Efficient Inference of Temporal Task Specifications from Human Demonstrations using Experiment Design

Shlok Sobti, Rahul Shome, and Lydia E. Kavraki

***Abstract*— Robotic deployments in human environments have motivated the need for autonomous systems to be able to interact with humans and solve tasks effectively. Human demonstrations of tasks can be used to infer underlying task specifications, commonly modeled with temporal logic. State-of-the-art methods have developed Bayesian inference tools to estimate a temporal logic formula from a sequence of demonstrations. The current work proposes the use of experiment design to choose environments for humans to perform these demonstrations. This reduces the number of demonstrations needed to estimate the unknown ground truth formula with low error. A novel computationally efficient strategy is proposed to generate informative environments by using an optimal planner as the model for the demonstrator. Instead of evaluating all possible environments, the search space reduces to the placement of informative orderings of likely eventual goals along an optimal planner's solution. A human study with 600 demonstrations from 20 participants for 4 tasks on a 2D interface validates the proposed hypothesis and empirical performance benefit in terms of convergence and error over baselines. The human study dataset is also publicly shared.**

I. Introduction

Interest in robotics applications has been increasingly shifting from purely autonomous industrial settings towards more assistive use cases among humans. Such problem domains necessitate robots to adapt to a wide variety of tasks potentially specified by humans. Wider adoption of automation requires removing some of the traditional accessibility barriers of rigorous scientific knowledge in robot programming. Frameworks need to be designed to effectively recover the structure of task constraints and objectives like the motivating scenarios in Fig 1 (top). Here, before helping with a recipe, the autonomous framework can use human demonstrations to first recover a representation of the underlying task. This motivates efficient inference of task specifications from humans to reduce the need for expert knowledge, demonstration effort, and error.

Different representations exist for task constraints and objectives [1]–[4]. Formal methods have been rigorously studied with linear temporal logic (LTL) [5] being a commonly used encoding for AI tasks that have been applied to robotics applications [6]–[8]. LTL augments first-order logic and proves useful in expressing a wide variety of general task specifications. Note that *temporal* in LTL primarily pertains to ordering task actions and a discrete notion of time. As a motivating example, consider a recipe for apple pie dictating

The authors were affiliated to the Department of Computer Science, Rice University, Houston during the completion of this work and were partially supported by NSF RI 2008720 and Rice University Funds. RS is currently affiliated to the School of Computing, the Australian National University, Canberra. Contact: rahul.shome@anu.edu.au, kavraki@rice.edu.

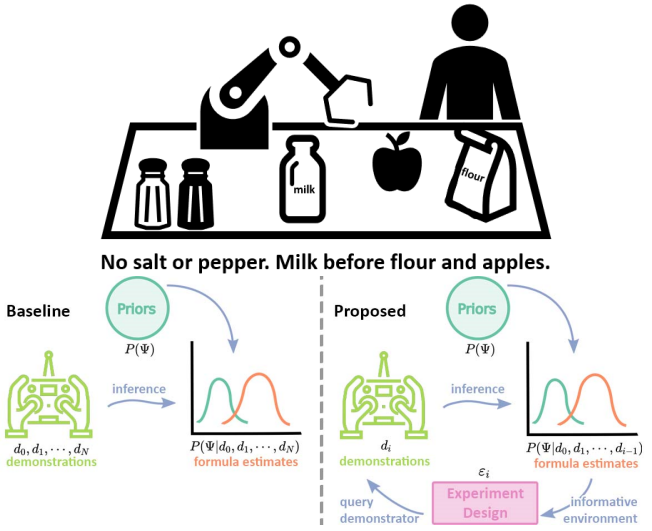

Fig. 1. A collaborative application (top) with a human demonstrating the recipe for apple pie to a robot. Task objective inference from demonstrations (bottom left) can be improved by experiment design (bottom right).

that we mix in milk before flour and apples, while we do not add salt or pepper at any point (Fig 1 (top)). We can express such an example in LTL.

Existing planners capable of handling LTL constraints require syntactically sound formulae as inputs. These input formulae can tend to be unintuitive and verbose to non-experts. A robot programmer needs time and expertise in formal methods to design these inputs. To provide easier mechanisms to define such task specifications, existing lines of work [9]–[11] have shown that human demonstrations can be effective. Here a demonstrator can teach the task representation to the robot. Existing methods have focused on making the inference process efficient given a set of input demonstrations (similar to Fig 1 (bottom left)). State-of-the-art Bayesian inference tools [9] have been developed, as well as methods [11] that model the conditions of demonstrator optimality to optimize task inference over demonstrations. In these works, there exists an implicit assumption or need for diverse demonstrations which an expert demonstrator can carefully introduce. There is also a preference for shorter, over-constrained specifications. In the recipe example the following diversity in demonstrations is required to recover that apples and flour can be swapped — *add milk then flour then apples* and *add milk then apples then flour*.

It is important to note that *a non-expert demonstrator might have no natural propensity toward introducing demonstration diversity for identical experiments.* This propels our primary hypothesis that experiment design can serve as a

way to obtain informative and diverse demonstrations from non-expert human demonstrators. This reduces the number of demonstrations required to infer the true (potentially under-constrained) underlying task specification. Using principles from experiment design imposes a computational bottleneck. Naive strategies to explore assignments that rearrange the environment forming the demonstration interface become combinatorially intractable as the number of objects, regions, and locations increases. We propose an efficient experiment design strategy by generating environments informed by an optimal planner as an approximate model for the demonstrator. At a high level, we first identify the likely global constraints. Then we can identify a permutation of eventual goals that maximally challenges the current ordering constraint estimates. In spatial settings, the regions associated with goals can be placed in sequence along an optimal path that respects global constraints. Such an environment incentivizes the demonstrator to provide a demonstration distinct from previous ones. We validate this strategy in a human study via a 2-D interface that is designed to be similar to benchmarks in state-of-the-art works [9]–[11].

Our primary contributions in the current work are as follows: a) we use experiment design to reduce the number of demonstrations needed to infer task specifications from humans; b) we propose an efficient environment design mechanism by using an optimal planner as a model for the demonstrator; c) we validate our hypothesis in extensive human studies with **600** experiments on a 2D interface with **20** non-expert participants where our method outperforms the state-of-the-art by needing fewer demonstrations and always converging to low error estimates within 10 demonstrations, d) we publicly share the dataset from the humans study.

II. Related Work

Formal Methods: Formal methods have been used to model complex robotic systems, affording various kinds of guarantees on behavior. These techniques are essentially combinations of *specification*, *verification*, and *implementation*. Various grammars [12] have been proposed to *specify* requirements, each differing in the kinds of behaviors they can express. LTL [5], used in this work, is particularly suitable to robotics applications since it can encode temporal constraints.

Planning under Logical Specifications: Logical formalisms can be used to generate plans that satisfy input requirements. Sampling-based planners [13], [14] have been adapted to deal with such formal constraints [15]. Beyond lower-level motion planning, these are effective tools to express constraints on manipulation planning [16], [17]. Further, LTL specifications have also been shown effective in improving upon learned demonstrations [7]. These lines of work assume as inputs *well-formed formulas* that correctly capture the underlying true constraints. Constructing these expressions typically requires expertise in formal methods. In contrast, we focus on inferring accurate estimates of these task specifications from non-expert human demonstrations.

Learning Models from Demonstrations: Learning from demonstration (LfD) has been widely applied to robotics [18], [19]. Here an expert agent provides demonstrations in an attempt to transfer concepts to the robot. This is framed as a *reward learning* [2], [20] problem. It is non-trivial to convert a reward function to a logical specification [21] that yields stronger guarantees and certificates. Learning [22] in human interactions has been applied to shared task knowledge focusing on continuous-time reactivity but not for recovering general temporal logic specification.

Inferring Logical Specifications: Given the benefits afforded by formal methods, there has been recent interest in effectively inferring these representations. This has been examined from the perspective of describing differences in demonstrations [23]. Inferring LTL has been reduced to a boolean satisfiability problem over an equivalent alternating finite automaton [24] in the presence of positive and negative examples. Decision trees have also been shown to be effective at inferring logical specifications given a set of traces.

Prior work has used the principle of maximum entropy to infer a specification given candidates [10]. Inference has also been cast as a constraint satisfaction problem given a finite set of demonstrations [25]. These methods need a large set of diverse demonstrations (or an expert demonstrator) or risk converging to overfitted estimates of the true task. Optimization techniques (given an objective function) have shown promise [11], [26]. The hard assumption of a cost function allows for an iterative search for a satisfying formula with a lower cost [11] - in the case that the demonstration happens to be at a minimum, the search for counterexamples will terminate (despite the possibility of learning an over-constrained formula). The current work connects to this line of work [11] in its use of an assumed model of the demonstrator to inform the process, though we do so for experiment design. The work proposed in this work builds upon state-of-the-art that frames this problem in a Bayesian context [9] - such a method relies on the assumption of a diverse dataset to avoid learning an over-constrained formula. We use this underlying inference scheme in the current work.

Active Learning: Active learning [27], where an expert is involved in the inference process, has been applied to specification learning to improve efficiency. These techniques have been successfully applied to reward learning [2] to learn complex reward landscapes with improved data complexity. The requirement of a handcrafted feature space (for reward functions) has been relaxed [28] - the human expert is asked to guide the system to regions where a feature (to be learned) is highly expressed. In the context of formal methods, all grammars are expressed over symbols that abstract the raw states. Active learning methods have been able to learn these symbols [29] via interactions. In the context of LTL inference, this has been posed as the system querying for positive or negative labels to performed actions [30] where the queries are sampled to reduce uncertainty. This requires that negative examples are not explicitly unsafe to perform. We note that advances in active learning stem from the field of *experiment design* [31], which has also inspired our work.

III. Problem Formulation

In this section, we set up the core task specification inference problem. A demonstrator $\mathcal{D}$ provides demonstrations in an environment $\mathcal{E}$. The environment contains a set of k regions $\{\mathcal{R}_1, \cdots \mathcal{R}_k\}$ which induce interactions along a demonstration. The demonstration can correspond to a recording of some degrees of freedom of a robot in teleoperation or possibly the tracked motions of the demonstrator's hand. A demonstration is a continuous trajectory d over time t. $\mathcal{D}$ executes a satisfying demonstration to some underlying task objective of interest $\mathcal{T}$ *which is unknown*. So, $\mathcal{T}(d)$ evaluates to TRUE for satisfying demonstrations.

We recover discrete symbolic representations from such continuous demonstrations. The task domain has an input set of predicates $\Pi = \{p_i\}$. Symbolic predicates p_i record region interactions and evaluate to TRUE or FALSE depending upon the point along the demonstration and the location of the regions. For example, consider a predicate associated with *visiting* goal regions which is TRUE when the agent passes through a region. Of note here is *where* the goal region is in the environment and *at what part* of the demonstration the region is traversed. We produce a discrete symbolic trace of predicate evaluations along the continuous demonstration. We find this trace by using an evaluation of all the predicates, $E(d(t)) = v$. A trace over the entire demonstration corresponds to a discrete[1] sequence $\alpha = \{v_1, \cdots, v_{\bar{T}}\},\ v_i \in [\text{TRUE}, \text{FALSE}]^{|\Pi|}$. For a *single demonstration* d in the environment $\mathcal{E}$, interactions with regions produce a trace α.

Specifically, we are interested in an LTL formula $\psi_{\mathcal{T}}$ for representing tasks. $\mathcal{T}(d)$ evaluates to TRUE when the α corresponding to d satisfies $\psi_{\mathcal{T}}$. Two formulae can be compared by computing the difference between sets of satisfying traces using Jaccard distance (intersection over union).

Multiple demonstrations can be obtained for the same task objective by querying the demonstrator multiple times. An **experiment** constitutes each such query with the i^{th} experiment involving $\mathcal{E}_i, d_i, \alpha_i$.

Task Specification Inference Problem: The problem of inferring the task specification that demonstrations are satisfying over multiple experiments.

Input: The method takes as input the symbolic predicates Π, evaluation function E, and a human demonstrator $\mathcal{D}$. A sequence of N experiments generates a sequence of demonstrations $(d_i)_{i\in[1,\cdots,N]}$ in environments $(\mathcal{E}_i)_{i\in[1,\cdots,N]}$.

Output: An estimate of the task specification ψ_{inferred}.

Objective: Reduce the error between ψ_{inferred} and $\psi_{\mathcal{T}}$ and reduce the number of experiments required, N.

IV. Foundations

This section outlines the underlying probabilistic inference problem. Note that we will use the convention of boldening to distinguish distributions and domains $\boldsymbol{\psi}$ from samples ψ. Given a domain of possible formulae $\boldsymbol{\psi}$, any probabilistic model that takes a prior $P(\boldsymbol{\psi}|d_0, \ldots, d_{i-1})$ and demonstration d_i and outputs a posterior $P(\boldsymbol{\psi}|d_0, \ldots, d_{i-1}, d_i)$ can be used by our method. We use the inference procedures proposed in state-of-the-art work [9]. Since the space of LTL formulas is computationally prohibitive for inference, previous work [9] has proposed a subset of this space following the template,

$$\boldsymbol{\psi} = \boldsymbol{\psi_{obs}} \wedge \boldsymbol{\psi_{goals}} \wedge \boldsymbol{\psi_{order}}. \tag{1}$$

This comprises three components: (1) **Global constraints** $\boldsymbol{\psi_{obs}}$ represent the set of propositions, or predicate-variable evaluations, that must *always* hold TRUE. (2) **Eventual goals** $\boldsymbol{\psi_{goals}}$ represent the set of predicate-variable evaluations that must hold TRUE at *some point temporally* over the demonstration. (3) **Temporal order** $\boldsymbol{\psi_{order}}$ represent the temporal ordering constraints among the predicate evaluations.

A Bayesian approach to inference lets us continually update beliefs in light of new demonstrations. Further, it provides significantly more information than the maximum a posteriori (MAP) estimates alone. At the heart of inference lies the use of Bayes' rule in this context,

$$P(\boldsymbol{\psi}|d_0, d_1, \ldots, d_i) = \frac{P(d_i|\boldsymbol{\psi})P(\boldsymbol{\psi}|d_0, d_1, \ldots, d_{i-1})}{\sum_{\psi\in\boldsymbol{\psi}} P(\psi)P(d_i|\psi)} \tag{2}$$

To make the problem tractable, the inference [9] assumes $\boldsymbol{\psi_{obs}}, \boldsymbol{\psi_{goals}}$ and $\boldsymbol{\psi_{order}}$ can be inferred simultaneously.

Priors: For $\boldsymbol{\psi_{obs}}$ and $\boldsymbol{\psi_{goals}}$, priors are constructed using Bernoulli trials for each predicate-variable evaluation in the discrete domain. For $\boldsymbol{\psi_{order}}$, sampling priors is equivalent to sampling forests of DAGs of temporal constraints [32].

Likelihood: Previous work [9] specified $P(d_i|\psi)$ in terms of the number of constraints $|\psi|$ and some small ϵ,

$$\frac{P(d_i|\psi_x)}{P(d_i|\psi_y)} = \begin{cases} \frac{2^{|\psi_x|}}{2^{|\psi_y|}} & \text{if } d_i \models \psi_y \\ \frac{2^{|\psi_x|}}{\epsilon} & \text{if } d_i \not\models \psi_y \end{cases} \tag{3}$$

Baseline Inference: Previous work [9] used the tools set up in this section within a high-level loop visually outlined in Fig 1 (bottom left) to obtain an estimate of ψ_{inferred}. A Bayesian update to obtain the posterior is done via Markov Chain Monte Carlo implemented in PyMC3 [33]. The ψ_{inferred} is a conjunction of MAP estimates for each distribution, as in Eq. 1.

$$\psi_{\text{inferred}} = \text{MAP}(\boldsymbol{\psi_{obs}}) \wedge \text{MAP}(\boldsymbol{\psi_{goals}}) \wedge \text{MAP}(\boldsymbol{\psi_{order}}).$$

The discussion thus far formulates the necessary inference tools that can recover the task specification given a sequence of N demonstrations. What is conspicuously missing is any consideration of the environment or a way to get more useful demonstrations, both of which affects the experiments.

V. Method

The current section outlines the key methodological contributions of the current work. Fig 1 (bottom right) outlines the broad structure of the proposed framework. The high-level contribution is the introduction of *environment design* over the baseline inference framework. The information obtained from inference, $P(\boldsymbol{\psi}|d_0, \ldots, d_{i-1})$, is used in the design stage to come up with an environment $\mathcal{E}_i$ that can aid inference in the i^{th} step. The primary hypothesis will follow.

[1]In our implementation we consider the discrete instants at which the predicate evaluations change.

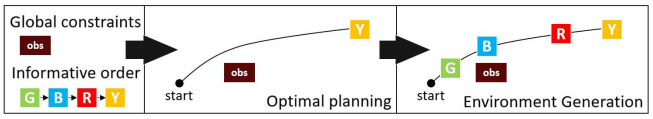

Fig. 2. Environment generation steps in an example spatial task with visit() predicates over obstacles and goal regions. (*Left:*) The estimates of global constraints, and ordering of likely eventual goals. (*Middle:*) A start, obstacle region, and last goal are randomly placed in the environment and optimally solved. (*Right:*) The other goals placed along optimal solution.

Hypothesis [Informative Environment Design]: *Varying the environment, $\mathcal{E}_i$, across demonstrations $d_1, \cdots d_N$ by accounting for the expected utility of the demonstrations in environments can yield improvements in terms of reducing the number of experiments N and the error between ψ_{inferred} and the unknown ground truth task $\psi_{\mathcal{T}}$.*

This problem can be framed as a search over all possible environments, that maximizes an expected utility function based on the posterior after $i-1$ demonstrations. This is referred to as *Bayesian experiment design (BED)* [31]. Designing an experiment can require an iteration over *all possible demonstrations* for *each possible experiment* which is computationally intractable. Consider the case of a 800 sq. pixel interface with 6 regions and a 100px discretization. This maps to 5.3×10^{10} possible environments. *A reduction to a simpler search space is essential.*

First we describe a trace α^* which maximizes a utility. Starting from the *previous* formula estimate $\boldsymbol{\psi}$ after $i-1$ demonstrations, α^* and the formula set $\boldsymbol{\psi}_{\alpha^*}$ provides the highest utility, for e.g., information gain, demonstration diversity, etc. We then need an environment from α^*.

We assume that an optimal planner is a useful model of a (near-)optimal demonstrator satisfying $\psi_{\mathcal{T}}$ to obtain a demonstration $d \approx \text{optPlan}(\mathcal{E}, \psi_{\mathcal{T}})$.

For example, in a spatial task setting we expect human demonstrators to provide short satisfying paths (reducing the Euclidean path length). This allows us to model the human demonstrations with (near-)optimal solutions for the formula. Also, if we start from a demonstration, we can generate environments where the demonstration is optimal. Now, with the optimal demonstrator as our model, we can narrow down an environment of interest as follows given α^*:

$$\mathcal{E}^* \mid d^* = \text{optPlan}(\mathcal{E}^*, \psi)\ \forall \psi \in \boldsymbol{\psi}_{\alpha^*},\ \alpha^* = E(d^*) \quad (4)$$

A. Efficient Environment Generation

Fig 2 outlines how we reduce the search space of all possible environments and demonstrations to the following:
(1) Estimate global constraints and eventual goals
(2) Identify informative sequence of eventual goals
(3) Generate an environment which is designed around the optimal traversal of the informative goal sequence.

If we have an ordering of these regions (Fig 2 left) we can design a corresponding environment (Fig 2 middle, right). We first need estimates of $\boldsymbol{\psi_{obs}}$ and $\boldsymbol{\psi_{goals}}$. The remaining required predicate-variable evaluations need to happen in some order ($\boldsymbol{\psi_{order}}$) and involves k likely eventual goals.

Informative Goal Permutation: We start from the current estimate of the ordering constraints ($\boldsymbol{\psi_{order}}$) given previous demonstrations. Some pairwise ordering constraints between eventual goal predicates will be estimated to be more likely (supported) than others in $\boldsymbol{\psi_{order}}$. A permutation of eventual goals is likewise some ordering of these predicates. We can assign higher utility to permutations that contradict highly supported pairwise ordering constraints. Since highly supported pairwise orderings were likely previously observed, the higher utility permutation is expected to be one that has likely *not been observed yet*. Intuitively, an environment designed around such a permutation incentivizes a (near-) optimal demonstration that is *distinct from the previous demonstrations* and challenges the current ordering estimate the most. This promotes informative, diverse demonstrations.

Optimal Planning for Environment Generation: Here we provide a way to address Eq 4 and generate such an environment. Consider that the order of k likely eventual goal predicates correspond to a sequence of regions $\mathcal{R}_1 \cdots \mathcal{R}_k$. First, all the global constraint regions can be placed in the environment randomly. Then a start of the demonstration is connected using optPlan to a sampled location of the last region $\mathcal{R}_k$. This demonstration is (near-)optimal and crucially, *stays (near-)optimal for $\mathcal{R}_1 \cdots \mathcal{R}_k$ in a new environment* where the remaining regions $\mathcal{R}_1 \cdots \mathcal{R}_{k-1}$ are sampled along the demonstration *in order* (see Fig. 2).

Algorithm 1: INFORMEDINFERENCE

input : demonstrator $\mathcal{D}$, predicates Π, evaluation E
output: LTL formula ψ_{inferred}

```
/* priors sampled as in [9]                                   */
ψ_goals, ψ_obs, ψ_order ← priorDistribution()
while not terminate do
    if notConfident(ψ_obs, ψ_goals) then
        E ← randomEnv(ψ_goals, ψ_obs)
    /* generate informative environments                      */
    else
        ψ̄_goals ← informedGoalSeq(ψ_order)
        E ← informedEnv(ψ̄_goals, MAP(ψ_obs))
    /* query demonstrator & update estimate                   */
    d ← getDemonstration(D, E)
    BayesianUpdate(ψ_goals, ψ_obs, ψ_order, trace(d, E))
return MAP(ψ_obs) ∧ MAP(ψ_goals) ∧ MAP(ψ_order)
```

B. Algorithm

Algorithms 1, 2, and 3 outline the steps in our method. Algorithm 1 is the high-level loop which designs an environment (lines 3-7), queries the demonstrator using experiments (line 8) to obtain demonstrations, and updates the distributions (line 9). The output task specification is a combination of the MAP estimates of the temporal formula distributions. The process begins with random environment design when the discrimination of global and eventual goal constraints shows low confidence (high entropy). Once the likely goals are identified informed environment design is used.

Algorithm 2 generates an informed sequence of eventual goals. It takes as input the ordering constraint distribution and iterates over the domain to score the support of each constraint-edge in each formula within a priority queue

Algorithm 2: INFORMEDGOALSEQ

input : $\boldsymbol{\psi_{order}}$
output: permutation over likely *eventual goals*

1 $Q \leftarrow \emptyset$; $\mathcal{C} \leftarrow \emptyset$; $c \leftarrow \emptyset$
/* populate priority queue of constraints */
2 **for** $\psi \in \text{domain}(\boldsymbol{\psi_{order}})$ **do**
3 **for** $e \in \text{getConstraintEdges}(\psi)$ **do**
4 $Q \leftarrow \text{updateQueue}(e, \text{score}(e))$
/* decide which constraints to *challenge* */
5 **while not** $\text{contradiction}(\mathcal{C} \cup \text{reverse}(c))$ **and** $\text{random}(0,1) < \Gamma_{\text{inform}}$ **do**
6 $\mathcal{C} \leftarrow \mathcal{C} \cup \text{reverse}(c)$
7 $c \leftarrow \text{popMostSupportedConstraint}(Q)$
8 **return** $\text{randomSatisfyingPermutation}(\mathcal{C})$

Algorithm 3: INFORMEDENV

input : permutation $\bar{\psi}_{goals} = (g_1, \cdots g_k), \psi_{obs}$
output: environment $\mathcal{E}$

1 **do**
2 $\{x_{obs}\} \leftarrow \text{sampleLocations}(\psi_{\text{obs}})$
3 $x_s \leftarrow \text{sampleDemoStart}()$
/* sample last goal location */
4 $x_g^k \leftarrow \text{sampleLocation}(g_k)$
/* optimal satisfying path */
5 $d \leftarrow \text{optPlan}(x_s, x_g^k, \{x_{obs}\})$
/* sample intermediate goals along path */
6 **for** $t \leftarrow 0, \quad i \in (1, \cdots, k-1)$ **do**
7 $t \leftarrow \text{random}(t, 1); \quad x_g^i \leftarrow d(t)$
/* create environment and add noise */
8 $\mathcal{E} \leftarrow \text{update}(x_s, \{x_g^1, \cdots x_g^k\}, \{x_{obs}\})$
9 $\mathcal{E} \leftarrow \text{addNoise}()$
10 **while not** $\text{isValid}(\mathcal{E})$
11 **return** $\mathcal{E}$

(Q). The most supported edges will be our candidates for *challenging*. We go down the priority queue (lines 5-7) and try building a temporal constraint by adding the challenged (reverse) edge. The process stops when a contradiction is reached (i.e., the reverse edge violates previously added constraints) or an exploration dropout (Γ_{inform}) is triggered. The output is a goal permutation satisfying constraints in $\mathcal{C}$.

Algorithm 3 uses the permutation of goals to construct an environment. The global constraint regions are first located in the scene. The start of the demonstration is sampled, and the final goal location is sampled. An optimal planner is invoked for the planning problem to connect the start to the final goal while respecting the global constraints. The choice of the planner depends on the domain [13], [14], [34]. The other goals $g_1 \cdots g_{k-1}$ are placed at sampled locations along the optimal path in order (lines 6-7). The environment is updated with these locations for regions (line 8). Line 9 accounts for some noise that is added to the locations. The process continues till the assignment of the locations creates a valid environment. This validity depends on the domain (for instance overlaps between regions might be not allowed).

TABLE I
TASK OBJECTIVE $\psi_{\mathcal{T}}$

ψ_{obs} = $\square(\neg\texttt{maroon}_1 \wedge \neg\texttt{maroon}_2)$
ψ_{goals} = $\diamond(\texttt{red} \wedge \texttt{blue} \wedge \texttt{green} \wedge \texttt{yellow})$

ID	ψ_{order}	Constraint DAG
1	$\neg\texttt{blue} \ \cup \ \texttt{red} \wedge \neg\texttt{green} \ \cup \ \texttt{blue} \wedge \neg\texttt{yellow} \cup \texttt{green}$	**R→B→G→Y**
2	$\top$	no ordering
3	$\neg\texttt{green} \cup \texttt{blue}$	**B→G**
4	$\neg\texttt{green} \cup \texttt{blue} \wedge \neg\texttt{yellow} \cup \texttt{blue}$	**Y←B→G**

VI. HUMAN STUDY RESULTS

This section outlines details of the human experiments and results that demonstrate the efficacy of our proposed method.

Human Participants: We recruited **20** participants from a pool of university students with low familiarity to robotics and temporal logics, assessed via a pre-study questionnaire. The participants were divided into 4 equal groups and were presented standardized instructions across experiments.

Study Interface: We designed a 2-D interface using PyGame where a user can freely move around a 'robot' (Fig 3) using the keyboard. The interface during the study presented **6 regions**, split between 4 differently colored diamonds (red, blue, green, yellow), and 2 maroon rectangles. For each of these objects, a predicate, *visit()* evaluates to true if the robot intersects with the region. Such a 2-D interface is a standard benchmark in state-of-the-art [9]–[11] exploring LTL.

Study Conditions: Each of the 4 groups of 5 participants evaluated a different LTL formulae varying in temporal constraints involving the colored regions (Table I). Each formula was converted to natural language instructions (like Fig 3 left). Participants for a formula were shown identical instructions. The trace data was recorded for each trial. Each participant was shown 10 different environments for 3 different underlying methods each. This provided a total of **600** human demonstrations across all the trials. The order of the presented methods were randomized. Between each set of 10 experiments the participants were given a 60s break. The participants were given an anonymized survey about their background, understanding of the task, and qualitative experience with the interface at the end of the protocol.

Methods and Metrics: We benchmark the proposed framework (*informed*) against two comparison points, the *baseline* [9] where the environment is not affected across demonstrations, and a *random* environment generation strategy. During each run of 10 experiments (participant, method, formula) we recorded the MAP estimate after each additional demonstration. The Jaccard distance error from the ground truth is reported as averages for each formula in Fig 4.

Analysis: *Formula 1 (Fig 4 top left):* The first formula is completely constraining, i.e. there is a unique satisfying trace. Trivially, all three cases perform well since the space of valid demonstrations is uniquely restricted.

Formula 2 (Fig 4 top right): Formula 2 covers the opposite case, where there are no temporal constraints placed upon the task. In the baseline case, the participants tend to provide

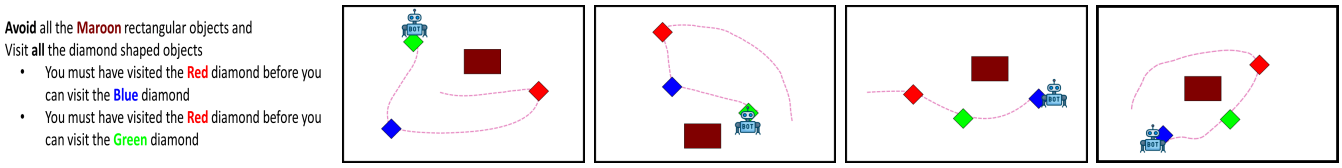

Fig. 3. A motivating example of the interface used in the human study. (Left:) Instructions specifying maroon obstacles, and ordering constraints for red(**R**), blue(**B**), and green(**G**) diamonds. (Right four:) Our approach over 4 experiment examples with the snapshot showing the path (dotted pink) and the position of the robot at the end of the demonstration. The first is a random environment. The right three generate environments that a) are confident about maroon being global constraints, and b) place an informative permutation of goals (**GBR**, **RGB**, **BGR** respectively) along an optimal path. Consider the second experiment. The current estimate supports **B** after **R** from the previous demonstration. The second environment is optimized to incentivize a demonstration with **GBR**. The fact that the demonstrator still chooses **R** before **B** is informative. Experiment design effectively makes inference converge.

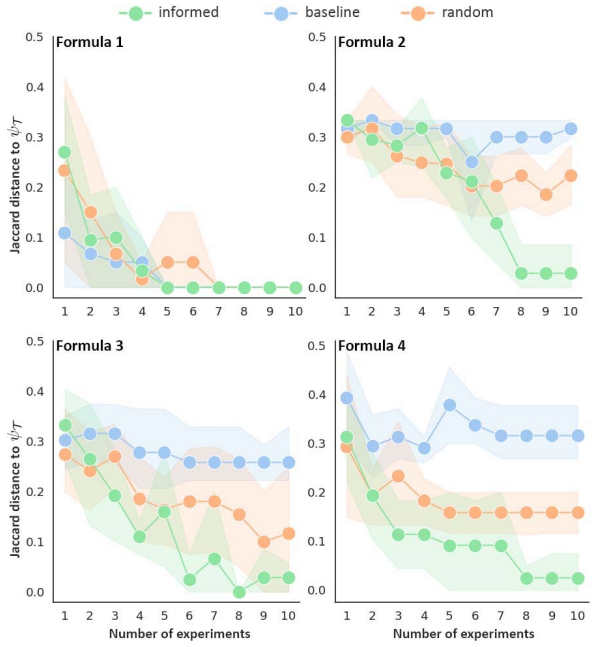

Fig. 4. Benchmarking results for 4 different task specification formulae. The X-axis displays the number of demonstrations, while the Y-axis represents the error from the true formula. Each plot is averaged over the 5 participants assigned to the each formula. The 90% confidence interval is also visualized. *Informed (ours)* consistently shows faster convergence to an estimated formula with low error.

very similar demonstrations - this infers an over-constraining ordering. The random case performs significantly better. Our method outperforms both the baselines by incentivizing the demonstrations that 'challenge' existing formula hypothesis

Formula 3 (Fig 4 bottom left): Next, we consider a formula with a single temporal constraint that blue must be visited before green. Here too the baseline fails to converge, while random has slow progress. The proposed method rapidly converges within 10 demonstrations.

Formula 4 (Fig 4 bottom right): Formula 4 induces an additional constraint to formula 3, i.e. *blue must be visited before both green and yellow*. As earlier the performance benefits are evident.

Note: Non-expert demonstrators can provide erroneous constraint-violating demonstrations. Despite associated fluctuations to the estimation process, our method converges more accurately to task specifications than the baselines.

Conclusions: The hypothesis introduced in this work that environment design can lead to efficient inference of task specification is validated by the empirical performance. The recorded data further supports the hypothesis that humans are (near-)optimal demonstrators of task specifications in the category of problems studied.

Dataset: The data from 600 human demonstrations is also shared in a dataset [2] that contains for each experiment a) a trace of symbolic predicate evaluations, b) the time taken by the participant, c) the ground truth, d) the estimate of the formula, e) error to the ground truth, alongside the anonymized survey responses.

VII. Discussion

We have demonstrated in this work that experiment design yields significant improvement in terms of reducing demonstration effort for non-expert demonstrators and inferring the task specifications with low errors. The work leveraged an optimal planner as a model for demonstrations to efficiently generate informative environments. Our hypothesis was validated in a human study with non-expert participants where data was recorded over 600 experiments with 20 non-expert participants. Furthermore, the dataset of the recorded trials is also publicly shared. While the experiments conducted in this work were limited to a 2D interface, the formulation is broadly applicable and lends itself to more sophisticated applications involving physical robots and interactions.

Beyond 2D, the proposed method remains efficient as long as a (near-)optimal planner is a good model for the human demonstrator. For instance, consider a tabletop problem where the planar coordinates of objects and the projection of the demonstrator's hand can resemble the 2-D interface shown. The choice of accurate distances and costs to reasonably model a demonstrator will be studied in future work. A compelling future direction to explore are human-robot interaction challenges for a robot to actively manipulate environments for gathering useful demonstrations.

Additionally, there are improvements that could be explored in the underlying baseline inference framework [9] that can sample and explore a more general class of temporal formulae. The connections between demonstrator optimality conditions [11] to the proposed process can also be explored for more general task domains. The work carves out a concrete improvement in the right direction towards enabling efficient interactions between intelligent systems and non-expert humans to infer tasks in collaborative applications.

[2]https://github.com/KavrakiLab/sobti2023-icra-data

REFERENCES

[1] M. Ghallab, A. Howe, C. Knoblock, D. Mcdermott, A. Ram, M. Veloso, D. Weld, and D. Wilkins, "PDDL—the planning domain definition language," *AIPS*, 1998.

[2] C. Daniel, M. Viering, J. Metz, O. Kroemer, and J. Peters, "Active reward learning," in *Proceedings of Robotics: Science and Systems*, 2014.

[3] M. Colledanchise and P. Ögren, *Behavior trees in robotics and ai: an introduction*. CRC Press, 2018.

[4] J. D. Hernández, S. Sobti, A. Sciola, M. Moll, and L. E. Kavraki, "Increasing robot autonomy via motion planning and an augmented reality interface," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 1017–1023, 2020.

[5] A. Pnueli, "The temporal logic of programs," in *18Th Annual Symposium on Foundations of Computer Science (Sfcs 1977)*. IEEE, 1977.

[6] E. Plaku and S. Karaman, "Motion planning with temporal-logic specifications: progress and challenges," *AI Communications*, vol. 29, no. 1, pp. 151–162, 2016.

[7] C. Innes and R. Ramamoorthy, "Elaborating on learned demonstrations with temporal logic specifications," in *Proceedings of Robotics: Science and Systems*, 2020.

[8] D. Gujarathi and I. Saha, "Mt*: multi-robot path planning for temporal logic specifications," *ArXiv Preprint ArXiv:2103.02821*, 2021.

[9] A. Shah, P. Kamath, J. A. Shah, and S. Li, "Bayesian inference of temporal task specifications from demonstrations," in *Advances in Neural Information Processing Systems*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds., vol. 31. Curran Associates, Inc., 2018.

[10] M. Vazquez-Chanlatte, S. Jha, A. Tiwari, M. K. Ho, and S. Seshia, "Learning task specifications from demonstrations," *Advances in Neural Information Processing Systems*, vol. 31, 2018.

[11] G. Chou, N. Ozay, and D. Berenson, "Explaining multi-stage tasks by learning temporal logic formulas from suboptimal demonstrations," in *Proceedings of Robotics: Science and Systems*, 2020.

[12] C. Baier and J.-P. Katoen, *Principles of model checking*. MIT press, 2008.

[13] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. Overmars, "Probabilistic Roadmaps for path planning in high-dimensional configuration spaces," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, 1996.

[14] S. M. LaValle and J. J. Kuffner Jr, "Randomized Kinodynamic planning," *The International Journal of Robotics Research*, vol. 20, no. 5, pp. 378–400, 2001.

[15] A. Bhatia, L. E. Kavraki, and M. Y. Vardi, "Sampling-based motion planning with temporal goals," in *Proceedings of the IEEE International Conference on Robotics and Automation*. IEEE, 2010.

[16] K. He, M. Lahijanian, L. E. Kavraki, and M. Y. Vardi, "Towards manipulation planning with temporal logic specifications," in *Proceedings of the IEEE International Conference on Robotics and Automation*, 2015, pp. 346–352.

[17] N. T. Dantam, Z. K. Kingston, S. Chaudhuri, and L. E. Kavraki, "An incremental constraint-based framework for task and motion planning," *The International Journal of Robotics Research*, vol. 37, no. 10, pp. 1134–1151, 2018.

[18] H. Ravichandar, A. S. Polydoros, S. Chernova, and A. Billard, "Recent advances in robot learning from demonstration," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 3, no. 1, pp. 297–330, 2020.

[19] S. Sobti, R. Shome, S. Chaudhuri, and L. E. Kavraki, "A sampling-based motion planning framework for complex motor actions," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2021, pp. 6928–6934.

[20] S. Malik, U. Anwar, A. Aghasi, and A. Ahmed, "Inverse constrained reinforcement learning," in *International Conference on Machine Learning*. PMLR, 2021, pp. 7390–7399.

[21] T. Arnold and D. Kasenberg, "Value alignment or misalignment - "what will keep systems accountable?" in *AAAI Workshop on AI, Ethics, and Society*, 2017.

[22] W. Erlhagen and E. Bicho, "A dynamic neural field approach to natural and efficient human-robot collaboration," *Neural Fields: Theory and Applications*, pp. 341–365, 03 2014.

[23] J. Kim, C. Muise, A. Shah, S. Agarwal, and J. Shah, "Bayesian inference of linear temporal logic specifications for contrastive explanations," in *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence*. International Joint Conferences on Artificial Intelligence Organization, 2019, pp. 5591–5598.

[24] A. Camacho and S. A. McIlraith, "Learning interpretable models expressed in linear temporal logic," in *Proceedings of the International Conference on Automated Planning and Scheduling*, vol. 29, 2019, pp. 621–630.

[25] R. Roy, D. Fisman, and D. Neider, "Learning interpretable models in the property specification language," in *Twenty-Ninth International Joint Conference on Artificial Intelligence*. IJCAI, 2020, pp. 2213–2219.

[26] D. Kasenberg and M. Scheutz, "Interpretable apprenticeship learning with temporal logic specifications," in *2017 IEEE 56Th Annual Conference on Decision and Control (CDC)*. IEEE, 2017, pp. 4914–4921.

[27] M. Cakmak and A. L. Thomaz, "Designing robot learners that ask good questions," in *Proceedings of the Seventh Annual ACM/IEEE International Conference on Human-Robot Interaction - HRI '12*, 2012.

[28] A. Bobu, M. Wiggert, C. Tomlin, and A. D. Dragan, "Feature expansive reward learning: rethinking human input," *Proceedings of the 2021 ACM/IEEE International Conference on Human-Robot Interaction*, pp. 216–224, 2021.

[29] J. Kulick, M. Toussaint, T. Lang, and M. Lopes, "Active learning for teaching a robot grounded relational symbols," *IJCAI International Joint Conference on Artificial Intelligence*, 01 2013.

[30] A. J. Shah and J. A. Shah, "Interactive robot training for non-Markov tasks," *ArXiv*, vol. abs/2003.02232, 2020.

[31] G. Terejanu, R. R. Upadhyay, and K. Miki, "Bayesian experimental design for the active nitridation of graphite by atomic nitrogen," *Experimental Thermal and Fluid Science*, vol. 36, pp. 178–193, 2012.

[32] D. J. Aldous, "Exchangeability and related topics," in *École D'ÉtÉ De ProbabilitÉs De Saint-Flour XIII — 1983*, P. L. Hennequin, Ed. Berlin, Heidelberg: Springer Berlin Heidelberg, 1985, pp. 1–198.

[33] J. Salvatier, T. V. Wiecki, and C. Fonnesbeck, "Probabilistic programming in python using PyMC3," *PeerJ Computer Science*, vol. 2, p. e55, 2016.

[34] P. E. Hart, N. J. Nilsson, and B. Raphael, "A formal basis for the heuristic determination of minimum cost paths," *IEEE Transactions on Systems Science and Cybernetics*, vol. 4, no. 2, pp. 100–107, 1968.