Leveraging Temporal Reasoning for Policy Selection in Learning from Demonstration

Estuardo Carpio, Madison Clark-Turner, Paul Gesel, and Momotaz Begum

Abstract— **High-level human activities often have rich temporal structures that determine the order in which atomic actions are executed. We propose the Temporal Context Graph (TCG), a temporal reasoning model that integrates probabilistic inference with Allen's interval algebra, to capture these temporal structures. TCGs are capable of modeling tasks with cyclical atomic actions and consisting of sequential and parallel temporal relations. We present Learning from Demonstration as the application domain where the use of TCGs can improve policy selection and address the problem of perceptual aliasing. Experiments validating the model are presented for learning two tasks from demonstration that involve structured human-robot interactions. The source code for this implementation is available at https://github.com/AssistiveRoboticsUNH/TCG.**

I. Introduction

Complex high-level tasks are typically composed of several atomic actions. The order in which these actions are executed is governed by a temporal structure that must be learned to develop a holistic model of the task. These temporal structures can be beneficial in domains such as learning from demonstration (LfD), where a task needs to be learned and replicated by a robotic agent. Specifically, the temporal structure of a task can be employed to perform policy selection and can be leveraged to address instances of perceptual aliasing [1]. The problem of perceptual aliasing occurs when two or more states that should lead to different actions generate the same set of perceptual information [2]. Perceptual aliasing has a negative effect in LfD, as it causes the model to confound task states that must be differentiated to learn an accurate policy.

LfD is a popular robot learning paradigm that derives task policies from the demonstrations of a lay user [3]. In the context of LfD, a policy is the mapping between the state of the world and the actions a robot should perform to complete a task. LfD has been widely used to learn policies for low-level tasks such as obstacle avoidance [4], assembly tasks [5], and tool handling [6]. Similarly, there has been work focused on applying LfD to learn high-level concepts and tasks such as object sorting [7], cooking related tasks [8], [9], and the delivery of a robot-mediated educational intervention [10]. Both in high and low-level LfD, tasks are completed by following its latent temporal structure to accomplish a goal. However, the vast majority of LfD frameworks create policies by focusing on the spatial features of a task, failing to take advantage of the implicit temporal structure that defines it.

Authors are with the Cognitive Assistive Robotics Laboratory, University of New Hampshire, Durham, NH 03824, USA {erp48, mbc2004, pac48, mbegum}@cs.unh.edu

In this paper we introduce the Temporal Context Graph (TCG), a graphical model that integrates the temporal semantics of Allen's interval algebra [11] with the probabilistic nature of n-gram models [12] to capture the temporal structure of a task. The combination of these elements makes TCGs capable of learning temporal structures with cyclical atomic actions and sequential and parallel relations. TCGs can be used in LfD frameworks to perform temporal reasoning and limit the action-space of the robotic agent, simplifying the policy selection process and addressing the issue of perceptual aliasing. We validate the performance of TCGs in two high-level LfD tasks.

II. Related Work

Different graphical models have been explored in the LfD literature to incorporate temporal information in the policy selection process. For example, the work reported in [13] employed a Hidden Markov Model (HMM) to construct skill trees that implicitly encode the sequence in which atomic actions are executed. The work in [14] proposed learning a sequence of temporal constraints that could be used by a high-level planner during execution. A similar algorithm is presented in [15], where tasks precedence graphs are introduced to encode spatio-temporal constraints between atomic actions. The work in [8] proposed influence graphs to model the sequence of actions needed to complete a task. These graphical approaches, however, are not capable of modeling tasks with repetitive atomic actions.

The LfD framework described in [16] constructs finite state machines to model the temporal relationships between atomic actions. Similarly, advanced sequence graph learning algorithms have been proposed in [17] to model tasks with repetitive actions. These models use multi-class classifiers to learn and control state transitions, which allows them to learn repetitive atomic actions. Nevertheless, this design makes them prone to fail when faced with perceptual aliasing.

The model in [18] employs an extension of the Hierarchical Dirichlet Process HMM (HDP-HMM) to address the problem of perceptual aliasing by learning a function that can provide multi-valued mappings. Then, a set of perceptual data that triggers a perceptual aliasing issue can map to multiple states, from which the HDP-HMM model selects the one with highest likelihood. In [19], the authors introduced the $\mathcal{IBP}$ (Indian Buffet Process) Coupled SPCM (Spectral Polytope Covariance Matrix) $\mathcal{CRP}$ (Chinese Restaurant Process)-HMM ($\mathcal{ICSC}$-HMM), a Bayesian non-parametric model that prevents perceptual aliasing by identifying and learning sub-goals that encode key temporal dependencies

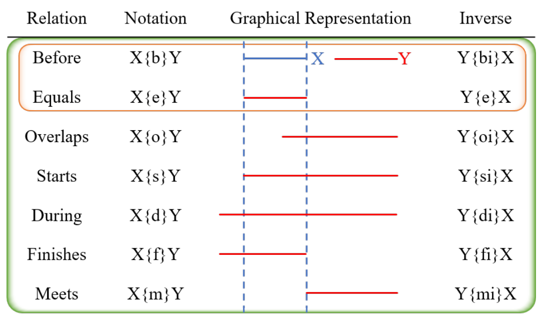

Fig. 1. Set of ITRs that can exist between two actions. The orange and green rectangles indicate the temporal relations that can be captured by point-based and interval-based temporal reasoning models respectively.

in the task. These models [18], [19], leverage temporal reasoning to deal with repetitive actions and perceptual aliasing. However, these approaches can only recognize point-based temporal features, meaning that they can only model three sequential temporal relations, namely *before, after,* and *equals*. This limitation makes it impossible for them to learn holistic models of tasks in which temporal relations other than the three point-based ones are present.

The Temporal Context Graph proposed in this paper is a novel way of learning complex temporal structures present in high-level tasks. TCGs employ Allen's interval algebra to encode the interval temporal relations (ITR) among the atomic actions of the task. These ITRs are then used to train an n-gram model that learns the dependencies between the state transitions and the temporal context of the task. This probabilistic approach is capable of handling cyclical atomic actions and can be leveraged to address instances of perceptual aliasing. To the best of our knowledge, this approach is the first to propose an interval-based temporal reasoning model capable of learning tasks with repetitive atomic actions.

III. Preliminaries

A. Interval Algebra

Complex tasks can be decomposed into a set of atomic actions. Each of these actions takes place over an interval of time that is defined by its start and end times. Allen and Ferguson [11] identified a set of 13 atomic interval temporal relations (ITR) that can exist between a pair of actions. These ITRs define and limit the order in which atomic actions take place during a task and can be used to create a model of its temporal dynamics. Employing interval algebra allows TCGs to model all the atomic ITRs, while point-based temporal models [18], [19] can only capture sequential temporal relations (Fig. 1).

B. N-grams

N-grams [12] are a popular sequence modeling tool in the field of natural language processing. N-gram models are utilized to simplify inference processes by using a predefined number of past states to select the future one. The number of past states that are used in the inference process is defined as $N-1$ where N is the order of the n-gram. These models have been employed in speech recognition [20], text categorization [21], and sentence completion [22]. In a TCG model, n-grams are used to perform policy selection based on the current temporal context of the task, addressing the issue of perceptual aliasing.

IV. Temporal Context Graph

A. Model Description

TCGs are temporal reasoning models capable of encoding the temporal structure of a task. This is achieved by identifying the ITRs present among the atomic actions of the task and using them to learn state transitions and their dependencies on the current temporal context of the task. The temporal context of a task is defined as the set of actions and observations that have taken place from the start of the execution of the task to the current point in time.

A TCG for a task is a directed graph for which each node represents a state of the task and edges represent internal or external events that, combined with the current temporal context, trigger a transition between two states. Accordingly, the TCG for a task can be formally defined as

$$TCG_T = \{N, E, P\} \tag{1}$$

where N is the set of nodes that constitute the task, E is the set of edges connecting the nodes in N, and P is a probabilistic n-gram model that encodes the transition probabilities between two nodes in N. The set of atomic actions of a task is used to create the nodes and edges in a TCG. Actions that generate a node are referred as *non-transition actions* while actions that spawn an edge in the model are called *transition actions*. An atomic action is defined by a quartet $\{l, s_t, e_t, \tau\}$ where l is a label used to identify the action, s_t and e_t are the start and end times, respectively, and τ indicates whether or not the action should be treated as a transition action.

Each node in a TCG consists of a quartet $\{\alpha, \delta, \omega_n, \Omega\}$ where α represents the atomic action that should be executed when the node is reached, δ indicates the duration of α, ω_n indicates the waiting period between the completion of α and the execution of a timeout transition, and Ω indicates whether or not the node is a terminal node.

An edge in a TCG consists of a quartet $\{\eta_o, \eta_d, \epsilon, \omega_e\}$ where η_o and η_d represent the origin and destination nodes, respectively, ϵ indicates the action that triggers the transition, and ω_e is the waiting period between the completion of ϵ and the execution of the atomic action indicated in node η_d.

Node transitions in a TCG can be of two different kinds, *action transitions* occur when an atomic action is observed. Meanwhile, *timeout transitions* are triggered when the waiting period ω_n at a given node is elapsed without observing a valid transition action. Transitions in TCGs are conditioned by the current temporal context of the task. If an incoming action observation cannot trigger a transition given the current state and temporal context of the task, it is considered an *invalid action observation*. These observations are treated as false positives and are disregarded by the model when performing temporal reasoning.

The state of a TCG at time t, S_t, is formally defined as

$$S_t = \{n_t, \omega_t, \epsilon, c\} \tag{2}$$

where $n_t \in N$ represents the current node of the TCG, ω_t is the waiting period before a timeout is performed and the next action is selected, ϵ is the action observed in the environment, and c is a sequence of ITRs describing the current temporal context of the task. The temporal context c and the current action observation ϵ are leveraged by the n-gram model P to perform inference during policy selection.

B. Learning a TCG

The structure and parameters of a TCG are learned from a demonstration set, D. Each demonstration in D consists of a set of atomic actions, their respective start and end times and a flag indicating if they are transition actions. For an LfD task, such a demonstration set can be created autonomously by employing segmentation techniques capable of identifying the start and end times of the atomic actions that constitute the task. For example, the research in [19] and [23] present segmentation techniques for low- and high-level LfD tasks, respectively. In an LfD task, the robot actions are marked as non-transition actions because the TCG is learned from the robot's perspective.

The three stages of the TCG learning process are outlined below and can be seen in Algorithm 1.

1) Structure Learning: The first stage of the TCG learning process consists of learning the graphical structure for the given task. The first step in this stage consists of sorting the atomic actions available in the demonstration set, according to their start times (line 3). The sorted action sequences of each demonstration are then processed individually to learn the graphical structure that represents the task. Nodes are created for each distinct non-transition action (line 16), while transition actions are used to create the edges connecting those nodes (line 15). Additionally, timeout transition edges are created when two consecutive non-transition actions exist in a sequence (line 13). During this process the mean duration and waiting periods for each atomic action are learned and stored in the edges and nodes of the TCG (lines 17,19, 21). The output of this stage can be considered a finite state machine that encodes the valid transitions between the atomic actions of the given task.

Fig. 2 displays the TCG learning process for a toy problem in which a set of two demonstrations ($d1, d2$) is provided for a task. In this example, non-transition and transition actions are represented by uppercase and lowercase letters, respectively. Fig. 2(b) shows the TCG structure learned from the set of demonstrations displayed in 2(a). Non-transition actions A, B and C generate the nodes of the graph, while the transition action x creates the edges that link the nodes together. Additionally, a timeout transition (T/O) is added due to the $B \rightarrow B$ transition in d_1.

2) ITR Sequence Generation: The second stage consists of generating a sequence of ITRs from the sorted sequence of actions of each demonstration. This is achieved by calculating the temporal distance between every pair of consecutive

Algorithm 1 TCG Learning

```
Input: $D$
Output: $TCG$
1:  initialize: $N \leftarrow \emptyset$, $E \leftarrow \emptyset$, $P \leftarrow \emptyset$, $ITR \leftarrow \emptyset$
2:  for $d$ in $D$ do
3:      $d \leftarrow sort(d)$
4:      $itr \leftarrow \emptyset$
5:      for $a$ in $d$ do
6:          $next \leftarrow a.next$
7:          $itr \leftarrow itr \cup get\_itr(a, next)$
8:          if $\neg\ a.\tau$ then
9:              if $next.\tau$ then
10:                 $next \leftarrow next.next$
11:                 $transition \leftarrow a.next$
12:             else
13:                 $transition \leftarrow Timeout$
14:             end if
15:             $E \leftarrow E \cup edge(a, next, transition)$
16:             $N \leftarrow N \cup a \cup next$
17:             $N.update\_duration(a)$
18:             if $transition == Timeout$ then
19:                 $N.update\_timeout(a, next)$
20:             else
21:                 $E.update\_timeout(transition)$
22:             end if
23:         end if
24:     end for
25:     $ITR \leftarrow ITR \cup itr$
26: end for
27: $ITR \leftarrow itr\_factoring(ITR)$
28: $P \leftarrow learn\_ngram(ITR)$
29: return $TCG(N, E, P)$
```

atomic actions (line 7). The temporal distance for two actions x and y is defined as follows: (the symbols x and y are not related to those in Fig. 2)

$$d(y, x) = \big(s_y - s_x, e_y - e_x, s_y - e_x, e_y - s_x\big) \tag{3}$$

where x is the temporal reference of y, and s and e represent the start and end times of an action, respectively. The results of this operation are used to identify the ITR that exists between the two actions [24].

3) Temporal Context Learning: The third, and final, stage is the temporal context (c, in equation (2)) learning. There are two steps in this process. First, the set of ITR sequences is simplified using a process called *ITR factoring* (line 27). This process consists of grouping together ITRs that share a common temporal context and lead to a common state in the task. ITR factoring is necessary to prevent the n-gram model from becoming too sparse. Fig. 2(c) shows an example of the ITR factoring process. The ITRs highlighted in green display an example in which distinct ITRs (o, b) are grouped into a single ITR factor (*1*), because they share a common temporal context and map to the same task state (A). Meanwhile, the ITRs marked in red show a case in which two ITRs (fi, b) create two distinct ITR factors (*3, 7*) because, even

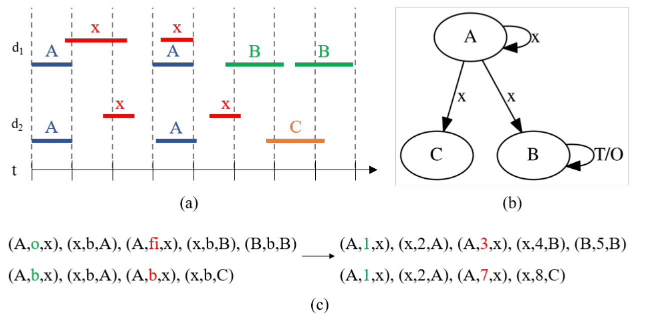
d1
A
x
x
A
B
B
d2
A
x
A
x
C
t
(a)
A
x
x
x
C
B
T/O
(b)
(A,o,x), (x,b,A), (A,fi,x), (x,b,B), (B,b,B)
(A,1,x), (x,2,A), (A,3,x), (x,4,B), (B,5,B)
(A,b,x), (x,b,A), (A,b,x), (x,b,C)
(A,1,x), (x,2,A), (A,7,x), (x,8,C)
(c)

Fig. 2. An example of TCG learning process. (a) The demonstration set used as input to learn the TCG model. (b) Graphical structure of the TCG model learned from the demonstrations shown in (a). (c) Illustrates the ITR factoring process. The left and right sides show the original and factored ITR sequences, respectively.

though they share a common temporal context, they map to a different task state (B and C respectively). The distinction between the *finishes* (fi) and *before* (b) ITRs illustrated in this toy example cannot be learned with the point-based temporal reasoning models reported in [18], [19].

The second step consists of learning a probabilistic n-gram model using the factored ITR sequences as input (line 28). During the training of the n-gram model, the previous $n-1$ ITRs of the sequence are used as the evidence to encode the temporal context that generates the action executed in the n-th ITR. The resulting n-gram model encodes all the observed temporal contexts; therefore, it can be leveraged during policy selection to address the issue of perceptual aliasing. This is possible because the actions selected by the n-gram model will be dependent on the current temporal context of the task.

C. Policy Selection Using a TCG

The policy selection process in a Temporal Context Graph consists of two phases that are executed at each time step. The first phase updates the state (S_t) of the TCG with the latest transition action ($S_t.\epsilon$) that is observed in the environment. Meanwhile, the second phase is used to verify if a node transition needs to be triggered, prompting the TCG to select the next action to be executed (n_{t+1}). This process is shown in Algorithm 2.

The first phase starts when a new transition action is observed in the environment. At that time, the model evaluates the possible transitions from the current node of the graph and discards the action observation if a valid transition is not found (line 2). If the action observation is valid, the timeout (ω_t) and transition action (ϵ) of S_t are updated (lines 3-4).

The actual policy selection occurs during the second phase of the process. This phase is only triggered when the state timeout ($S_t.\omega_t$) is reached (line 6). To select the next action (n_{t+1}) the TCG leverages the current temporal context (c), which is utilized by the n-gram model to perform an inference operation (line 7). After the next atomic action is selected, the current node n_t, temporal context c, and timeout ω_t of S_t are updated to reflect the current state of the task (lines 11-13).

Algorithm 2 Policy Selection in a TCG

```
Input: $P, S_t, t, \epsilon$
Output: $n_{t+1}$
 1: initialize: $n_{t+1} \leftarrow \emptyset$
 2: if $is\_valid\_obs(S_t.n_t, \epsilon)$ then
 3:     $S_t.\epsilon_t \leftarrow \epsilon_t$
 4:     $S_t.\omega_t \leftarrow \epsilon_t.\omega_e$
 5: end if
 6: if $t > S_t.\omega_t$ then
 7:     $n_{t+1} \leftarrow P.evaluate(S_t.c)$
 8:     if $n_{t+1} == \emptyset$ then
 9:         $n_{t+1} \leftarrow Abort$
10:     end if
11:     $S_t.c \leftarrow S_t.c \cup get\_itr(S_t.n_t, n_{t+1})$
12:     $S_t.n_t \leftarrow n_{t+1}$
13:     $S.w_t \leftarrow n_{t+1}.\omega_n + n_{t+1}.\delta + t$
14: end if
15: return $n_{t+1}$
```

V. Evaluation Domain

The performance of the TCG was evaluated using two human-robot interactions (HRI) tasks. The tasks and the user studies organized to collect the demonstration sets are explained below.

A. Object Naming Intervention

The first use case consists of a robot-mediated educational intervention for children with autism spectrum disorder (ASD). Mounting anecdotal evidence in the HRI literature shows that robots may outperform human teachers in teaching a wide range of basic skills to children with ASD [25]. To facilitate autonomous learning (by the robot) of the steps of an educational intervention from the demonstration of a human teacher, we are particularly interested in Applied Behavior Analysis (ABA)-based interventions. ABA is well known for its rigid structure and unparalleled success in teaching basic skills to children with ASD [26]. Our previous work designed an LfD framework to learn ABA-based *social greetings* intervention from an expert's demonstration [10]. In any ABA-based intervention, the interaction between two agents (a robot teacher and a student) evolves in the following way: COMMAND $\rightarrow$ Response (CORRECT or INCORRECT) $\rightarrow$ PROMPT (if required) $\rightarrow$ REWARD $\rightarrow$ ABORT. In this paper, we learn an *object naming* intervention designed to improve the vocabulary of a child. The intervention begins with a robot teacher delivering a COMMAND by asking a participant to name an object placed on the workspace and pointing at it. The participant may stay silent or respond verbally with a CORRECT or INCORRECT response. If the participant does not provide a correct response, the teacher proceeds by delivering a PROMPT that indicates the correct answer and invites them to try again, e.g. "John, this is a basketball. Can you tell me what this is?". If the intervention is not being successful, the teacher can provide more prompts or ABORT the session. If

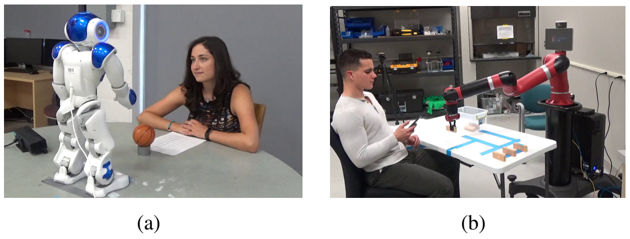

Fig. 3. Physical setup used for the (a) Object naming intervention (b) Collaborative packaging task.

the participant provides an appropriate response, the teacher concludes the session by giving a verbal REWARD to the participant, such as “Great job!”. Our goal is to learn the entire task from demonstrations.

In this intervention the robot can perform four actions: COMMAND, PROMPT, REWARD and ABORT. In the context of TCG, these actions are labeled as non-transition actions. The only transition action is the response provided by the human participant, which can be CORRECT or INCORRECT. This intervention was selected as a validation use case for three reasons. First, it presents a cyclical atomic action (PROMPT), which in this design, can be executed up to four times if a CORRECT response is not provided by the participant. Second, it triggers multiple instances of perceptual aliasing. Perceptual aliasing occurs when the robot needs to decide if a PROMPT or ABORT action needs to be delivered after receiving an INCORRECT response, or no response at all, from the participant since both of these responses are perceptually similar. Third, it has the presence of parallel temporal relations because, in this design, participants can elicit responses that exhibit an *overlaps* or *during* ITR with respect to a COMMAND or PROMPT action executed by the robot.

An IRB-approved user study was organized to collect demonstration data for this use case. A total of 11 subjects (9 male, 2 female) without ASD participated in the study. Each subject was requested to engage in 10 successful interactions (i.e. providing the CORRECT response at some point) and 5 unsuccessful interactions (i.e. never providing a CORRECT response) with a tele-operated robot. Participants were asked to provide at least 3 responses that exhibited the parallel relations (*overlaps* or *during*). The participants were allowed to choose the number of prompts they would like to receive before they elicit a CORRECT response and whether they wanted to provide an INCORRECT response or no response at all when they were not providing a CORRECT response.

A NAO humanoid robot was tele-operated to deliver the intervention in this study (Fig. 3(a)). The on board speech recognition module of the robot was used to detect the responses provided by the human participant. The resulting set of 165 demonstrations was separated in a training and a validation set that contained 123 and 42 demonstrations, respectively.

B. Collaborative Packaging Task

This use case is about human-robot collaboration in a packaging task in which the goal is to place a predefined number of items in a box to prepare them for shipping. This is a very common task in manufacturing environments. During this task, a robotic arm will reach for one of the items (PICK) and place it in a pre-designated area (P_DQA) for human inspection. Once the robot does that, the human participant will start a QA inspection (INS). This type of inspection consists of reviewing the state of the item and logging its serial number in an electronic form that notifies the robot when the inspection is completed. As the human conducts the inspection, the robot will reach for another item and will observe the following rule for the placement of item: if an inspection is currently being conducted by the human, the robot will place the newly picked item directly in the box (P_BOX); if the last inspection was completed *before* the item was retrieved, the robot will place it in the pre-designated area (P_DQA) for inspection; if the last inspection exhibited a *during* ITR with respect to the current item retrieval, the item is dropped in the superficial QA inspection (P_SQA) area. The superficial inspection consists of a quicker revision of the item and logging the item type in the electronic form mentioned above. This set of actions is repeated until 6 items have been retrieved by the robot. Our goal is to learn the entire operation from demonstrations.

In this task, the robot can perform four actions: PICK, P_DQA, P_SQA and P_BOX. These actions are labeled as non-transition actions, and the human action (INS) is the only transition action in the TCG context. There are multiple actions in this task that can exhibit a cyclical behavior. For example, the PICK and P_DQA actions can be executed up to six times each in a single demonstration. Meanwhile, perceptual aliasing is present every time a robot action needs to be executed. This is because the only features available to the robot are the states of its joints, and every robot action starts and ends at a common resting position. Lastly, this use case has actions that depend directly on specific temporal relations among events. For example, where the robot will place a retrieved item after the PICK action depends on the status of the INS action of the human. This highlights the advantages of using interval-based temporal models, as the distinction between the *before, during*, and *after* ITRs needed to perform policy selection in this case cannot be learned with point-based approaches.

A second IRB-approved user study was organized to collect demonstration data for this use case. A tele-operated Sawyer robot was used for data collection. The physical setup for this task is shown in Fig. 3(b). Two male college students participated in the study and provided 5 demonstrations of the task. The demonstration data was further complemented with simulated demonstrations in which the start and end times of the human actions were generated at random.

VI. Experimental Results

Experiments were conducted to evaluate the ability of a trained TCG model to execute a learned task autonomously. The same physical set-ups shown in Fig. 3 were used in these experiments. A video demonstrating the automated executions can be found at: https://goo.gl/TeibPC.

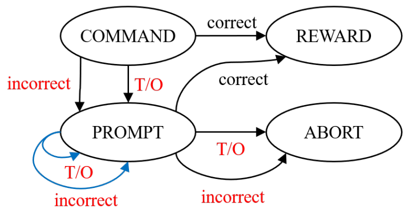

Fig. 4. TCG model learned from the demonstration data of the object naming intervention. Cyclical actions and transitions that trigger cases of perceptual aliasing are marked in blue and red, respectively.

A. Object Naming Intervention

The TCG model learned for the object naming task is shown in Fig 4. The transition actions that can generate perceptual aliasing are marked with red to highlight the importance of leveraging temporal information during the execution of this task.

The policy selection capabilities of the model were tested by using it to autonomously deliver the 42 interventions of the validation set. An intervention was considered successful if the TCG generated the exact same sequence of actions as the original recording. The model was able to successfully deliver the intervention in 96% of the sessions (2 failures). The first failed intervention was caused by a failure of the speech recognition module. The second was caused when the model delivered a PROMPT action before the participant provided a response. These results prove that the model effectively handled all the instances of perceptual aliasing that exist in the validation set. Moreover, the learning capabilities of the TCG were highlighted by the fact that 4 prompts were always delivered before aborting a session, which matched the maximum number of prompts delivered in the demonstration sessions. A validation user study was conducted to evaluate the performance of the TCG models in real-time autonomous interventions. Four college students without ASD participated in this study and were made aware that the robot was acting autonomously. Each participant completed a set of 15 sessions for the object naming intervention, following the same instructions described in section V-A. An intervention was considered successful if the model allowed the robot to act in accordance to the behavior exhibited by the human participant and the state of the intervention until a terminal action (REWARD or ABORT) was executed. In this experiment, 96.7% of the automated interventions were successful (58 successful, 2 failures). Both of the failed interventions were caused by failures of the speech recognition module to recognize the responses of the participants. The behavior observed in this user study mimicked what had been observed in the demonstration set, with the TCG model executing exactly 4 prompts, in the case of a non-compliant participant, before aborting the session.

B. Collaborative Packaging Task

The TCG model learned for the collaborative packaging task is displayed in Fig 5. An additional validation user study was conducted to evaluate the performance of the TCG models in this use case. Two college students participated in this study and were made aware that the robot was acting autonomously. In this experiment, all 10 of the automated executions were successful. An execution was deemed successful if the model allowed the robot to act in accordance to the actions performed by the human participant and the state of the intervention until the all the items were placed in the box. The behavior observed in the autonomous executions of this task matched what was observed in the demonstration set. Specifically, the TCG model was able to select the appropriate action depending on the ITRs observed between the latest INS and PICK actions on all of the 60 instances where this decision was performed.

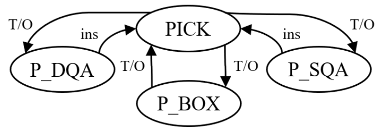

Fig. 5. TCG model learned from the demonstrations of the collaborative packaging task.

VII. Conclusion

This paper introduces the Temporal Context Graph, the first interval-based temporal reasoning model capable of learning structures with cyclical atomic actions. The model relies on three principal components to perform temporal reasoning. The first is a graphical structure that captures the set of valid state transitions of the task and is used to filter incorrect action observations. The second is Allen's interval algebra, which is used to create ITR sequences that encode the temporal context of the task. The third, and last, component is a probabilistic n-gram model. This model leverages the rich temporal context created with the ITR sequences to perform policy selection during the execution of the task. The model was evaluated using two use cases consisting of structured human-robot interactions. The results demonstrate that TCGs can be used to learn the underlying temporal structure of a task and perform policy selection, exploiting this structure to address the issue of perceptual aliasing. Additionally, the validation use cases demonstrate that using an interval-based approach allows TCGs to learn non-sequential temporal relationships. As a result, TCGs can effectively learn tasks that cannot be modeled using point-based temporal reasoning models.

Future work could include expanding the model to encode the uncertainty of incoming action observations and learning relevant non-sequential ITR sequences during the learning phase. These enhancements would increase the performance of the model during policy execution, reducing the dependency on accurate perception modules and increasing its robustness when faced with unseen ITR sequences during the execution of the task.

Acknowledgment

This work was supported in part by the National Science Foundation (IIS 1664554).

References

[1] S. D. Whitehead and D. H. Ballard, "Learning to perceive and act by trial and error," *Machine Learning*, vol. 7, no. 1, pp. 45–83, 1991.

[2] D. H. Grollman and O. C. Jenkins, "Incremental learning of subtasks from unsegmented demonstration," in *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*. IEEE, 2010, pp. 261–266.

[3] 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.

[4] S. R. Ahmadzadeh, R. Kaushik, and S. Chernova, "Trajectory learning from demonstration with canal surfaces: A parameter-free approach," in *Humanoid Robots (Humanoids), 2016 IEEE-RAS 16th International Conference on*. IEEE, 2016, pp. 544–549.

[5] C. Paxton, F. Jonathan, M. Kobilarov, and G. D. Hager, "Do what i want, not what i did: Imitation of skills by planning sequences of actions," *arXiv preprint arXiv:1612.01215*, 2016.

[6] S. Elliott, Z. Xu, and M. Cakmak, "Learning generalizable surface cleaning actions from demonstration," in *Robot and Human Interactive Communication (RO-MAN), 2017 26th IEEE International Symposium on*. IEEE, 2017, pp. 993–999.

[7] R. Cubek, W. Ertel, and G. Palm, "High-level learning from demonstration with conceptual spaces and subspace clustering," in *Robotics and Automation (ICRA), 2015 IEEE International Conference on*. IEEE, 2015, pp. 2592–2597.

[8] N. Koenig and M. J. Matarić, "Robot life-long task learning from human demonstrations: a bayesian approach," *Autonomous Robots*, vol. 41, no. 5, pp. 1173–1188, 2017.

[9] K. Bullard, B. Akgun, S. Chernova, and A. L. Thomaz, "Grounding action parameters from demonstration," in *Robot and Human Interactive Communication (RO-MAN), 2016 25th IEEE International Symposium on*. IEEE, 2016, pp. 253–260.

[10] M. Clark-Turner and M. Begum, "Deep reinforcement learning of abstract reasoning from demonstrations," in *Proceedings of the 2018 ACM/IEEE International Conference on Human-Robot Interaction*. ACM, 2018, pp. 160–168.

[11] J. F. Allen and G. Ferguson, "Actions and events in interval temporal logic," *Journal of logic and computation*, vol. 4, no. 5, pp. 531–579, 1994.

[12] C. E. Shannon, "A mathematical theory of communication," *ACM SIGMOBILE Mobile Computing and Communications Review*, vol. 5, no. 1, pp. 3–55, 2001.

[13] G. Konidaris, S. Kuindersma, R. Grupen, and A. Barto, "Robot learning from demonstration by constructing skill trees," *The International Journal of Robotics Research*, vol. 31, no. 3, pp. 360–375, 2012.

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

[15] M. Pardowitz, R. Zollner, and R. Dillmann, "Learning sequential constraints of tasks from user demonstrations," in *Humanoid Robots, 2005 5th IEEE-RAS International Conference on*. IEEE, 2005, pp. 424–429.

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

[17] S. Manschitz, J. Kober, M. Gienger, and J. Peters, "Learning to sequence movement primitives from demonstrations," in *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*. IEEE, 2014, pp. 4414–4421.

[18] J. Butterfield, S. Osentoski, G. Jay, and O. C. Jenkins, "Learning from demonstration using a multi-valued function regressor for time-series data," in *Humanoid Robots (Humanoids), 2010 10th IEEE-RAS International Conference on*. IEEE, 2010, pp. 328–333.

[19] N. Figueroa and A. Billard, "Learning complex manipulation tasks from heterogeneous and unstructured demonstrations," in *IROS Workshop on Synergies between Learning and Interaction*, 2017.

[20] D. Jurafsky and J. H. Martin, *Speech and language processing*. Pearson London:, 2014, vol. 3.

[21] W. B. Cavnar, J. M. Trenkle, *et al.*, "N-gram-based text categorization," *Ann arbor mi*, vol. 48113, no. 2, pp. 161–175, 1994.

[22] Z. Su, Q. Yang, Y. Lu, and H. Zhang, "Whatnext: A prediction system for web requests using n-gram sequence models," in *Web Information Systems Engineering, 2000. Proceedings of the First International Conference on*, vol. 1. IEEE, 2000, pp. 214–221.

[23] A. Murali, A. Garg, S. Krishnan, F. T. Pokorny, P. Abbeel, T. Darrell, and K. Goldberg, "Tsc-dl: Unsupervised trajectory segmentation of multi-modal surgical demonstrations with deep learning," in *Robotics and Automation (ICRA), 2016 IEEE International Conference on*. IEEE, 2016, pp. 4150–4157.

[24] Y. Zhang, Y. Zhang, E. Swears, N. Larios, Z. Wang, and Q. Ji, "Modeling temporal interactions with interval temporal bayesian networks for complex activity recognition," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 10, pp. 2468–2483, 2013.

[25] M. Begum, R. W. Serna, D. Kontak, J. Allspaw, J. Kuczynski, H. A. Yanco, and J. Suarez, "Measuring the efficacy of robots in autism therapy: How informative are standard hri metrics'," in *Proceedings of the Tenth Annual ACM/IEEE International Conference on Human-Robot Interaction*. ACM, 2015, pp. 335–342.

[26] R. M. Foxx, "Applied behavior analysis treatment of autism: The state of the art," *Child and adolescent psychiatric clinics of North America*, vol. 17, no. 4, pp. 821–834, 2008.