs	3	21	10	7	3
28 Repairs	3	28	10	6	4
Constrained	**3**	**1**	**10**	**10**	**0**

(a) Pouring Task performance results. With CC-LfD, a skill repaired with a single constraint-annotated demonstration achieves a 100% success rate.

Placement Results					
Type	**Poor Demonstrations**	**Repairs**	**Attempts**	**Correct**	**Incorrect**
Baseline	3	0	10	1	9
7 Repairs	3	7	10	5	5
14 Repairs	3	14	10	4	6
21 Repairs	3	21	10	7	3
28 Repairs	3	28	10	7	3
Constrained	**3**	**1**	**10**	**9**	**1**

(b) Placement Task performance results. CC-LfD with a single repair trajectory achieved a 90% success rate.

Fig. 5: Skill execution success rates for two tasks across one baseline and five skill repair experimental conditions.

E. Task Evaluation Criteria

The success of each task execution is determined by observing whether the robot's executed trajectory satisfies the evaluation criteria below.

1) Pouring Task: In the cup pouring task, the robot must lift the cup off the table, carry the cup above a certain height until it is over top of a target, then lower and pour its contents into the target without spilling along the way. The task is considered a failure if at any point the robot violates these conditions (Fig. 1 a).

2) Placement Task: In the placement task, the robot must lift the cup off the table and place the cup on top of a sideways crate (Fig. 1 b). Success for this task requires the robot to place the object on top of the crate without collisions or spills, carrying the cup above a safe height over the crate until it is over top of the placement zone. If at any point before placement a disqualifying event occurs, the entire task is considered a failure.

For each task we evaluate the success or failure of the robot's performance based on the criteria above. We present results from six experimental conditions investigating different levels of skill repair, each consisting of ten trials per task with skill models trained using varying amounts of low-quality (LQ) and high-quality (HQ) demonstration trajectories. A LQ demonstration fail to successfully meet the given task evaluation criteria. A HQ demonstration properly executes the task according to the given task evaluation criteria. A constraint-annotated trajectory is one that is performed correctly while also being annotated with concept constraints.

- **Baseline:** 3 LQ trajectories
- **7 Repairs:** Baseline + 7 HQ trajectories
- **14 Repairs:** Baseline + 14 HQ trajectories
- **21 Repairs:** Baseline + 21 HQ trajectories
- **28 Repairs:** Baseline + 28 HQ trajectories
- **Constrained:** Baseline + 1 HQ constraint-annotated trajectory

Our results (Fig. 5) show that a single constrained demonstration is enough to repair the poorly trained baseline skill for both tasks, with sufficient variation across executions to guarantee that this is not an artifact of model overfitting. The low-quality baseline demonstrations result in incorrect skill performance nearly always, while a single constrained repairing demonstration results in nearly perfect skill performance, maintaining the (allowable) feature variances provided by the baseline training trajectories. The single failure during the constrained condition for the placement task occurred due to a minor obstacle collision that, while not affecting the final placement, violated the collision avoidance success criteria.

Importantly, we observe that while the introduction of additional high-quality demonstrations shows a trend of improvement over the baseline, it does not quickly converge to a high level of success. For both tasks, even twenty-eight unconstrained HQ repairing demonstrations is not enough to overcome the problems the model inherits from the initial LQ training data. In all cases, states are sampled from distributions that contain the LQ trajectories, but by using CC-LfD the harmful aspects of these demonstrations are successfully discarded while potentially informative signal is maintained.

F. Applications to Transfer Learning

The keyframe constraint optimization performed by CC-LfD can also be used to effectively generalize skills across contexts where interpretations of constraints differ. As an empirical proof-of-concept, CC-LfD is able to generalize the pouring task to a new cup requiring a grasp that is orthogonal to the grasp encountered during training (and thus, requiring a very different trajectory through configuration space). The application of the upright constraint to the new cup universally results in skill failure, as at least one of the boundary keyframe distributions is unable to produce any viable samples that conform to the required constraints.

By introducing a single demonstration performed under the new conditions, CC-LfD is able to learn a model that can perform the skill correctly, despite the fact that the rest of its training data was performed such that it never exhibited the correct upright behavior. This example suggests skill transfer as a promising application of future work extending CC-LfD.

V. Conclusion

In this work, we present *Concept Constrained Learning from Demonstration* (CC-LfD), a novel algorithm for robot skill learning from human teachers. CC-LfD enables a robot to both learn robust skill policies from kinesthetic demonstration and to repair existing skills through a minimal amount of additional demonstrations. Our results show that the presence of a few low-quality training trajectories can have a dramatic, negative impact on skill execution that

is difficult to overcome even with many additional high-quality trajectories. We demonstrate that through CC-LfD, a single well-performed constraint-annotated demonstration can dramatically repair poor skill performance, increasing skill robustness with minimal additional data. Our evaluation confirms that CC-LfD is a time-efficient mechanism for LfD skill repair, obtaining a remarkable decrease in required additional training to overcome noisy or low-quality demonstrations.

A key benefit of CC-LfD is that it does not require the identification and removal of existing low-quality demonstrations for successful skill repair. This is significant because these trajectories may still encode useful information, and the introduction of conceptual constraints may serve to preserve this signal. Our preliminary empirical result in applying CC-LfD to skill transfer offers support to the hypothesis that useful information may still be preserved from off-task (but related) demonstrations, effectively serving as a heuristic for exploration in the constrained space.

REFERENCES

[1] 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.

[2] 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.

[3] A. Billard, S. Calinon, R. Dillmann, and S. Schaal, "Robot programming by demonstration," in *Springer handbook of robotics*. Springer, 2008, pp. 1371–1394.

[4] P. Abbeel and A. Y. Ng, "Apprenticeship learning via inverse reinforcement learning," in *Proceedings of the twenty-first international conference on Machine learning*. ACM, 2004, p. 1.

[5] G. M. Hayes and J. Demiris, *A robot controller using learning by imitation*. University of Edinburgh, Department of Artificial Intelligence, 1994.

[6] S. J. Russell, P. Norvig, J. F. Canny, J. M. Malik, and D. D. Edwards, *Artificial intelligence: a modern approach*. Prentice hall Upper Saddle River, 2003, vol. 2, no. 9.

[7] A. Kurenkov, B. Akgun, and A. L. Thomaz, "An evaluation of gui and kinesthetic teaching methods for constrained-keyframe skills," in *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*. IEEE, 2015, pp. 3608–3613.

[8] S. Alexandrova, M. Cakmak, K. Hsiao, and L. Takayama, "Robot programming by demonstration with interactive action visualizations." in *Robotics: science and systems*, 2014.

[9] M. Stenmark, M. Haage, and E. A. Topp, "Simplified programming of re-usable skills on a safe industrial robot: Prototype and evaluation," in *Proceedings of the 2017 ACM/IEEE International Conference on Human-Robot Interaction*. ACM, 2017, pp. 463–472.

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

[11] S. Calinon and A. Billard, "Incremental learning of gestures by imitation in a humanoid robot," in *Proceedings of the ACM/IEEE international conference on Human-robot interaction*. ACM, 2007, pp. 255–262.

[12] C. L. Nehaniv and K. E. Dautenhahn, *Imitation and social learning in robots, humans and animals: behavioural, social and communicative dimensions*. Cambridge University Press, 2007.

[13] B. Akgun, M. Cakmak, J. W. Yoo, and A. L. Thomaz, "Trajectories and keyframes for kinesthetic teaching: A human-robot interaction perspective," in *2012 7th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, March 2012, pp. 391–398.

[14] A. Y. Ng, S. J. Russell, *et al.*, "Algorithms for inverse reinforcement learning." in *Icml*, 2000, pp. 663–670.

[15] B. Akgun, M. Cakmak, K. Jiang, and A. L. Thomaz, "Keyframe-based learning from demonstration," *International Journal of Social Robotics*, vol. 4, no. 4, pp. 343–355, 2012.

[16] N. Vuković, M. Mitić, and Z. Miljković, "Trajectory learning and reproduction for differential drive mobile robots based on gmm/hmm and dynamic time warping using learning from demonstration framework," *Engineering Applications of Artificial Intelligence*, vol. 45, pp. 388–404, 2015.

[17] B. Akgun and A. Thomaz, "Simultaneously learning actions and goals from demonstration," *Autonomous Robots*, vol. 40, no. 2, pp. 211–227, 2016.

[18] E. Pignat and S. Calinon, "Learning adaptive dressing assistance from human demonstration," *Robotics and Autonomous Systems*, vol. 93, pp. 61–75, 2017.

[19] A. Jain, B. Wojcik, T. Joachims, and A. Saxena, "Learning trajectory preferences for manipulators via iterative improvement," in *Advances in neural information processing systems*, 2013, pp. 575–583.

[20] M. Cakmak and A. L. Thomaz, "Designing robot learners that ask good questions," in *Proceedings of the seventh annual ACM/IEEE international conference on Human-Robot Interaction*. ACM, 2012, pp. 17–24.

[21] C. Basu, M. Singhal, and A. D. Dragan, "Learning from richer human guidance: Augmenting comparison-based learning with feature queries," *13th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, 2018.

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

[23] B. Hayes and B. Scassellati, "Discovering task constraints through observation and active learning," in *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*. IEEE, 2014, pp. 4442–4449.

[24] ——, "Autonomously constructing hierarchical task networks for planning and human-robot collaboration."

[25] S. Ekvall and D. Kragic, "Robot learning from demonstration: a task-level planning approach," *International Journal of Advanced Robotic Systems*, vol. 5, no. 3, p. 33, 2008.

[26] A. Vakanski, I. Mantegh, A. Irish, and F. Janabi-Sharifi, "Trajectory learning for robot programming by demonstration using hidden markov model and dynamic time warping," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 42, no. 4, pp. 1039–1052, 2012.

[27] B. Hayes and J. A. Shah, "Interpretable models for fast activity recognition and anomaly explanation during collaborative robotics tasks," in *Robotics and Automation (ICRA), 2017 IEEE International Conference on*. IEEE, 2017, pp. 6586–6593.

[28] A. Coates, P. Abbeel, and A. Y. Ng, "Learning for control from multiple demonstrations," in *Proceedings of the 25th international conference on Machine learning*. ACM, 2008, pp. 144–151.

[29] R. O. Duda, P. E. Hart, D. G. Stork, *et al.*, *Pattern classification*. Wiley New York, 1973, vol. 2.

[30] E. Keogh and C. A. Ratanamahatana, "Exact indexing of dynamic time warping," *Knowledge and information systems*, vol. 7, no. 3, pp. 358–386, 2005.

[31] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "Ros: an open-source robot operating system," in *ICRA workshop on open source software*, vol. 3, no. 3.2. Kobe, Japan, 2009, p. 5.

[32] S. Chitta, I. Sucan, and S. Cousins, "Moveit![ros topics]," *IEEE Robotics & Automation Magazine*, vol. 19, no. 1, pp. 18–19, 2012.

[33] D. H. Grollman and A. Billard, "Donut as i do: Learning from failed demonstrations," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. IEEE, 2011, pp. 3804–3809.