Simultaneous End-User Programming of Goals and Actions for Robotic Shelf Organization

Ying Siu Liang, Damien Pellier, Humbert Fiorino, Sylvie Pesty, Maya Cakmak

Simultaneous End-User Programming of Goals and Actions for Robotic Shelf Organization

Ying Siu Liang[1], Damien Pellier[1], Humbert Fiorino[1], Sylvie Pesty[1], and Maya Cakmak[2]

***Abstract*— Arrangement of items on shelves in stores or warehouses is a tedious, repetitive task that can be feasible for robots to perform. The diversity of products that are available in stores and the different setups and preferences of each store makes pre-programming a robot for this task extremely challenging. Instead, our work argues for enabling end-users to customize the robot to their specific objects and setup at deployment time by programming it themselves. To that end, this paper contributes (i) a task representation for shelf arrangements based on a large dataset of grocery store shelf images, (ii) a method for inferring goal configurations from user inputs including demonstrations and direct parameter specifications, and (iii) a system implementation of the proposed approach that allows simultaneously learning task goals and actions. We evaluate our goal inference approach with ten different teaching strategies that combine alternative user inputs in different ways on the large dataset of grocery configurations, as well as with real human teachers through an online user study (N=32). We evaluate our full system implemented on a Fetch mobile manipulator on eight benchmark tasks that demonstrate end-to-end programming and execution of shelf arrangement tasks.**

I. INTRODUCTION

The supermarkets and grocery stores industry employs millions of workers for tedious tasks, including restocking and facing products on shelves[1]. These tasks have certain regularities that can be exploited by automation solutions. For instance, most objects are rectangular prisms (boxes) or cylinders (cans) and they are often organized on a shelf in a grid pattern with the label facing forward. On the other hand, the arrangement task is slightly different for every item, with varying grid configuration parameters (rows, columns, stacks, or object distances) due to differences in shelf space, product types, and product dimensions. Furthermore, different robot end-effectors require different ways of manipulating objects so as to get them tightly arranged in confined shelf settings. As a result, developing universal robotic shelf arrangement capabilities that work for all possible items, in all possible stores is extremely difficult.

Instead, our work argues for robot shelf arrangement tasks to be programmed by end-users at the time of deployment. Rather than developing universal capabilities, we embrace the idea that a robot will be customized to a specific store and the specific items in it. While this presents a simpler programming challenge, enabling end-users to do it is non-trivial. The system needs to not only be intuitive and easy to use, but it should also allow efficient programming of both *what* the desired arrangement of items looks like and *how* the robot can use its manipulator to move each item to the desired position relative to other items. In this paper we tackle this challenge.

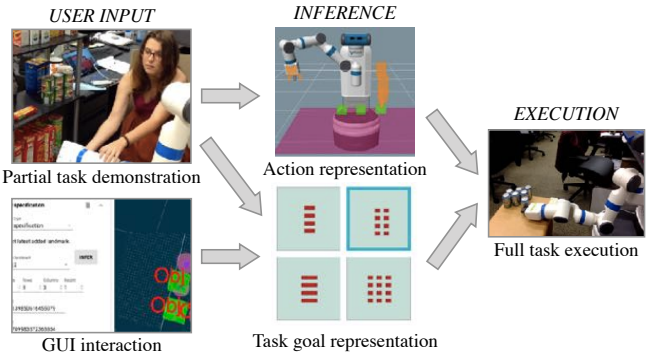

Fig. 1. Overview of the developed system that allows users to demonstrate part of a shelf arrangement task and interact with a GUI to simultaneously program both the complete task goal (fully specified shelf arrangement) and the actions for achieving that goal.

We propose an approach for enabling users to simultaneously program robotic shelf arrangement task goals and actions for a given item, by demonstrating a few steps of the shelf stocking task. We develop a user interface to visualize inferred task goals and actions, as well as enable other user input to augment their demonstrations. To better understand common structure in shelf arrangement tasks, we analyzed the Freiburg dataset of close to 5,000 images covering more than 2,400 unique grocery items from 25 different categories [1]. Our goal inference model, based on this dataset, takes demonstrations and other input from the user and proposes them most likely shelf arrangements in order to accelerate the teaching process. We analyze how quickly correct goal configurations in the dataset can be inferred from user input according to 10 different teaching strategies. We present an online user study that empirically investigates strategies that people use in a simplified arrangement task domain. We implement our approach on a Fetch mobile manipulator and demonstrate the programming and execution of 8 arrangement tasks for objects from the dataset.

II. RELATED WORK

Our work relates to several topics explored in previous robotics research. The larger umbrella of end-user robot

[1]Y.S. Liang, D.Pellier, H.Fiorino, S.Pesty are with the University Grenoble Alpes, LIG, F-38000 Grenoble, France {liangyi, pellierd,fiorinoh,pestys}@univ-grenoble-alpes.fr

[2]M. Cakmak is with the Paul G. Allen School of Computer Science & Engineering, University of Washington, Seattle, Washington 98195, USA mcakmak@cs.washington.edu

[1]https://www.statista.com/statistics/473811/supermarket-and-grocery-stores-employment-in-the-us/

programming has seen a range of recent work focused on programming of mobile robots [2], social robots [3], [4], [5], industrial manipulators [6] and mobile manipulators [7], [8]. The most widely explored approach is Programming by Demonstration (PbD) which involves taking demonstrations of a task as input and inferring the goal of the task or a policy that can be used to accomplish the task [9], [10]. A majority of the work focuses on directly learning a policy or modeling higher level actions from lower level control signals [11], [12], [13], [14], while some explore learning task goals or task structure represented in various ways [15], [16], [17], [18], [19]. Most closely related to our work, Akgun *et al.* explored simultaneous learning of actions and goals by demonstration, focusing on manipulation tasks, such as closing a box and pouring beans into a bowl [20].

Robot shelf stacking was part of the Amazon picking challenge in the last iteration [21]. Teaching robots to tidy up shelves was addressed previously by Abdo et al. [22], but their focus is on dividing different products into categories, rather than configuring them on a shelf.

Our work argues for using end-user programing for teaching task goals and robot actions to perform arrangement of grocery items on shelves. Shelf arrangements are different in every store; only the store owners or staff can correctly specify the task goals for the robot. Hence, the use of end-user programming is essential for that part of the problem. Previous work has explored alternative interfaces, such as speech or different GUIs for specifying task goals [7], [23], [24]. On the other hand, programming of actions is not necessarily tied to end-user programming. An alternative is to give robots universal capabilities for grasping any object, planning a motion to move it without collisions, and placing it in any desired configuration. Motion plans could further involve non-prehensile actions [25], [26] to reconfigure objects closer together as in some of the actions programmed by demonstration in this work. While a lot of research in robotics focuses on giving robots those capabilities, they are still far from being universal. Another approach is for the robot to learn those actions by training on the job *i.e.,* self-exploration using reinforcement learning, but is likely undesirable due to long exploration times and negative impacts of trial errors.

III. APPROACH

Our work aims to enable a user to demonstrate part of the task of arranging an item on a shelf and have the robot complete the task on its own. From the few user demonstrations, the robot needs to infer both *what* the desired arrangement of objects is and *how* to use its end effector to configure objects in the desired relative configuration. While both of these are well studied problems, their joint inference is common in previous work (*e.g.,* [20]). The specific challenge addressed in this paper is the efficient inference of the task goal from few demonstrations that convey both goal and action. To that end we explore two practical ideas:

Fig. 2. Main product shape categories, that comprise 90.1% of items in the dataset, are defined based on the the shape of their base and their tip. The shape of the tip determines whether the objects can be stacked or not. The shape of the base determines whether the object is more compactly arranged on a regular grid (rectangular) or an off-grid arrangement (circular).

1) We perform a thorough analysis of the task domain to come up with compact domain-specific task representations that exploit common structure.
2) We augment the user's input with direct specification of certain task parameters that are less efficiently conveyed through demonstrations.

We detail this approach in the following subsections.

A. Freiburg Dataset Analysis

The Freiburg dataset [1] contains 4,947 images of common grocery items and was originally collected for training visual classifiers. We labeled the images according to product type and shelf arrangement. For the shelf arrangement data we excluded 998 (20.2%) images that either did not show a supermarket shelf (*e.g.,* product on the table or vending machine) or that were duplicate images of a product on a shelf already included. The remaining $3,949$ were used for understanding the structure of shelf arrangement tasks and defining a common task representation (Sec. III-B).

Our first observation is that almost all items in stores are arranged on a 2D or 3D *grid*. We considered *rows* to be the depth (front-to-back), *columns* the width (left-to-right), and *stacks* to be the height (bottom-to-top) of the grid. As the dataset focused more on product types rather than product configuration, the configuration parameters were sometimes not all clearly identifiable. The number of rows was often not visible but likely to be filled for the entire depth of the shelf. Thus, we coded the specific number of columns and stacks and assumed rows to be *as many as possible*.

Product packaging is often designed with the objective of compact packaging and efficient use of shelf space in stores. As a result a large portion of items (54.3% in our dataset) have a flat top surface that allow *stacking*, of which 33.9% are rectangular prisms (*i.e.,* boxes) and 20.4% are cylinders (*i.e.,* cans). Non-stackable items also have rectangular or circular bases to stand stably on the shelf surface, but less regular tops (*e.g.,* bottle lids) that prevent stable stacking. We consider these objects equivalent to square pyramids (19.5%) and cones (16.3%). Only 9% of the items did not fit into the shape categories mentioned above and were categorized as soft packaging (8.7%) or other (1.3%), *e.g.,* triangular pyramid or hexagon prism.

Fig. 3. Cylindrical item arranged in (a) regular grid and (b) off-grid configurations. (c) Soft-packaged item arranged in interleaved grid configuration.

TABLE I
THE DISTRIBUTION OF SHELF ARRANGEMENT PARAMETER VALUES ACROSS THE FREIBURG DATASET.

	stacks					
columns	1	2	3	4	>4	
1	49.4%	4.4%	0.7%	0.2%	2.3%	**57%**
2	29.6%	3.4%	0.4%	0.0%	1.4%	**35%**
3	5.8%	0.5%	0.0%	0.0%	0.0%	**6%**
4	1.3%	0.2%	0.0%	0.0%	0.1%	**2%**
5	0.1%	0.0%	0.0%	0.0%	0.0%	**0%**
6	0.0%	0.0%	0.0%	0.0%	0.0%	**0%**
7	0.0%	0.0%	0.0%	0.0%	0.0%	**0%**
	86%	**8%**	**1%**	**0%**	**4%**	**100%**

We also observe that items with rectangular bases are most compactly arranged on a grid with both side surfaces touching the neighboring items (Fig. 3a). Whereas objects with circular bases can be more tightly arranged when two consecutive rows are offset by half the radius length of the item, which we refer to as an *off-grid* arrangement (Fig. 3b). Despite the compactness advantage, only a small percentage (1.75%) of the dataset included off-grid configurations. Although a majority of items were observed to be arranged as compactly as possible (touching neighboring items on all sides), some arrangements left space between objects, possibly to allow customers to take items without knocking down others. We observed that at least 0.26% of arrangements left space between items either on the row or column direction (not possible in the stacking direction due to gravity).

B. Shelf Arrangement Representation

Based on the common product arrangements observed in the dataset, we propose a simplified grid-based task representation with the following variables: number of columns (m), rows (n), stacks (s), row distance (d^n) and column distance (d^m), where $m, n, s \geq 1$, and $d^m, d^n \in \{0, 1\}$ represent the binary state for touching and not touching respectively. Thus, we represent a shelf arrangement task (where the shelf is orthogonal to the robot) as a tuple $\tau = (n, m, s, d^n, d^m)$. Table I shows the distribution of the values for these variables observed in the Freiburg dataset. Note that this representation excludes soft-packaged or irregularly shaped items, as well as off-grid arrangements. However, the representation could easily be extended for more complex tasks, such as continuous values for d^n and d^m, variables describing off-grid configurations, or variables for orientation of the items.

C. Goal Inference from Demonstration

The problem of goal inference is the estimation of most likely shelf arrangement based on the input obtained from the user. We represent this as a maximum a posteriori estimation problem, *i.e.,* :

$$\tau^* = \arg\max_{\tau_i \in T} P(\tau_i | D) \tag{1}$$

where T is the set of all possible shelf configurations and $D = \{A_j, O_j\}_{j=1}^{M}$ is the set of M demonstrations provided by the user with A_j actions each leading to the placement of one additional object on the shelf. Hence, each O_j corresponds to configurations $(x, y, \theta, s) \in (\mathbb{SE}2 \times \mathbb{N})$ (s for stack number) of j objects that have been placed so far on the shelf surface or on top of another object. Since the demonstrations are assumed to be progressions of the same shelf configuration task, the goal inference only depends on the latest shelf configuration O_M rather than all of D.

Given the M object configurations in O_M we estimate the current shelf arrangement $\tau_M = (n_M, m_M, s_M, d^n_M, d^m_M)$ as follows. The number of rows and columns are determined by separately finding alignments (*i.e.,* values within a small distance) in x and y dimensions of the objects. The number of aligned groups gives the number of estimated rows and columns (n_M, m_M). The number of stacks (s_M) is estimated as the highest stack number of any object in the demonstrated configuration. The contact between rows and columns (d^n_M, d^m_M) are estimated based on the majority relation between rows and columns.

Next, we substitute τ_M for D and use the Bayes Rule to calculate the posteriors of the term we would like to maximize as follows:

$$P(\tau_i | \tau_M) = \frac{P(\tau_M | \tau_i) P(\tau_i)}{\sum_{\tau_j \in T} P(\tau_M | \tau_j) P(\tau_j)}$$

We assume that the probability $P(\tau_M | \tau_i)$ is zero for all τ_i whose row, column, and stack parameter values are already exceeded in τ_M. Similarly, any arrangement τ_i whose binary variable (d^n, d^m) is contradicted in the demonstration is considered zero. Note that d^n and d^m are only relevant if there are at least two rows or columns respectively (n>1, m>1). We initialize the priors $P(\tau_i)$ based on their observed occurrence in the dataset (Sec. III-A). To have a finite set of possible shelf configurations we bound the grid parameters based on the dataset as well, where $m \in [1..7]$, $n \in [1..2]$ and $s \in [1..5]$, where $n = 2$ corresponds to having *as many rows as possible* based on the shelf, rather than an exact number of rows.

While the computation in Eqn. 1 gives a single most likely configuration, we propose to give the top few arrangements with the highest likelihoods as *suggestions* to the user. This requires ranking all possible arrangements in terms of their likelihoods, given the demonstration. Since ties are possible due to equal priors we rank configurations with smaller parameter values as higher.

D. Goal Inference with Direct Specification

To allow the user to directly specify goal shelf arrangement parameters to augment demonstrations, we exploit the way that our goal inference works as described in Sec. III-C. If the user directly specifies the value of a goal arrangement parameter, all arrangements that are inconsistent with that specification are considered to have zero probability. This corresponds to a slight change in the formulation, where the input from the user relevant for the inference does not only include τ_M but also any direct specification $\mathcal{S}$, where $P(\tau_i|\mathcal{S}, \tau_M)$ is zero if τ_i is inconsistent with $\mathcal{S}$. This allows for a fast elimination of many candidate arrangements.

E. Action Representation and Inference

We use a simple action representation proposed in previous work [27], [28]. We assume that the robot can perceive the configuration of the shelf and of each object on it. Each of these perceived entities are considered as potential *landmarks*. The action is then represented as a sparse sequence of end-effector poses relative to one of the landmarks and gripper states (open/close). For example, placement of an object next to another object could be done by moving the gripper to a pose near the reference object, lower the gripper, open the gripper, clear the gripper from potential collisions. All of these poses would be relative to the reference object. Non-prehensile actions, such as moving the placed object closer to the reference object, could also be easily represented in the same way, as poses of the gripper relative to detected objects.

Such actions can be directly programmed with a single demonstration using heuristic assignment of poses to landmarks. The assignment can then be corrected by the user [28] or through multiple demonstrations of the same step [29]. We take the first approach; hence an individual action demonstration in A_j is already an executable action. While there can be ways to combine multiple demonstrations, in our implementation the robot randomly chooses an action from the subset of A_j that is reachable based on the configuration of landmarks in the scene. Although all demonstrated actions should be reachable for the objects in the shelf arrangement that were placed during the demonstration, some actions may become unreachable when they are shifted for placement of other objects to complete the desired shelf arrangement. Hence having multiple demonstrations in A_j is still valuable.

Execution of actions simply involved detecting all landmarks in the environment, computing the actual configuration of end-effector poses that are relative to those landmarks, and then moving through the landmarks one by one.

F. Implementation

We implemented our approach on a Fetch mobile manipulator which has a 7-DoF arm with a load capacity of 6kg. Our system builds on the open-source system Rapid PbD[2] which implements the action representation described in Sec. III-E. The robot detects the shelf surface and objects on it using standard Point Cloud Library functionality. As stacked objects are perceived as one cluster on the surface, we detect stacking from height changes that are approximately equivalent to the height of the object measured at the demonstration of the first action.

[2]https://github.com/jstnhuang/rapid_pbd

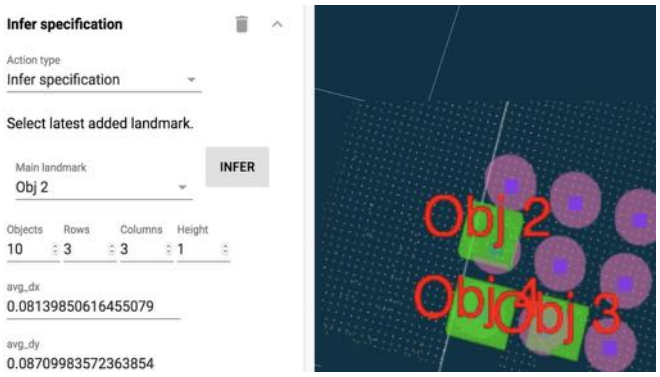

Fig. 4. The extended Rapid PbD interface with the inferred object arrangement (purple) visualized overlaying the detected objects (green).

We extend the Rapid PbD graphical user interface (GUI) in several ways. Users first create an empty program using the GUI. Then they kinesthetically move the robot arm to desired poses and save poses or change the gripper state through the GUI. We assume that the items to be placed on the shelf are always picked up from the same location. Demonstrations involve a sequence of picking, moving, placing, and possibly non-prehensile readjusting of one object relative to the shelf or to another object already on the shelf.

When the user confirms the end of an action demonstration the robot perceives the end state of the demonstration O_M, which is expected to include M objects. The robot then infers τ_M as described in Sec. III-C and visualizes the most likely configuration overlaid onto the detected objects that are already placed (Fig. 4). In our implementation the average row and column distances ($d^m, d^n \geq 0$) are read from the demonstrated arrangement and off-grid arrangements are recognized. The user can create interleaved arrangements (Fig. 3c) by changing d^m, d^n on the GUI. It also displays the top four arrangements that have the highest likelihood given the demonstrations so far. The GUI allows the user to select an arrangement from this list or directly specify task goal parameters (Sec. III-D). When the goal for the shelf arrangement is confirmed, the robot can execute the complete shelf arrangement task either continuing from what has already been demonstrated or starting from scratch.

Our interface also allows users to reuse previously programmed tasks and actions for shelf arrangement with different objects or grid parameters. To that end the user can set up full or partial shelf arrangements and then run the object detection and goal inference. They can also change task parameters to match the new task. Programmed manipulation actions can also be transferred in some cases, but need to be tested and possibly adjusted to work with different objects.

IV. GOAL INFERENCE EVALUATION

We evaluate our approach in three ways. First, in this section we present (1) a systematic evaluation of the goal inference on the Freiburg dataset and (2) a user evaluation through an online study with a simplified interface. Next, in Sec. V we present the full system evaluation.

A. Teaching Strategies

Our system provides the user with different possibilities for programming shelf arrangement tasks. As described in Sec. III-C, at any point during the programming process the user has three possible actions:

- *Demonstrate* object placement kinesthetically,
- *Specify* grid parameters on the interface,
- *Select* an arrangement from most likely arrangements,

The user can have different strategies for using a combination of the different actions to minimize cost. Often times, learning algorithms are evaluated with *sample efficiency*, *i.e.,* the number of examples needed to learn a concept. Similarly, users of our system can try to minimize the number of programming actions they need to take in order to correctly teach a shelf arrangement. Alternatively, each action can be associated with a different cost, such as *clock time* and the user can try to minimize that cost.

Given the three programming actions above, we devised the following programming strategies:

1) Naive demonstration (ND) involves demonstrating the full shelf arrangement filling up rows, columns, and stacks in progression.
2) Optimal demonstration (OD) involves giving the minimum number of demonstrations to make the inference correct; *i.e.,* filling up one row, one column and one stack.
3) Demonstrate and specify (D&S) involves demonstrating distances d^m and d^n and specifying remaining variables n, m, s.
4) Naive specification (NS) involves specifying all values in order of n, m, s, d^n, d^m.
5) Prior-aware specification (PaS) involves only specifying values that are different from the most likely values according to the prior.

These programming strategies are akin to optimal teaching algorithms that can be devised for a given concept and known learner [30], [31]. The selection action can be used in combination with any of these strategies, therefore resulting in 10 teaching strategies. We assume that the interface displays the top four most likely configurations. Hence if the most desired arrangement is in the top four, the user can directly select this configuration and complete the programming process. Note that in order to learn the action as well as the task goal, the robot needs to obtain at least one demonstration from the user, which is not the case for all strategies. However, for the purposes of the analysis of goal inference, we ignore that fact. In practice, those strategies that do not involve any demonstrations would require at least one demonstration in addition to the other programming steps.

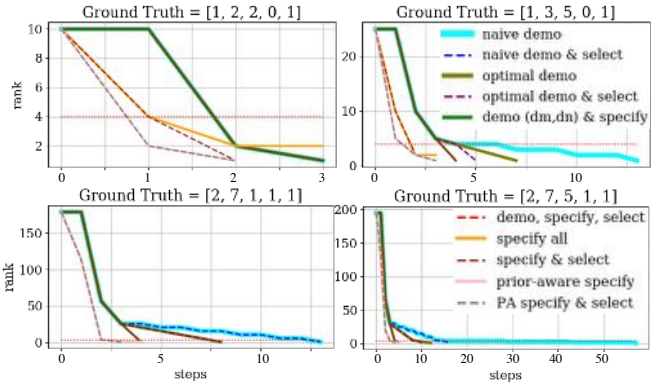

Fig. 5. Learning curve of 10 teaching strategies for 4 different shelf arrangements from the Freiburg dataset, with number of actions (x-axis) and ranking of the ground truth shelf arrangement (y-axis).

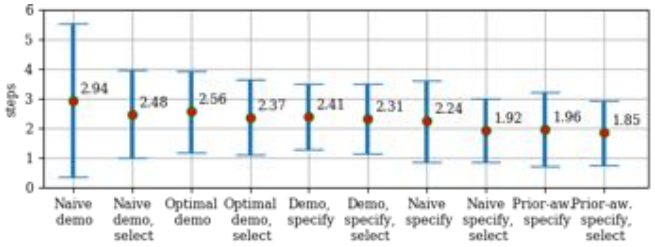

Fig. 6. Average number of steps required to infer the correct arrangement per teaching strategy across the dataset (error bars show standard deviation).

B. Evaluation on the Freiburg Dataset

We measured the performance of each teaching strategy in terms of the number of steps required to infer the desired arrangement across the Freiburg dataset. Fig. 5 shows the learning curves for the ten teaching strategies for four example shelf configurations. We observe that for simpler configurations ($m, n, s \leq 2$), strategies that involve demonstrations are comparable or better than specification strategies, because (1) d^n, d^m are automatically inferred from the object placements and (2) our priors are based on the dataset where simpler configurations are ranked higher.

For higher m, n, s values, the naive demonstrations become infeasible, optimal demonstrations are bounded by $(n + m + s - 2)$ steps, while specification strategies are always bounded by a maximum of 5 steps. The version of each strategy that involves a final selection step from the top four arrangements is more efficient or the same (in terms of number of actions) as the original strategy, since it takes fewer demonstrations or specifications to get the desired arrangement in top four versus top one. The most efficient strategy is the *prior-aware specify+select* strategy requiring 1.85 steps on average.

Overall, the average number of steps required to teach shelf configurations for all teaching strategies (Fig. 6) is low (between 2-3 actions) despite some outlier cases (*e.g., naive demonstrations* requiring over 50 demonstrations Fig. 5d). This is a positive outcome of using priors that are based on the dataset, which in turn allows faster inference on the majority of the data.

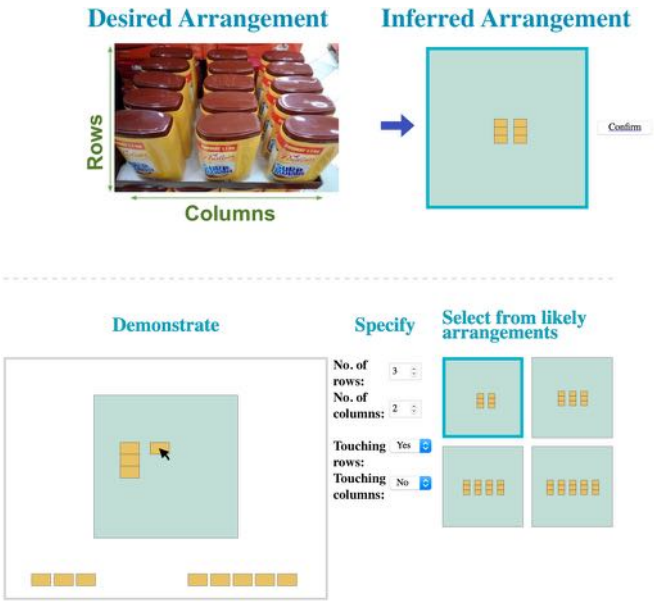

Fig. 7. Simplified user-interface created to evaluate our goal inference model in an online user study. The top part shows the desired configuration with a picture and the visualization of the current shelf arrangement inference. The bottom part has three parts corresponding to three types of programming actions the user can take: demonstration (drag-and-drop items onto shelf), specification (select parameter values from drop-down menus), and selection (choose one of the top four most likely arrangements by clicking on it).

C. User Study Evaluation

Next we conducted a user study on Amazon Mechanical Turk (AMT) to evaluate the goal inference method and investigate teaching strategies preferred by human users. For this, we created a graphical interface for a simplified 2D arrangement task domain, where users had to instruct the robot how to arrange grocery store items by row and column. We were interested in measuring how quickly the desired goal configuration could be inferred from user inputs, what programming actions were preferred, and how they performed in comparison to the teaching strategies defined in Sec. IV-A.

1) Protocol: First users were shown a brief instruction video on how to perform the different programming actions. Then the user moved onto programming desired arrangements, which were communicated to them through a grocery store picture. The user interface, shown in Fig. 7 had three parts for performing the three programming actions. Demonstrations were performed by dragging and dropping items onto a shelf area, which automatically updated the specification fields with the inferred values of task parameters. Specifications were done through drop down menus. The visualization of the most likely top four arrangements was updated after every user action. The user could select one of these by clicking on it. Participants were told to confirm the inferred arrangement once it matched the desired arrangement. Each user completed one practice task and 8 arrangement tasks in a randomized order. After the final task, participants were asked to fill out a questionnaire to provide further insight into their preferred actions and strategies.

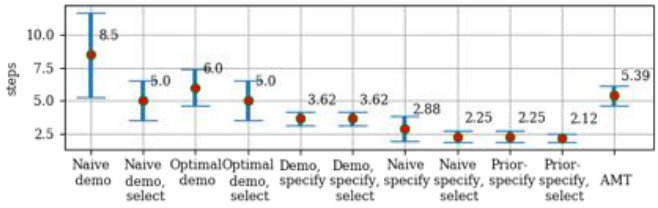

Fig. 8. Average number of steps required for 8 arrangement tasks chosen for the user study, by teaching strategy and human performance (AMT).

2) Metrics: We recorded the user's interactions with the interface, when they performed a demonstration, when they changed the value of a specification field, when they selected one of the proposed arrangements, and when they confirmed the inferred arrangement. We used these recordings to measure the user's preferred action and strategy, the number of steps and time spent per task, and if the inferred arrangement was correct. In the survey at the end, we asked users about their most and least preferred teacher actions, to explain their preference and strategies, and to rate the usefulness of the proposed likely arrangements on a 4-point Likert scale.

3) Results: The study included 32 participants (21M, 11F) with an average age of 32.5 years (SD=9.6). Users took an average of 26.5 seconds (SD=7.8) to complete a task.

a) User performance: Participants correctly completed an average of 5.6 tasks (SD=1.8), 1.8 tasks (SD=1.6) with correct number of rows and columns but wrong distances, and 0.7 (SD=0.9) incorrectly. The wrong distances were likely caused by the angle of the example pictures, as some users did not consider objects as touching, *e.g.,* when they were cone-shaped. Other mistakes were likely due to mis-counting rows or columns. Fig. 8 shows the average number of steps taken by participants to complete a task in comparison to the teaching strategies defined in Sec. IV-A. Overall users were slightly more efficient than the optimal demonstration strategy, but worse than most other strategies.

b) User strategies: Fig. 9 shows the strategies employed by participants across the different shelf configurations they programmed. We see that using specifications was most common, followed by two strategies that combined specifications with the other actions. Most users (71.9%) stated their most preferred action to be *specify* and the least preferred to be *demonstrate*, with 16 (50%) out of 32 users explaining that they found specification to be "easier". The majority (84.4%) of the users found the top 4 proposed arrangements to be useful, but the select action was not used as much as specify. Most users (28.1%) who described their teaching strategy stated *specify & select*; *e.g.,* "type in the rows and columns and then pick the picture that was best". Few users (9.4%) stated their strategy to be a combination of the three actions as shown in the tutorial video. One user mentioned a prior-aware strategy: "I liked to specify exactly what I wanted, but I also could have adjusted the proposed configurations a little bit".

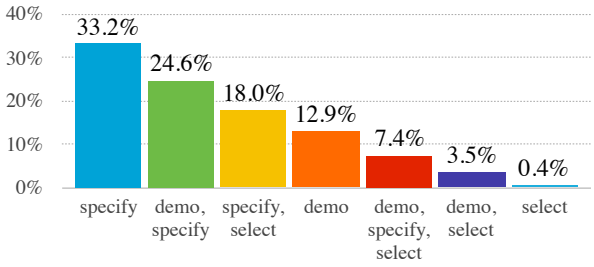

Fig. 9. Distribution of user strategies employed in the AMT study

TABLE II
BENCHMARK TASKS USED FOR EVALUATING THE FULL SYSTEM IMPLEMENTATION ON THE FETCH ROBOT.

#	Grid (n,m,s)	Used manipulation actions	Product
1	(4,1,1) not-touching	Pick and place	Tin cans
2	(2,2,2) not-touching	Pick and place	Tin cans
3	(1,4,1) touching	Pick, place, push left	Tin cans
4	(1,3,1) close distance	Pick, place, push left	Cereal boxes
5	(1,3,1) close distance	Pick, place, push left	Toothpaste
6	(3,3,1) off-grid	Pick, place, push left&front	Tin cans
7	(3,3,1) off-grid	Pick, place, push left&front	Soda cans
8	(2,3,1) interleaving	Pick and place from top	Candy bags

V. SYSTEM EVALUATION

To demonstrate our full system's ability to learn and execute shelf arrangement tasks, we defined a list of benchmark tasks described in Table II. The tasks varied in product type and grid configuration parameters, some of which required different manipulation actions (*e.g.,* to place an objects in touching configurations while avoiding collisions of the gripper with other objects). We also included more complex arrangements such as off-grid cylinders and interleaved soft packaging (Fig. 3(b-c)) which were excluded from our goal inference analysis due to low frequency in the dataset.

A. Protocol

All tasks were programmed by one of the authors, with a subset programmed by another author to ensure consistency. We programmed each arrangement task as efficiently as possible using the teaching strategies from Sec. IV-A and reusing previously taught actions. For each task, the experimenter decided what type of manipulation action was required depending on the object type and the grid configuration. The experimenter could also decide if a new action needed to be demonstrated or if a previously existing one could be reused (as described in Sec. III-F). The actions were chosen to be as simple as possible for the given task, omitting any unnecessary gestures or movements (*e.g.,* if a simple place action was sufficient, no additional push action should be taught). When programming an action, the experimenter could let the robot execute the partial arrangement task for a subset of objects as part of the programming process to mitigate errors in the final task execution.

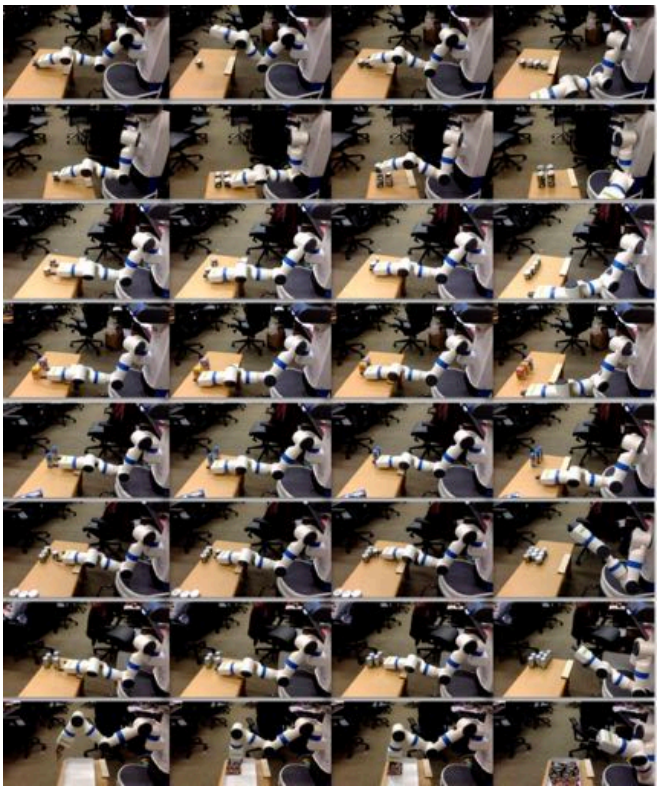

Fig. 10. Snapshots from the executions of the eight system evaluation benchmark tasks (Table II).

B. Results

We programmed four different manipulation actions to complete the eight benchmark tasks. Manipulation actions included pick, place (from front), place from top, push left, and push left&front. We were able to reuse these actions for similar tasks by modifying the required steps of the program. Each task was executed successfully from start to end at least two times. Fig. 10 shows snapshots from the action executions of the benchmark tasks. The evaluation demonstrates our system's capability to learn manipulation actions for the main product types, as well as soft packaging (*e.g.,* candy), which cover 98.7% of the Freiburg dataset[3].

VI. DISCUSSIONS

In the AMT study, users performed demonstrations using drag-and-drop operations which is simpler than stacking objects on a shelf with a real robot. As users preferred specification strategies over drag-and-drop demonstrations, it is likely that demonstrating object stacking would be less preferred as well. Some limitations of our approach are as follows.

1) Our system only considers one demonstrated manipulation action, even if multiple demonstrations are performed. An extension could consider the poses of multiple action demonstrations and infer an action

[3]Sample executions of shelf arrangement tasks can be seen at https://youtu.be/liaSirH0CeM

adapted to a specific scenario (*e.g.,* only use push action if the gripper would collide with other landmarks).
2) The system's perception system is limited as it does not recognize separate objects that are too close together. This limits our ability to detect demonstrated configurations for objects that are meant to be touching.
3) We did not consider the orientation of the object (*e.g.,* product label facing front) as commonly applied in shelf organization tasks. Our perception system only recognizes the object's bounding box, so an extension could include better object recognition and object orientations.

VII. CONCLUSIONS

We present an end-user robot programming system to efficiently teach a robot shelf arrangement tasks and actions. We propose augmenting Programming by Demonstration with domain-aware goal inference and direct user inputs to accelerate the teaching process. Our task goal representation is based on an analysis of a large database of grocery store shelf images. We evaluated our goal inference with different teaching strategies and with real human teachers. We then demonstrated our system's ability to efficiently learn and perform various shelf arrangement tasks and actions on a Fetch mobile manipulator.

ACKNOWLEDGMENT

We would like to thank Justin Huang for providing the open-source system Rapid PbD and his continuous support. This work was supported by IDEX Université Grenoble Alpes, CNRS (PEPS), and the National Science Foundation, grant #IIS-1552427 "CAREER: End-User Programming of General-Purpose Robots".

REFERENCES

[1] P. Jund, N. Abdo, A. Eitel, and W. Burgard, "The freiburg groceries dataset," *arXiv preprint arXiv:1611.05799*, 2016.

[2] J. Huang, T. Lau, and M. Cakmak, "Design and evaluation of a rapid programming system for service robots," in *Intl. Conf. on Human Robot Interaction*. IEEE Press, 2016, pp. 295–302.

[3] E. I. Barakova, J. C. Gillesen, B. E. Huskens, and T. Lourens, "End-user programming architecture facilitates the uptake of robots in social therapies," *Robotics and Autonomous Systems*, vol. 61, no. 7, pp. 704–713, 2013.

[4] D. Glas, S. Satake, T. Kanda, and N. Hagita, "An interaction design framework for social robots," in *Robotics: Science and Systems*, vol. 7, 2012, p. 89.

[5] D. F. Glas, T. Kanda, and H. Ishiguro, "Human-robot interaction design using Interaction Composer: Eight years of lessons learned," in *Intl. Conf. on Human Robot Interaction*. IEEE Press, 2016, pp. 303–310.

[6] M. Stenmark, M. Haage, and E. A. Topp, "Simplified programming of re-usable skills on a safe industrial robot: Prototype and evaluation," in *Intl. Conf. on Human-Robot Interaction*. ACM, 2017, pp. 463–472.

[7] S. Alexandrova, Z. Tatlock, and M. Cakmak, "Roboflow: A flow-based visual programming language for mobile manipulation tasks," in *Intl. Conf. on Robotics and Automation (ICRA)*. IEEE, 2015, pp. 5537–5544.

[8] J. Huang and M. Cakmak, "Code3: A system for end-to-end programming of mobile manipulator robots for novices and experts," in *Intl. Conf. on Human-Robot Interaction*. ACM, 2017, pp. 453–462.

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

[10] 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.

[11] S. Schaal, J. Peters, J. Nakanishi, and A. Ijspeert., "Learning movement primitives," in *International Symposium on Robotics Research (ISRR)*, 2003.

[12] S. Calinon and A. Billard, "Statistical learning by imitation of competing constraints in joint and task space," *Advanced Robotics*, vol. 23, no. 15, pp. 2059–2076, 2009.

[13] B. Akgun, M. Cakmak, K. Jiang, and A. Thomaz, "Keyframe-based learning from demonstration," *(In press) Journal of Social Robotics, Special issue on Learning from Demonstration*, 2012.

[14] J. Schulman, J. Ho, C. Lee, and P. Abbeel, "Learning from demonstrations through the use of non-rigid registration," *International Journal of Robotics Research*, 2013.

[15] A. Mohseni-Kabir, C. Rich, S. Chernova, C. L. Sidner, and D. Miller, "Interactive hierarchical task learning from a single demonstration," in *Intl. Conf. on Human-Robot Interaction*. ACM, 2015, pp. 205–212.

[16] M. Pardowitz, S. Knoop, R. Dillmann, and R. Zollner, "Incremental learning of tasks from user demonstrations, past experiences, and vocal comments," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, vol. 37, no. 2, pp. 322–332, 2007.

[17] 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.

[18] S. Niekum, S. Chitta, A. Barto, B. Marthi, and S. Osentoski, "Incremental semantically grounded learning from demonstration." in *Robotics: Science and Systems*, vol. 9, 2013.

[19] B. Jansen and T. Belpaeme, "A model for inferring the intention in imitation tasks," in *Robot and Human Interactive Communication, 2006. ROMAN 2006. The 15th IEEE International Symposium on*. IEEE, 2006, pp. 238–243.

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

[21] N. Correll, K. E. Bekris, D. Berenson, O. Brock, A. Causo, K. Hauser, K. Okada, A. Rodriguez, J. M. Romano, and P. R. Wurman, "Analysis and observations from the first amazon picking challenge," *IEEE Transactions on Automation Science and Engineering*, 2016.

[22] N. Abdo, C. Stachniss, L. Spinello, and W. Burgard, "Robot, organize my shelves! tidying up objects by predicting user preferences," in *Robotics and Automation (ICRA), 2015 IEEE Intl. Conf. on*. IEEE, 2015, pp. 1557–1564.

[23] A. Kurenkov, B. Akgun, and A. L. Thomaz, "An evaluation of gui and kinesthetic teaching methods for constrained-keyframe skills," in *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, 2015.

[24] H. Nguyen, M. Ciocarlie, K. Hsiao, and C. C. Kemp, "Ros commander (rosco): Behavior creation for home robots," in *2013 IEEE Intl. Conf. on Robotics and Automation (ICRA)*. IEEE, 2013, pp. 467–474.

[25] M. R. Dogar and S. S. Srinivasa, "Push-grasping with dexterous hands: Mechanics and a method," in *Intl. Conf. on Intelligent Robots and Systems (IROS)*. IEEE, 2010, pp. 2123–2130.

[26] J. E. King, J. A. Haustein, S. S. Srinivasa, and T. Asfour, "Nonprehensile whole arm rearrangement planning on physics manifolds," in *Intl. Conf. on Robotics and Automation (ICRA)*. IEEE, 2015, pp. 2508–2515.

[27] 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.

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

[29] S. Niekum, S. Osentoski, G. Konidaris, and A. G. Barto, "Learning and generalization of complex tasks from unstructured demonstrations," in *Intl. Conf. on Intelligent Robots and Systems (IROS)*. IEEE, 2012, pp. 5239–5246.

[30] M. Cakmak and M. Lopes, "Algorithmic and human teaching of sequential decision tasks," in *Association for the Advancement of Artificial Intelligence (AAAI)*, 2012.

[31] F. Khan, B. Mutlu, and X. Zhu, "How do humans teach: On curriculum learning and teaching dimension," in *Advances in Neural Information Processing Systems*, 2011, pp. 1449–1457.