Learning to Use a Ratchet by Modeling Spatial Relations in Demonstrations

Li Yang Ku*, Scott Jordan*, Julia Badger†, Erik G. Learned-Miller*, and Roderic A. Grupen*
*College of Information and Computer Sciences
University of Massachusetts Amherst, Amherst, MA
Email: {lku, sjordan, elm, grupen}@cs.umass.edu
†National Aeronautics and Space Administration, Houston, TX
Email: julia.m.badger@nasa.gov

***Abstract*—We introduce a framework where visual features describing the interaction between robot hand, a tool, and an assembly fixture can be learned efficiently using a small number of demonstrations. We demonstrate the approach by torquing a bolt with Robonaut-2 using a hand held ratchet. The difficulty lies in the uncertainty of the ratchet pose after grasping and the high precision required for mating the socket to the bolt and replacing the tool in the tool holder. The approach learns the desired relative position between visual features on the ratchet and the bolt. It does this by identifying goal offsets from visual features that are consistently observable over a set of demonstrations. With this approach we show that Robonaut-2 is capable of grasping the ratchet, tightening a bolt, and putting the ratchet back into a tool holder. We measure the accuracy of the socket-bolt mating subtask over multiple demonstrations and show that a small set of demonstrations can decrease the error significantly.**

I. Introduction

Learning from demonstration (LfD) is an appealing approach to programming robots because it is similar to how humans teach each other. However, most work on LfD has focused on learning the demonstrated motion, action constraints, and/or trajectory segments and has assumed that object labels and poses can be identified correctly. This assumption may be warranted in well-structured industrial settings, but does not hold, in general, for the kinds of uncertainty and variability common in everyday human environments.

We present an integrated approach that treats identifying informative sensory features as part of the learning process. This gives the robot the capacity to manipulate objects without fiducial markers and to learn actions focused on salient parts of the object. Instead of defining actions as relative movements with respect to the object pose, our actions are based on features that represent meaningful sensory milestones. In this paper, features correspond to sensory patterns that can be localized in the robot environment. With additional guidance provided by the operator, the informative features specific to an object instance can be identified automatically. The goal is to have the robot interact with the same objects that appear in the demonstrations.

We show that a challenging tool use task—tightening a bolt using a ratchet—can be learned from a small set of demonstrations. A different in-hand ratchet pose may result in failure on mating the socket to the bolt if the robot only considers the pose of the hand and the bolt. The proposed approach learns what part of the ratchet should be aligned with the bolt by recognizing consistent spatial relations between features among a set of demonstrations.

Two major contributions in this work are as follows.

1) Action demonstrations are classified into three different types based on the interaction between visual features and robot end effectors. This allows robots to repeat tool usage demonstrations by modeling the spatial relations between visual features from the tool and the workpiece.
2) An approach that distills multiple demonstrations of the same action to produce more accurate actions by identifying spatial relations that are consistent across demonstrations.

II. Related Work

Much research has focused on methods for "learning from demonstration (LfD)," in which robots acquire approximate programs for replicating solutions to sensory and motor tasks from a set of human demonstrations. In work by Calinon et al. [5] [4], Gaussian mixture models are used to model multiple demonstrated trajectories by clustering segments based on means and variances. A Gaussian mixture regression is used to generate motions for different start and goal states during execution. In work by Pastor et al. [13], dynamic movement primitives are used to generalize trajectories with different start and end point. Instead of modeling trajectories in terms of motion invariants, our work focuses on learning consistent perceptual feedback that provides informative guidance for situated actions.

Approaches that learn from multiple demonstrations often require an experienced user to show a variety of trajectories in order to estimate task information like state variables, task constraints, relative frame, *etc.* In work by Alexandrova et al. [2], instead of generalizing from multiple examples, the user demonstrates once and provides additional task and feature information via a user interface. The approach in this paper is similar, the user specifies action types and the informative features are identified automatically.

In the work by Phillips et al. [15], experience graphs are built from demonstration and used to speed up motion planning. A manipulation task such as approaching a door

and opening can be planned in a single stage by adding an additional dimension that represents the state of the object. However, the demonstrated tasks are restricted to cases where the object can be manipulated in a one dimensional manifold that is detectable based on the position of a single contact point. In this work, demonstrations are stored as aspect transition graphs (ATGs). ATGs are directed multi-graphs composed of aspect nodes that represent observations and edges that represent action transitions. Aspect nodes represent observations directly and can, therefore, be used to model a higher dimensional space.

ATGs were first introduced by Sen [17] as an object representation that models how different control programs can lead from one aspect to another. Ku et al. [9] [10] further applied belief space planning on ATGs and show that ATG models learned from demonstration can be used to plan sequences of actions to compensate for the robot's reachability constraint. In this work, we extend the ATG representation to model interaction between objects and demonstrate that by distilling multiple ATGs learned from demonstrations the accuracy of actions can increase significantly.

In the work by Akgun et al. [1], a demonstrator provides a sparse set of consecutive keyframes that summarizes trajectory demonstrations. Pérez-D'Arpino and Shah [14] also introduced C-Learn, a method that learns multi-step manipulation tasks from demonstrations as a sequence of keyframes and a set of geometric constraints. In our work, aspect nodes that contain informative perceptual feedback play a similar role as keyframes that guide the multi-step manipulation. Instead of considering geometric constraints between an object frame and the end effector frame, relations between visual features and multiple robot frames are modeled.

There has been a lot of work on developing visual descriptors that are robust to viewpoint variations [12] [3] [21]. Recently, several papers have investigated learning image descriptions using Convolutional Neural Networks (CNNs) [7] [18] [11]. The hierarchical CNN features [11] were shown to represent parts of an object that are informative for manipulation. We construct aspects and actions using hierarchical CNN features that are consistent among multiple examples.

III. Approach

In this section, we describe our approach by teaching the robot to use a ratchet from demonstrations. First, we describe the aspect transition graph (ATG) that is used to model demonstrations. Second, we explain how the user demonstrates each part of the task and provides additional information for building ATG models. Third, we illustrate how multiple ATGs created from demonstrations are merged to create more robust models. Finally, we demonstrate how a set of ATG models are used during execution.

A. Aspect Transition Graph Model

In previous work, aspect transition graphs (ATG) created from demonstrations were used to represent how actions lead from one observation to another using a directed multi-graph [10]. In this work, we extend the ATG representation to include actions that support interactions between objects and add force feedback to the aspect representation. In the following, we describe the definition of an ATG, the visual and force features, the aspect representation, and the actions.

1) Definition: An aspect transition graph (ATG) is a directed *multigraph* $G = (\mathcal{X}, \mathcal{U})$, composed of a set of aspect nodes $\mathcal{X}$ connected by a set of action edges $\mathcal{U}$ that capture the probabilistic transition between aspect nodes. We define an aspect as a multi-feature observation that is stored in the model. An action edge U is a triple (x_1, x_2, a) consisting of a source node x_1, a destination node x_2 and an action a that transitions between them.

2) Visual Features: Aspect nodes in an ATG include visual features that participate in measurements of the similarity between aspect nodes and observations. Their positions are also used as references for end effector positions in actions edges. The hierarchical CNN features introduced by Ku et al. [11] exploit the fact that CNNs are by nature hierarchical; a filter in a higher layer with coarse location information is a combination of lower level features with higher spatial accuracy. Instead of representing a feature with a single filter in a certain CNN layer, hierarchical CNN features use a tuple of filter indices to represent a feature such as (f_i^5, f_j^4, f_k^3), where f_i^5, f_j^4, and f_k^3 represent the i^{th} filter in the 5^{th} convolutional layer, the j^{th} filter in the 4^{th} convolutional layer, and the k^{th} filter in the 3^{rd} convolutional layer respectively. A lower layer CNN filter often represents the local structure of a more complicated structure represented by a higher layer filter. The hierarchical CNN feature identifies local structures at each level that are related to a hierarchical "part-based" representation of an object, each part of which affords opportunities for control and interaction. In this paper, the layer of a hierarchical CNN feature refers to the lowest layer in the filter tuple. A variant of hierarchical CNN features that are localized through guided backpropagation are used as visual features in this work [19].

3) Force Features: The force features are based on load cells in Robonaut-2's forearms. We collect the force values and project them to the body frame at 10 Hz. The values are then averaged over the preceding one-second interval. This force feature is concatenated with visual features and proprioceptive feedback to represent aspect nodes.

4) Aspect Representation: In previous work, Ku et al. [10] introduced an aspect representation based on visual and proprioceptive feedback. The representation uses a set of visual feature responses and their spatial relations with robot end effectors to represent an aspect node. In this work, we extend this aspect representation to include force features and handle multiple object interactions.

In previous work, a single object is considered per ATG model. In this work, we consider multiple objects and their interactions. An aspect node can be used to represent a particular "view" of an object or a distinctive interaction between adjacent objects. For example, two disjoint feature clusters generated by two objects are modeled by two aspect

nodes, each representing how the robot perceives them. In contrast, a single feature cluster can span two (partially) assembled objects to focus on object-object interactions. The ATG representation can therefore model object interactions that result in transitions between these two types of aspect nodes.

5) Actions: An action is represented using a controller in the control basis framework [8] and is written in the form $\phi|_{\tau}^{\sigma}$, where ϕ is a potential function that describes the error between the current and target robot configuration, σ represents sensory resources allocated, and τ represents the motor resources allocated. The potential functions are formulated as $\phi_V = \sum_{v \in V}(v - g_v)^2$, where v and g_v are visual features and goal locations for these features ($v, g_v \in \mathbb{R}^3$) and $\phi_R = \sum_{r \in R}(r - g_r)^2$, where r and g_r are robot frames and goals for these frames ($r, g_r \in SE(3)$).

Demonstrated actions are distinguished into three types:

1) *robot-visual actions* $a_{RV} = \phi_{\mathrm{R}}|_{\tau}^{\sigma_{\mathrm{V}}}$
2) *robot-proprioceptive actions* $a_{RP} = \phi_{\mathrm{R}}|_{\tau}^{\sigma_{\mathrm{P}}}$
3) *visual-visual actions* $a_{VV} = \phi_{\mathrm{V}}|_{\tau}^{\sigma_{\mathrm{V}'}}$

Parameters σ_V and σ_P are the sensory resources containing a set of observed visual features V and a set of robot frames P based on proprioceptive feedback, respectively. Potential functions ϕ_R and ϕ_V have a minimum when a set of robot frames R and a set of visual features V that are controllable by the robot matches a set of corresponding goals G calculated based on offsets to σ. We give examples of these three types of demonstrations in the following.

a) The *robot-visual action* (a_{RV}) specifies the target pose of a set of robot frames with respect to a set of visual feature locations. The left and right image in Figure 1 shows the result of executing an a_{RV} action where the goal is to reach the ratchet pregrasp pose. The yellow and cyan dots represent 5th and 4th layer hierarchical CNN features and the red and green circles represent the minima of potential functions for the hand and fingers. The arrows represent the offset from features used as references to construct potential functions and the red and green ellipses represent the contour lines of the potential functions for the hand and index finger. This type of action was used in previous work [10] for tool grasping.

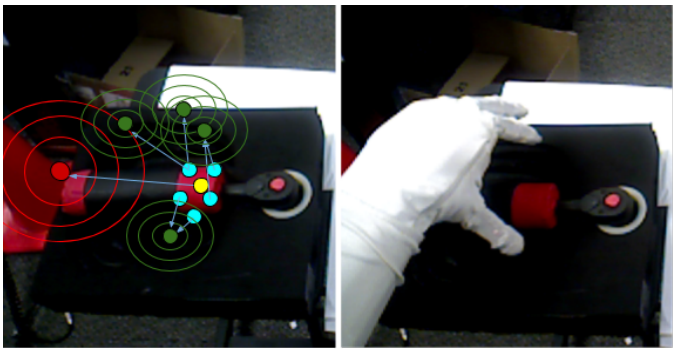

Fig. 1. Example of a *robot-visual action* (a_{RV}) that reaches the ratchet pregrasp pose.

b) The *robot-proprioceptive action* (a_{RP}) specifies the target pose of a set of robot frames with respect to a set of current robot frames based on proprioceptive feedback. The left and right image in Figure 2 shows the result of executing an a_{RP} action where the goal is to move the hand relative to the current hand frame so that the grasped ratchet is extracted from the tool holder. The yellow ellipse is the current hand pose and the arrow indicates the reference offset derived from demonstration. The red ellipses represent the contour lines of the potential functions for the hand.

Fig. 2. Example of a *robot-proprioceptive action* (a_{RP}) that extracts the ratchet.

c) The *visual-visual action* (a_{VV}) specifies the goal position of a set of controllable visual features relative to another set of visual features on a different object. The left and right image in Figure 3 shows the result of executing an a_{VV} action where the goal is to place the socket on top of the bolt. The purple dots are features on the bolt used as references for constructing the potential function and the orange dot is the feature on the socket controlled by the potential function. The blue dots are goal positions generated based on relative positions to features indicated by the black arrows. The red dotted arrow shows a path for the feature to reach the minimum of the potential function represented by the blue ellipse contours. After a single visual-visual action, visual features on the grasped object may fail to reach the goal location due to movement error, change in object in-hand pose, or imperfect camera calibration. To tackle this problem, the same action is executed multiple times with updated visual feature locations on the grasped object until convergence. Unlike *robot-visual actions*, modeling spatial relations between visual features allows situations where the in-hand ratchet poses are different to have the same intended action outcome.

The detected locations of visual features and robot frames are inevitably influenced by noise in the system that may be caused by imperfect sensors or changes in the environment. This makes tasks that require high precision challenging. To accommodate this problem we assume that the references for motor resources τ is generated by adding zero mean noise $N(0, \Sigma)$ to the original reference. By sampling from this distribution during execution, the controller superimposes an additive zero mean search to the motion. Such movement increases the tolerance of the insertion task to uncertainty.

Figure 4 shows the sensorimotor architecture that drives

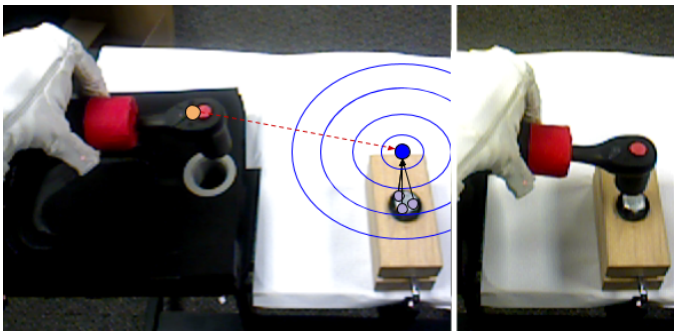

Fig. 3. Example of a *visual-visual action* (a_{VV}) that places the ratchet on top of the bolt.

transitions in the ATG model. The perceptual feedback is used to represent aspect nodes and actions are executed based on these sensory resources defined in action edges.

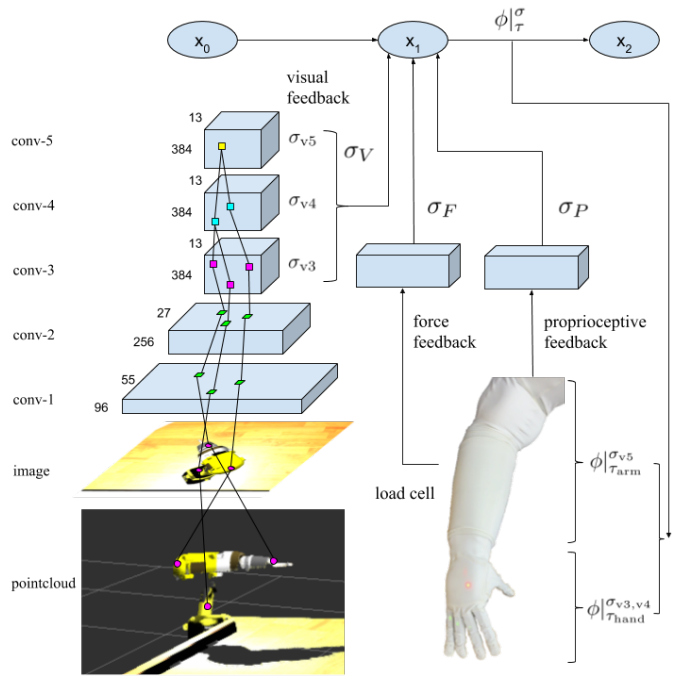

Fig. 4. The sensorimotor architecture driving transitions in the ATG framework. The sensory resources σ_{F} that represent a set of features based on visual and force feedback and σ_{P} that represents a set of robot frames based on proprioceptive feedback are used to parameterize actions $\phi|_\tau^\sigma$. Here ϕ is a potential function that describes the error between the current and target robot configuration and τ represents the motor resources allocated. In this example, the 5th layer hierarchical CNN features σ_{v5} are used to control the arm motors τ_{arm} and the 3rd and 4th layer hierarchical CNN features $\sigma_{\mathrm{v3,v4}}$ are used to control the hand motors τ_{hand}.

B. Building Models From Demonstrations

Each demonstration coupled with information provided by the operator is used to create an ATG model. During execution, this set of ATG models is used to determine the next action based on the current observation and the given target. We describe the user interface and how ATGs are created from demonstration in the following.

1) User Interface: Demonstrated tasks are performed using a teleoperator implemented in the MoveIt! platform [20], in which the user can drag interactive markers in a graphical interface to move the robot end effector or change the robot hand configuration. Similar to the keyframe demonstration approach [1] [14], users indicate intermediate steps for each demonstration required for creating the ATG model. After the robot reaches an intermediate step, the interface asks the user to provide the type of demonstration listed in Section III-A5 that the user performed.

For the *robot-proprioceptive action* $a_{RP} = \phi_{\mathrm{R}}|_\tau^{\sigma_{\mathrm{P}}}$, the user can select either the end effector frame or the body frame as the proprioceptive sensor resource σ_P. The user also has the option to add a structural search movement described in Section III-A5 to the demonstrated action.

2) Creating ATG models: During the demonstration, an aspect node is created for each observed feature cluster at each intermediate step. A feature cluster can be a single object or multiple objects in contact based on the Euclidean cluster extraction algorithm in the point cloud library [16]. Based on the demonstration type selected by the user, the system connects new aspect nodes x_t to aspect nodes x_{t-1} created at the previous time step with action edges a_{t-1} that model the demonstrated action.

For the *robot-visual action* $a_{RV} = \phi_{\mathrm{R}}|_\tau^{\sigma_{\mathrm{V}}}$, the relative poses between a set of visual features V and a set of robot frames R are recorded in the action edge. This set of visual features is selected based on the feature's proximity to the robot end effector after action execution. In this work, hierarchical CNN features mentioned in section III-A2 are used as visual features. Five 5th layer hierarchical CNN features that are closest to each hand frame and eight 4th layer hierarchical CNN features that are closest to each finger tip frame are selected. For example, the action that moves the robot hand to a pregrasp pose for grasping the ratchet will use features such as the corner of the handle or the neck of the ratchet that are close to the fingers as references for placing the fingers relative to the ratchet. The aspect nodes connected by this action is identified based on their proximity to the active robot end effector.

For the *robot-proprioceptive action* $a_{RP} = \phi_{\mathrm{R}}|_\tau^{\sigma_{\mathrm{P}}}$, the relative poses between the set of robot frames R and the set of reference robot frame P are recorded in the action edge. For example, the action that lifts the ratchet up after grasping it is modeled by moving the hand frame relative to the current hand frame. The aspect nodes connected by this action is identified based on their visual similarity to the last connected aspect node.

For the *visual-visual action* $a_{VV} = \phi_{\mathrm{V}}|_\tau^{\sigma_{\mathrm{V}'}}$, the relative poses between a set of visual features V on the tool grasped by the robot and a set of visual features V' on the target object interacting with the tool is recorded in the action edge. The set of visual features on the tool is selected based on the feature's stability with respect to movement under the assumption that the grasped object is rigid. This is determined by the position differences of the features in the hand frame before and after

the action. The set of visual features on the target object is then selected based on the feature's distance to the selected features on the tool after the action. This *visual-visual action* is represented by two action edges that connect the aspect nodes that represent the tool and the target object to the aspect node that represents the interaction. These aspect nodes can be identified base on their relative distance and proximity to the active end effector.

This sequence of aspect nodes connected by action edges become the ATG model that represents the demonstration. Aspect nodes that are created from other feature clusters that are not chosen are grouped into a background ATG that is used to recognize feature clusters that are not targets for manipulation. For example, a demonstration that only grasps the ratchet does not care about the bolt platform. A background ATG that represents the bolt is therefore used to match to the feature cluster of the bolt during execution.

C. Learning from Multiple Demonstrations

With a single demonstration, there remain ambiguities regarding the goal. For example, in the action that puts the socket on top of the bolt, it is ambiguous whether the demonstration intends to convey a spatial relationship between the socket and the bolt or some other part of the ratchet and the bolt. With multiple demonstrations, this ambiguity may be resolved by observing consistent relations between features. In this section, we describe how to take multiple demonstrations of the same task to create more robust ATG models. We call these ATGs created from multiple demonstrations *distilled* ATGs.

1) Identifying Common Features: A set of features are stored in the aspect node to represent the observation of an aspect. Correctly associating the current observations with a memorized aspect node is crucial for implementing transitions to goal status. However, not all features provide the same amount of information. Moreover, some features are more sensitive to lighting changes and some may belong to parts of the visual cluster that may change appearance across examples. With a single demonstration, these kinds of features may be indistinguishable. With multiple demonstrations, common features can be identified by estimating the feature variance across demonstrations.

Given demonstrations of the same task with the same sequence of intermediate steps, our approach looks for features that are consistent across multiple demonstrations. For the observations at each intermediate step, the N most consistent features are chosen. The consistency score is defined as $S_c = n_f / std(f)$, where n_f is the number of times feature f appears among the matched intermediate steps and $std(f)$ is the standard deviation of the value of feature f. We score visual features, proprioceptive features, and force features together with weights of $1, 1, 0.001$, respectively.

2) Recognizing Consistent Actions: For action edges that represent a *robot-visual action* a_{RV} or a *visual-visual action* a_{VV} in an ATG model, the action reference is specified in terms of a subset of features stored in the aspect node. As result of a single demonstration, features are chosen based on their proximity to robot frames or features controllable by the robot. With multiple demonstrations, a more robust set of features can be identified and used to define the aspect.

For the *robot-visual action* $a_{RV} = \phi_{\mathrm{R}}|_{\tau}^{\sigma_{\mathrm{V}}}$, the top N pairs of robot frames $r \in R$ and visual features $v \in V$ that have the lowest variances in XYZ position offsets are chosen to represent the action. For example, when learning from multiple demonstrations of the action that grasps the ratchet, this approach concludes that features on the ratchet are more reliable than features on the tool holder since the ratchet may be placed at different positions in the tool holder across demonstrations.

For the *visual-visual action* $a_{VV} = \phi_{\mathrm{V}}|_{\tau}^{\sigma_{\mathrm{V}'}}$, the top N pairs of visual features in the tool aspect node $v \in V$ and the target object aspect node $v' \in V'$ that have the lowest variance $var(v, v')$ is selected. $var(v, v')$ is the variance of the XYZ position offsets between feature v and feature v' after the action across demonstrations. For example, the action that places the socket of the ratchet on top of the bolt determines that a consistent spatial relation exists between the features on the socket and those on the bolt after executing the action. Figure 5 shows the top feature pairs identified for constructing a *visual-visual action* from demonstrations. The robot is able to comprehend that the head of the ratchet should be aligned with the bolt autonomously.

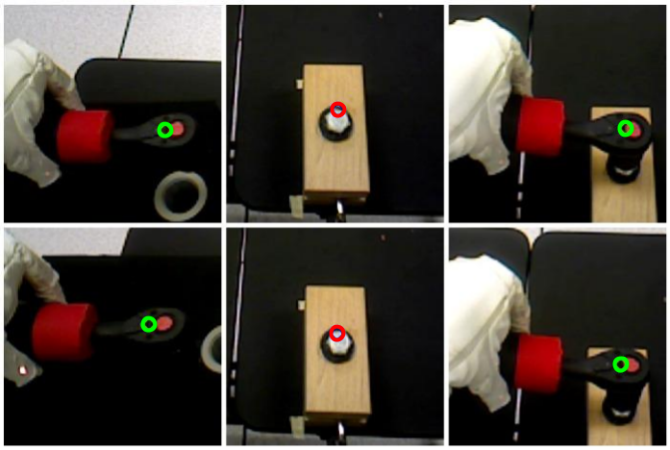

Fig. 5. Identifying informative features from multiple demonstrations. The two rows represent two demonstrations that place the socket of the ratchet on top of the bolt. The columns from left to right show the aspect nodes representing the tool, the target object, and the interaction for this *visual-visual action* $a_{VV} = \phi_{\mathrm{V}}|_{\tau}^{\sigma_{\mathrm{V}'}}$. The green circles in the tool and interaction aspect nodes represent the top visual feature $v \in V$ used to reach the minimum of the potential function ϕ_V while the red circles in the target object aspect node represent corresponding features $v' \in V'$ that are used as references.

To confirm that the selected visual features represent meaningful parts of an object, we visualize the feature identified on the ratchet head in Figure 5 using the visualization tool introduced by Yosinski et al. [22]. Figure 6 shows the top 9 images that the filters f_{23}^5, f_{60}^4, and f_{184}^3 have the highest response on among the ImageNet dataset. The feature tuple $(f_{23}^5, f_{60}^4, f_{184}^3)$ can be interpreted as a red region surrounded

by black regions. Although there are no ratchet class trained on the neural network, it reuses similar visual patterns learned among other classes such as bird species. The left image in Figure 7 shows a visualization on what pixels contribute to the feature using guided backpropagation [19]. The background grey corresponds to zero contribution. Notice that the feature is only contributed by the head of the ratchet and represents meaningful parts for modeling the action. The right image is the corresponding input image.

Fig. 6. Visualization of the hierarchical CNN feature $(f^5_{23}, f^4_{60}, f^3_{184})$ that is identified on the ratchet head by showing the top 9 images that have the highest response among ImageNet for filter f^5_{23}, f^4_{60}, and f^3_{184}.

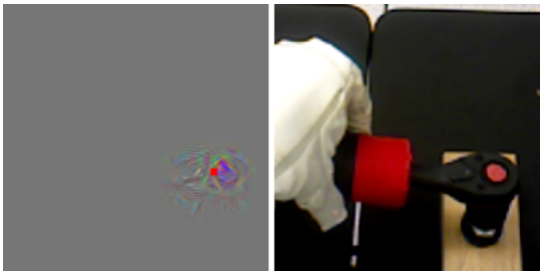

Fig. 7. Visualization on what pixels contribute to the hierarchical CNN feature $(f^5_{23}, f^4_{60}, f^3_{184})$ using guided backpropagation.

D. Planning Actions with ATG models

The set of ATG models created from demonstrations stores observations of each feature cluster in aspect nodes and predicts transitions caused by action edges (Figure 8). During execution, this set of ATG models is used to plan actions to reach a given goal state.

At time step t, for each feature cluster the aspect node x_t in the set of ATGs that has the highest posterior probability $p(x_t|z_t)$ given the current observation z_t of the feature cluster is first identified. The prior probability $p(x_0)$ of being in an aspect node x_0 at time 0 is set to be uniform over all aspect nodes in the set of ATGs unless modified by the user. The probability $p(x_t)$ of being in an aspect node x_t is updated by the Bayes filter algorithm, $p(x_t) = \sum_{x_{t-1}} p(x_t|a_{t-1}, x_{t-1}) \cdot p(x_{t-1})$, where a_{t-1} is the action taken at time step $t-1$. The transition probability $p(x_t|a_{t-1}, x_{t-1})$ is set proportional to $p(a_{t-1}|x_t, x_{t-1})$, which is modeled as a multivariate Gaussian distribution based on the value difference between the parameters of the executed action a_{t-1} and the action edge $\hat{a}$ that connects aspects x_{t-1} and x_t in the ATG model. The maximum a posteriori (MAP) aspect node for each feature

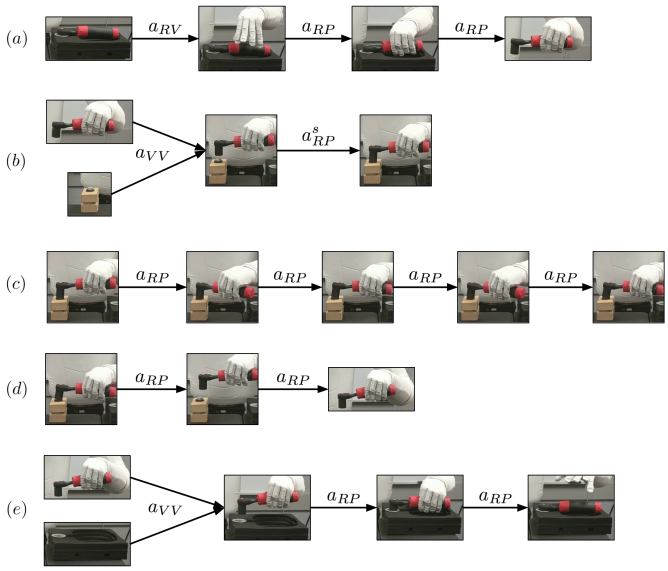

Fig. 8. The visualization of the set of ATGs created from demonstrations for the ratchet task. Each connected ATG represents a sub-task. The images represents aspect nodes and the edges indicate the type of actions used to model transitions.

cluster can therefore be determined by calculating $p(x_t|z_t) = p(z_t|x_t) \cdot p(x_t)$, where $p(z_t|x_t)$ is modeled with generalized Gaussian distributions as introduced in [10].

During execution, the user selects a goal aspect. Based on the MAP aspect node x_t of each feature cluster, the next action is chosen based on the first action edge on the shortest path from the MAP aspect node to the goal aspect node. If the chosen action edge is a *visual-visual action* type, the planner needs to confirm that both the tool aspect node and the target object aspect node is observed. There are two ways to transition between nodes, 1) follow edges learned from demonstrations, or 2) identify equivalent aspect nodes in ATGs and transition between them. If there is no valid path from the current aspect node x_t to the given goal aspect node, the planner guesses possible paths by merging similar aspect nodes from the current ATG to other ATGs until a path exists. The similarity between two aspect node uses the same model as the likelihood function $p(z_t|x_t)$. These two ways of identifying paths in ATGs allow the robot to learn subtasks separately and repeat the full task during execution.

IV. Experiments

In this work, we show that with a small set of demonstrations, Robonaut-2 is capable of performing a ratchet task that involves grasping the ratchet, tightening a bolt, and putting the ratchet back into a tool holder. The complete task sequence can be seen in the supplementary video. We compare the success rate of mating the socket to the bolt as a function of the number of demonstrations using Robonaut-2. To evaluate the accuracy of the position of the socket with respect to the bolt, we further experimented in the Robonaut-2 simulator [6] using up to five demonstrations. The demonstration collection process,

the experimental setting, and the result of the comparison is described in the following.

A. Demonstrations for the Ratchet Task

Instead of demonstrating the entire ratchet task in one session, we segment the task into shorter sequences of sub-tasks that are easier to demonstrate. The ratchet task is segmented into five different subtasks, *a)* grasping the ratchet, *b)* mating socket to the bolt, *c)* tightening the bolt, *d)* removing the socket from the bolt, and *e)* putting the ratchet back into the tool holder. For subtasks *a)*, two demonstrations are provided. For subtask *b)* and *e)* four demonstrations are combined to create the distilled ATG model as described in Section III-C. For subtasks *c)* and *d)*, only one demonstration is performed since the features that support these actions are unambiguous.

Figure 8 shows the ATGs created from these five sub-tasks from top to bottom. The type of demonstrations classified for each action are listed next to the action edges. For example, the ATG created for subtask *a)* (grasping the ratchet) has four different relations between the hand, the ratchet, and the tool holder: the ratchet in the tool holder (no hand), pregrasp, grasped within and without the tool holder. ATG for *b)* shows that in order to execute the *visual-visual action*, both the ratchet-in-hand aspect and the bolt aspect have to exist. The second action edge that mates the socket to the bolt incorporates a structural search motion as well. ATG for subtask *c)* is created from demonstrations of two tightening turns. Each clockwise and counter-clockwise turn is categorized as a type a_2 demonstration that moves relative to the hand frame.

During execution, the aspect where the bolt is tightened is first submitted as a goal aspect to the robot. The planner described in Section III-D identifies the current aspect node and finds a path to reach the goal aspect. Once the robot finishes tightening the bolt, the aspect where the ratchet is put back to the tool holder is set as the goal aspect. Figure 9 shows the sequence of the complete task. With this approach Robonaut-2 is capable of executing the complete ratchet task successfully even when there are small differences in the initial tool, bolt, and tool holder locations.

B. Evaluating Ratchet Task

In this experiment, the robustness of the framework is tested on the ratchet task based on the ATGs created from demonstrations. A total of 22 settings are tested. For each setting, the initial location of the tool holder or bolt platform is altered. For the first 16 settings, the bolt platform is moved away 5 cm from the demonstrated position and the tool holder is placed at 16 different locations on a four by four grid that are 1 cm apart. For the other 6 settings, different bolt platform positions and orientations and one randomly chosen tool holder position on the four by four grid are set. These initial poses are shown in Figure 10. For each different settings, if grasping fails, the robot retries grasping. If mating the socket with the bolt fails, the robot skips tightening and continue. The number of successes for each subtask are shown in table I. Grasping failed twice when the ratchet got stuck

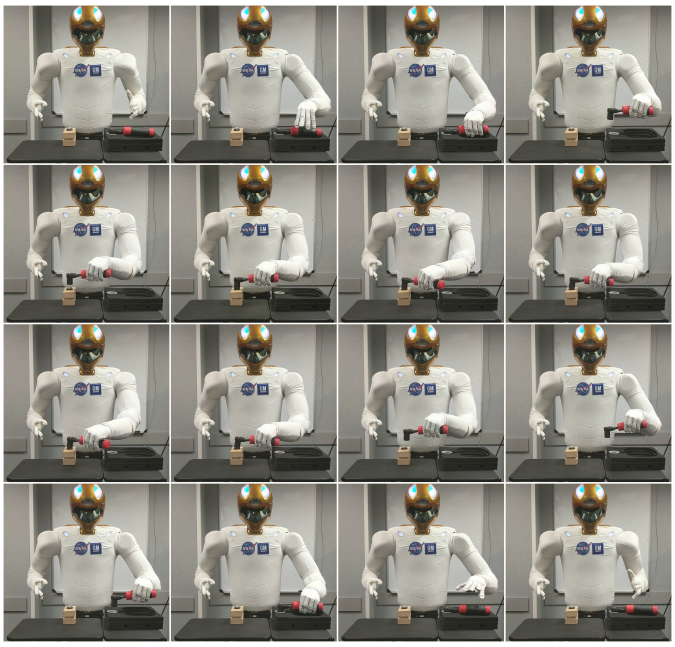

Fig. 9. The ratchet task sequence performed by Robonaut-2. The images from left to right, then top to bottom, show a sequence of actions where Robonaut-2 grasps the ratchet, tightens a bolt on a platform, and puts the ratchet back into a tool holder.

and the robot lifted the whole tool holder. Mating socket with the bolt and placing the ratchet back have 86.3% and 81.8% success rate. Tightening failed once when the socket slipped away from the bolt while tightening. 14 out of 24 trials succeeded the complete task.

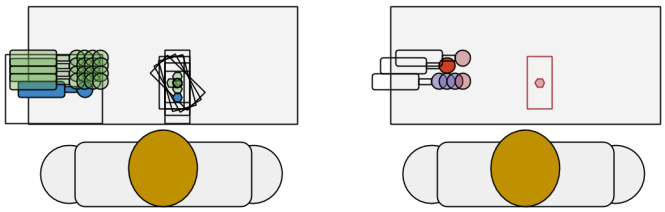

Fig. 10. Top down views of initial poses and failed poses on the ratchet task. The green objects in the left image shows a set of initial poses tested and the blue objects are the initial poses for the demonstrations. The pink objects in the right image shows a set of initial poses that failed to mate the socket with the bolt and the purple objects are the initial poses that failed to place the ratchet back. The red ratchet pose failed in both subtasks in two different trials.

TABLE I
NUMBER OF SUCCESSFUL TRIALS ON SUBTASKS.

subtask	(a)	(b)	(c)	(d)	(e)
successful trials / total trials	22 / 24	19 / 22	18 / 19	22 / 22	18 / 22

C. Evaluating Distilled ATGs

In this experiment, the success rate of mating the socket with the bolt is compared between two ATGs that are created from single demonstrations and an ATG distilled from the exact two ATGs. For each trial, the robot starts with the grasped ratchet

and the bolt placed on the right side of the robot. The trial succeeds if the robot mates the socket to the bolt. The starting hand position and the relative pose between the ratchet and the hand vary for each trial due to the different grasping result. We performed 11 trials for each ATG created from the single demonstration. The results are shown in table II. The ATGs created with single demonstration performed poorly due to using features that are on the handle of the ratchet instead of the socket.

TABLE II
COMPARING SUCCESS RATE BETWEEN DISTILLED ATG AND SINGLE DEMONSTRATION ATGS.

	distilled ATG	ATG from example 1	ATG from example 2
number of successful trials / total trials	16 / 22	0 / 11	4 / 11
success rate	0.727	0.182	

D. Comparing Accuracy in Simulation

To further understand how the number of demonstrations used affect the action accuracy, we compare ATGs created with one to five demonstrations in the Robonaut-2 simulator. For each of these five ATGs we tested 125 trials of the placing socket on top of the bolt task with different in-hand ratchet poses and bolt platform locations. For each trial, a perturbation $P = (r_{xy}, r_\theta, b_{xy})$ is added to an initial configuration, where r_{xy} is the ratchet offset in the XY plane in hand, r_θ is the ratchet angle difference on the Z axis in hand, and b_{xy} is the bolt platform offset in the XY plane. We tested all combinations of the following set of parameters, $r_{xy} = \{(0,0),(2,0),(0,2),(-2,0),(0,-2)\}$ in centimeters, $r_\theta = \{-0.2,-0.1,0,0.1,0.2\}$ in radians, and $b_{xy} = \{(0,0),(3,0),(0,3),(-3,0),(0,-3)\}$ in centimeters. For each trial, the distance between the final socket location and the ground truth socket location calculated based on the demonstration is recorded.

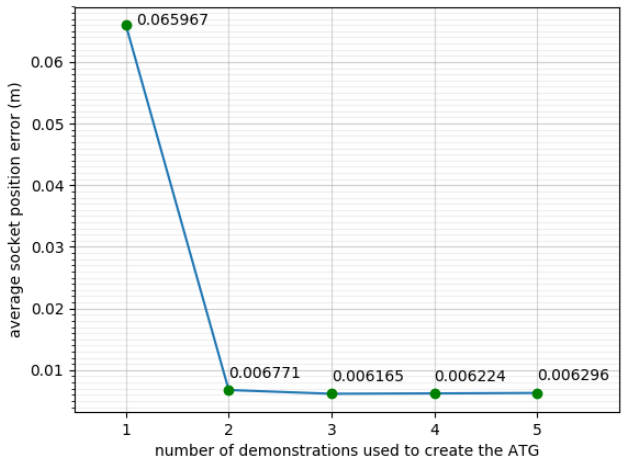

Fig. 11. The accuracy of the placing socket on top of the bolt task versus the number of demonstrations used to create the ATG.

The results are shown in Figure 11. With two demonstrations a distilled ATG can lower the socket position error significantly. Adding more demonstrations did not improve the accuracy much on this task. This may be because that with two demonstrations the visual features identified are already the best among the detected set. Figure 12 shows the informative features, represented by green dots, identified on the ratchet for ATGs created with one, two, and five demonstrations. The feature selected in the ATG created from a single demonstration is further away from the socket than the features selected by ATGs created from multiple demonstrations. This offset may result in less accurate actions when the ratchet is held in the hand with a different angle. In this case, the features identified among ATGs created from two to five demonstrations are similar, and therefore result in similar accuracy. This result is consistent to our findings in the Robonaut-2 experiment. With a small set of demonstrations, the distilled ATG identifies more informative features and lowers the action error significantly.

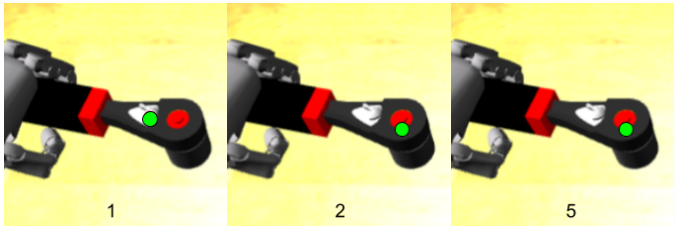

Fig. 12. Informative features identified in experiments in simulation. The images from left to right corresponds to tool aspect nodes for the putting socket on top of the bolt task using ATGs created from one, two, and five demonstrations. The green dots represent the visual features selected to represent the action. The feature selected in the ATG created from a single demonstration is further away from the socket and may result in less accurate actions.

V. CONCLUSIONS

In this work, we introduced a learning from demonstration approach that learns both actions and features. We categorize demonstrations into three different types depending on what frames or features are used as references and what is used to calculate the error to the target. Having the user provide additional information about the type of the demonstration allows the system to define the goal of the task by modeling the spatial relations between features automatically. Through multiple demonstrations, informative visual features and relative poses may be identified and used to model actions that are more accurate than models of single demonstrations. This effect is clearly observed in the improvement in success rate and accuracy over single demonstration models when mating the socket to the bolt. We show that with the proposed approach Robonaut-2 is capable of grasping the ratchet, tightening a bolt, and putting the ratchet back into a tool holder with a small set of demonstrations. In future work, we would like to determine the demonstration types automatically from multiple examples, generalize to unseen objects of the same class, and apply to other tool usage tasks.

VI. ACKNOWLEDGMENT

We are thankful to our colleagues Dirk Ruiken, Mitchell Hebert, Michael Lanighan, Tiffany Liu, Takeshi Takahashi, and former members of the Laboratory for Perceptual Robotics for their contribution on the control basis framework code base. We are also grateful to Philip Strawser, Jonathan Rogers and the Robonaut Team for their support on Robonaut-2. This material is based upon work supported under Grant NASA-GCT-NNX12AR16A and a NASA Space Technology Research Fellowship. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Aeronautics and Space Administration.

REFERENCES

[1] Baris Akgun, Maya Cakmak, Jae Wook Yoo, and Andrea Lockerd Thomaz. Trajectories and keyframes for kinesthetic teaching: A human-robot interaction perspective. In *Proceedings of the seventh annual ACM/IEEE international conference on Human-Robot Interaction*, pages 391–398. ACM, 2012.

[2] Sonya Alexandrova, Maya Cakmak, Kaijen Hsiao, and Leila Takayama. Robot Programming by Demonstration with Interactive Action Visualizations. In *Robotics: science and systems*, 2014.

[3] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. SURF: Speeded up robust features. In *Computer vision–ECCV 2006*, pages 404–417. Springer, 2006.

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

[5] Sylvain Calinon, Florent Guenter, and Aude Billard. On learning, representing, and generalizing a task in a humanoid robot. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 37(2):286–298, 2007.

[6] Paul Dinh and Stephen Hart. NASA Robonaut 2 Simulator, 2013. URL http://wiki.ros.org/nasa_r2_simulator. [Online; accessed 7-July-2014].

[7] Philipp Fischer, Alexey Dosovitskiy, and Thomas Brox. Descriptor matching with convolutional neural networks: a comparison to sift. *arXiv preprint arXiv:1405.5769*, 2014.

[8] Manfred Huber. *A hybrid architecture for adaptive robot control*. PhD thesis, University of Massachusetts Amherst, 2000.

[9] Li Yang Ku, Shiraj Sen, Erik G Learned-Miller, and Roderic A Grupen. Action-Based Models for Belief-Space Planning. *Workshop on Information-Based Grasp and Manipulation Planning, at Robotics: Science and Systems*, 2014.

[10] Li Yang Ku, Erik Learned-Miller, and Rod Grupen. An Aspect Representation for Object Manipulation Based on Convolutional Neural Networks. In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pages 794–800. IEEE, 2017.

[11] Li Yang Ku, Erik G Learned-Miller, and Roderic A Grupen. Associating Grasp Configurations with Hierarchical Features in Convolutional Neural Networks. In *Intelligent Robots and Systems (IROS), 2017 IEEE International Conference on*. IEEE, 2017.

[12] David G. Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[13] Peter Pastor, Heiko Hoffmann, Tamim Asfour, and Stefan Schaal. Learning and generalization of motor skills by learning from demonstration. In *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*, pages 763–768. IEEE, 2009.

[14] Claudia Pérez-D'Arpino and Julie A Shah. C-LEARN: Learning geometric constraints from demonstrations for multi-step manipulation in shared autonomy. In *IEEE International Conference on Robotics and Automation*, 2017.

[15] Mike Phillips, Victor Hwang, Sachin Chitta, and Maxim Likhachev. Learning to plan for constrained manipulation from demonstrations. In *Robotics: Science and Systems*, volume 5, 2013.

[16] Radu Bogdan Rusu and Steve Cousins. 3D is here: Point Cloud Library (PCL). In *IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China, May 9-13 2011.

[17] Shiraj Sen. *Bridging the gap between autonomous skill learning and task-specific planning*. PhD thesis, University of Massachusetts Amherst, 2013.

[18] Edgar Simo-Serra, Eduard Trulls, Luis Ferraz, Iasonas Kokkinos, Pascal Fua, and Francesc Moreno-Noguer. Discriminative learning of deep convolutional feature point descriptors. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 118–126, 2015.

[19] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller. Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806*, 2014.

[20] Ioan A. Sucan and Sachin Chitta. Moveit!, 2013. URL http://moveit.ros.org. [Online].

[21] Tinne Tuytelaars and Krystian Mikolajczyk. Local Invariant Feature Detectors: A Survey. *Foundations and Trends in Computer Graphics and Vision*, 3(3):177–280, 2007. ISSN 1572-2740, 1572-2759. doi: 10.1561/0600000017.

[22] Jason Yosinski, Jeff Clune, Anh Nguyen, Thomas Fuchs, and Hod Lipson. Understanding neural networks through deep visualization. *arXiv preprint arXiv:1506.06579*, 2015.