Human-in-the-Loop Synthesis for Partially Observable Markov Decision Processes

Steven Carr[1] Nils Jansen[2] Ralf Wimmer[3] Jie Fu[4] Ufuk Topcu[1]

***Abstract*— We study planning problems where autonomous agents operate inside environments that are subject to uncertainties and not fully observable. Partially observable Markov decision processes (POMDPs) are a natural formal model to capture such problems. Because of the potentially huge or even infinite belief space in POMDPs, synthesis with safety guarantees is, in general, computationally intractable. We propose an approach that aims to circumvent this difficulty: in scenarios that can be partially or fully simulated in a virtual environment, we actively integrate a human user to control an agent. While the user repeatedly tries to safely guide the agent in the simulation, we collect data from the human input. Via behavior cloning, we translate the data into a strategy for the POMDP. The strategy resolves all nondeterminism and non-observability of the POMDP, resulting in a discrete-time Markov chain (MC). The efficient verification of this MC gives quantitative insights into the quality of the inferred human strategy by proving or disproving given system specifications. For the case that the quality of the strategy is not sufficient, we propose a refinement method using counterexamples presented to the human. Experiments show that by including humans into the POMDP verification loop we improve the state of the art by orders of magnitude in terms of scalability.**

I. Introduction

We aim at providing guarantees for planning scenarios given by dynamical systems with uncertainties and partial observability. In particular, we want to compute a *strategy* for an agent that ensures certain desired behavior [15].

A popular formal model for planning subject to stochastic behavior are Markov decision processes (MDPs). An MDP is a nondeterministic model in which the agent chooses to perform an action under full knowledge of the environment it is operating in. The outcome of the action is a probability distribution over the system states. Many applications, however, allow only *partial observability* of the current system state [20], [40], [45]. For such applications, MDPs are extended to *partially observable Markov decision processes* (POMDPs). While the agent acts within the environment, it encounters certain *observations*, according to which it can infer the likelihood of the system being in a certain state. This likelihood is called the *belief state*. Executing an action leads to an update of the belief state according to new observations. The belief state together with the update function form a (possibly infinite) MDP, commonly referred to as the underlying *belief MDP* [35].

[1]The University of Texas at Austin, USA
[2]Radboud University, Nijmegen, The Netherlands, n.jansen@science.ru.nl
[3]Albert-Ludwigs-Universität Freiburg, Freiburg im Breisgau, Germany
[4]Worcester Polytechnic Institute (WPI), USA

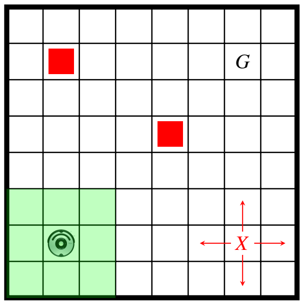

Fig. 1: Example Gridworld Environment. *Features* are (1) an agent with restricted range of vision (green area), (2) static and randomly moving obstacles (red), and (3) a goal area G.

As a motivating example, take a motion planning scenario where we want to devise a strategy for an autonomous agent accounting for both randomly moving and static obstacles. *Observation* of these obstacles is only possible within a restricted field of vision like in Fig. 1. The strategy shall *provably* ensure a *safe* traversal to the goal area with a certain high probability. On top of that, the expected *performance* of the agent according to the strategy shall encompass taking the quickest possible route. These requirements amount to having quantitative reachability specifications like "The probability to reach the goal area without crashing into obstacles is at least $90\,\%$" and expected cost specifications like "The expected number of steps to reach the goal area is at most 10".

Quantitative verification techniques like *probabilistic model checking* (PMC) [21] provide strategies inducing guarantees on such specifications. PRISM [25] or Storm [12] employ efficient methods for finite MDPs, while POMDP verification – as implemented in a PRISM prototype [29] – generates a large, potentially infinite, belief MDP, and is intractable even for rather small instances. So-called *point-based methods* [30], [38] employ sampling of belief states. They usually have slightly better scalability than verification, but there is no guarantee that a strategy provably adheres to specifications.

We discuss two typical problems in POMDP verification.

1) For applications following the aforementioned example, verification takes any specific position of obstacles and previous decisions into account. More generally, strategies inducing optimal values are computed by assessment of the full belief MDP [35].
2) Infinite horizon problems may require a strategy to have infinite memory. However, *randomization* over possible choices can often trade off memory [9]. The intuition is that *deterministic* choices at a certain state may need to

vary depending on previous decisions and observations. Allowing for a probability distribution over the choices – relaxing determinism – is often sufficient to capture the necessary variability in the decisions. As also finite memory can be encoded into a POMDP by extending the state space, randomization then supersedes infinite memory for many cases [2], [19].

Here, we propose to make active use of humans' *power of cognition* to (1) achieve an implicit abstraction of the belief space and (2) capture memory-based decisions by randomization over choices. We translate POMDP planning scenarios into virtual environments where a human can actively operate an agent. In a nutshell, we create a game that the human plays to achieve a goal akin to the specifications for the POMDP scenario.

This game captures a *family of concrete scenarios*, for instance varying in characteristics like the agent's starting position or obstacle distribution. We collect data about the human actions from a number of different scenarios to build a *training set*. With Hoeffding's inequality [46], we statistically infer how many human inputs are needed until further scenarios won't change the likelihoods of choices. Using behavior cloning techniques from Learning-from-Demonstration (LfD) [3], [34], we cast the training set into a *strategy* for a POMDP that captures one specific scenario. Such a strategy fully resolves the nondeterminism and partial observability, resulting in a discrete-time Markov chain (MC). PMC for this MC is efficient [25] and proves or disproves the satisfaction of specifications for the computed strategy.

A human implicitly bases their decisions on experiences over time, i. e., on memory [11]. We collect likelihoods of decisions and trade off such implicit memory by translating these likelihoods into randomization over decisions. In general, randomization plays a central role for describing human behavior in cognitive science. Take for instance [23] where human behavior is related to quantifying the trade-off between various decisions in Bayesian decision theory.

Formally, the method yields a *randomized* strategy for the POMDP that may be extended with a finite memory structure. Note that computing such a strategy is already NP-hard, SQRT-SUM-hard, and in PSPACE [41], justifying the usage of heuristic and approximative methods.

Naturally, such a heuristic procedure comprises no means of optimality. However, employing a refinement technique incorporating stochastic counterexamples [1], [17] enables to pointedly immerse the human into critical situations to gather more specific data. In addition, we employ *bounds on the optimal performance* of an agent derived from the underlying MDP. This delivers an indication whether no further improvement is possible by the human.

Besides simple motion planning, possible applications include self-driving cars [13], autonomous trading agents in the stock market [42], or service robots [22].

We implemented this synthesis cycle with humans in the loop within a prototype employing efficient verification. The results are very promising as both PRISM and point-based solvers are outperformed by orders of magnitude both regarding running time and the size of tractable models.

Our approach is inherently *correct*, as any computed strategy is verified for the specifications.

Related Work: Closest to our work is [10], where deep reinforcement learning employs human feedback. In [32], a robot in a partially observable motion planning scenario can request human input to resolve the belief space. The availability of a human is modeled as a stochastic sensor. Similarly, *oracular POMDPs* [4] capture scenarios where a human is always available as an oracle. The latter two approaches do not discuss how to actually include a human in the scenarios. The major difference of all approaches listed above in comparison to our method is that by employing verification of inferred strategies, we obtain hard guarantees on the safety or performance.

Verification problems for POMDPs and their decidability have been studied in [7]. [44] investigates abstractions for POMDP motion planning scenarios formalizing typical human assessments like "the obstacle is either near or far", learning MDP strategies from human behavior in a shared control setting was used in [16]. Finally, various learning-based methods and their (restricted) scalability are discussed in [5].

Structure of the paper: After formalisms in Sect. II, Section III gives a full overview of our methodology. In Sect. IV, we formally discuss randomization and memory for POMDP strategies; after that we introduce strategy generation for our setting together with an extensive example. We describe our experiments and results in Sect. VI.

II. Preliminaries

A *probability distribution* over a finite or countably infinite set X is a function $\mu\colon X \to [0,1] \subseteq \mathbb{R}$ with $\sum_{x\in X}\mu(x) = \mu(X) = 1$. The set of all distributions on X is $\mathit{Distr}(X)$. The support of a distribution μ is $\mathit{supp}(\mu) = \{x \in X \mid \mu(x) > 0\}$.

A. Probabilistic Models

Definition 1 (MDP) *A* Markov decision process *(MDP) M is a tuple $M = (S, s_{\mathrm{I}}, \mathit{Act}, P)$ with a finite (or countably infinite) set S of* states, *an* initial state $s_{\mathrm{I}} \in S$, *a finite set Act of* actions, *and a* probabilistic transition function $P\colon S \times \mathit{Act} \to \mathit{Distr}(S)$.

The *available actions* in $s \in S$ are $\mathit{Act}(s) = \{a \in \mathit{Act} \mid (s,a) \in \mathrm{dom}(P)\}$. We assume the MDP M contains no deadlock states, i. e., $\mathit{Act}(s) \neq \emptyset$ for all $s \in S$. A *discrete-time Markov chain* (MC) is an MDP with $|\mathit{Act}(s)| = 1$ for all $s \in S$.

A *path* (or run) of M is a finite or infinite sequence $\pi = s_0 \xrightarrow{a_0} s_1 \xrightarrow{a_1} \cdots$, where $s_0 = s_{\mathrm{I}}$, $s_i \in S$, $a_i \in \mathit{Act}(s_i)$ The set of (in)finite paths is $\mathsf{Paths}^M_{\mathit{fin}}$ (Paths^M). To define a probability measure for MDPs, *strategies* resolve the nondeterministic choices of actions. Intuitively, at each state a strategy determines a distribution over actions to take. This decision may be based on the *history* of the current path.

Definition 2 (Strategy) *A* strategy σ *for M is a function $\sigma\colon \mathsf{Paths}^M_{\mathit{fin}} \to \mathit{Distr}(\mathit{Act})$ s. t. $\mathit{supp}\big(\sigma(\pi)\big) \subseteq \mathit{Act}\big(\mathit{last}(\pi)\big)$ for all $\pi \in \mathsf{Paths}^M_{\mathit{fin}}$. Σ^M denotes the set of all strategies of M.*

A strategy σ is *memoryless* if $\mathit{last}(\pi) = \mathit{last}(\pi')$ implies $\sigma(\pi) = \sigma(\pi')$ for all $\pi, \pi' \in \mathrm{dom}(\sigma)$. It is *deterministic* if

$\sigma(\pi)$ is a Dirac distribution for all $\pi \in \mathrm{dom}(\sigma)$. A strategy that is not deterministic is *randomized*. Here, we mostly use strategies that are memoryless and randomized, i. e., of the form $\sigma\colon S \to \mathit{Distr}(\mathit{Act})$.

A strategy σ for an MDP resolves all nondeterministic choices, yielding an *induced Markov chain* (MC) M^σ, for which a *probability measure* over the set of infinite paths is defined by the standard cylinder set construction.

Definition 3 (Induced Markov Chain) *Let MDP $M = (S, s_\mathrm{I}, \mathit{Act}, P)$ and strategy $\sigma \in \Sigma^M$. The MC induced by M and σ is given by $M^\sigma = (S, s_\mathrm{I}, \mathit{Act}, P^\sigma)$ where:*

$$P^\sigma(s,s') = \sum_{a \in A(s)} \sigma(s)(a) \cdot P(s,a)(s') \quad \forall s,s' \in S \ .$$

B. Partial Observability

Definition 4 (POMDP) *A* partially observable Markov decision process (POMDP) *is a tuple $D = (M, Z, O)$ such that $M = (S, s_\mathrm{I}, \mathit{Act}, P)$ is the* underlying MDP of *D, Z is a finite set of observations and $O\colon S \to Z$ is the observation function.*

We require that states with the same observations have the same set of enabled actions, i. e., $O(s) = O(s')$ implies $\mathit{Act}(s) = \mathit{Act}(s')$ for all $s, s' \in S$. More general observation functions take the last action into account and provide a distribution over Z. There is a transformation of the general case to the POMDP definition used here that blows up the state space polynomially [8].

Furthermore, let $\Pr(s|z)$ be the probability that given observation $z \in Z$, the state of the POMDP is $s \in S$. We assume a maximum-entropy probability distribution [18] to provide an initial distribution over potential states for an observation z given by $\Pr(s|z) = \frac{1}{|\{s' \in S \mid z = O(s')\}|}$. Vice versa, we set $\Pr(z|s) = 1$ iff $z = O(s)$ and $\Pr(z|s) = 0$ otherwise.

The notion of paths directly transfers from MDPs to POMDPs. We lift the observation function to paths: For a POMDP D and a path $\pi = s_0 \xrightarrow{a_0} s_1 \xrightarrow{a_1} \cdots s_n \in \mathsf{Paths}^M_{\mathit{fin}}$, the associated *observation sequence* is $O(\pi) = O(s_0) \xrightarrow{a_0} O(s_1) \xrightarrow{a_1} \cdots O(s_n)$. Note that several paths in the underlying MDP may yield the same observation sequence. Strategies have to take this restricted observability into account.

Definition 5 *An* observation-based strategy *of POMDP D is a function $\sigma\colon \mathsf{Paths}^M_{\mathit{fin}} \to \mathit{Distr}(\mathit{Act})$ such that σ is a strategy for the underlying MDP and for all paths $\pi, \pi' \in \mathsf{Paths}^M_{\mathit{fin}}$ with $O(\pi) = O(\pi')$ we have $\sigma(\pi) = \sigma(\pi')$. Σ^D_z denotes the set of observation-based strategies for D.*

An observation-based strategy selects actions based on the observations encountered along a path and past actions. Note that applying an observation-based strategy to a POMDP yields an induced MC as in Def. 3 where all nondeterminism and partial observability is resolved. Again, we use memoryless and randomized strategies of the form $\sigma_z\colon Z \to \mathit{Distr}(\mathit{Act})$.

The semantics of a POMDP can be described using a *belief MDP* with an uncountable state space. The idea is that each state of the belief MDP corresponds to a distribution over the states in the POMDP. This distribution is expected to correspond to the probability to be in a specific state based on the observations made so far.

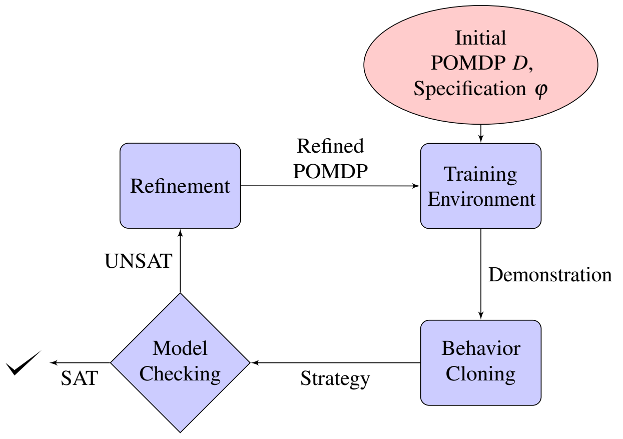

Fig. 2: Workflow of human-in-the-loop (HiL) methodology.

C. Specifications

For a POMDP $D = (M, Z, O)$, a set $G \subseteq S$ of *goal states* and a set $B \subseteq S$ of *bad states*, we consider *quantitative reach-avoid specifications* of the form $\varphi = \mathbb{P}_{\geqslant\lambda}(\neg B \ \mathrm{U}\ G)$. A strategy $\sigma_z \in \Sigma_z$ satisfies this specifications if the probability of reaching a goal state without entering a bad state in between is at least λ in the induced MC, written $D^{\sigma_z} \models \varphi$. We also use similar specifications of the form $\mathrm{EC}_{\leq\kappa}(\neg B \ \mathrm{U}\ G)$, measuring the *expected cost* to safely reach a goal state. For POMDPs, observation-based strategies in their full generality are necessary [33].

Consider a specification φ that is not satisfied by an MC or MDP M. One common definition of a *counterexample* is a (minimal) *subset* $S' \subseteq S$ of the state space such that the MC or sub-MDP induced by S' still violates φ [1]. The intuition is, that by the reduced state space critical parts are highlighted.

III. Methodology

A. Problem Statement

We are given a partially observable planning scenario, which is modeled by a POMDP D, and a specification φ. The POMDP D is one of a *family* of similar planning scenarios, where each concrete scenario can modeled by an individual POMDP. The goal is to compute an observation-based randomized memoryless strategy $\sigma_z \in \Sigma^D_z$ such that $D^{\sigma_z} \models \varphi$.

The general workflow we employ is shown in Fig. 2. Note that we mostly assume a family of POMDP scenarios to train the human strategy, as will be explained in what follows. We now detail the specific parts of the proposed approach.

B. Training Environment

Our setting necessitates that a virtual and interactive environment called *training environment* sufficiently captures the underlying POMDP planning scenarios. The initial training environment can be parameterized for: the size of the environment; the numbers and locations of dynamic obstacles and landmarks; and the location of the goal state.

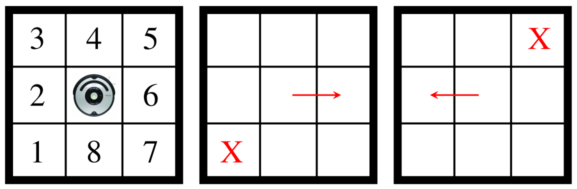

Fig. 3: Possible observations (left) and two observations triggering similar actions.

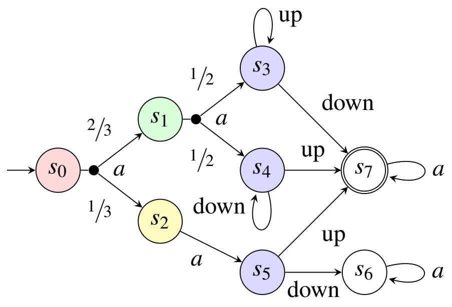

Fig. 4: Randomization vs. memory

Similar classes of problems would require similar initial training environments. For example, an environment may incorporate a small grid with one dynamic obstacle and two landmarks, while the actual POMDP we are interested in needs the same number of dynamic obstacles but may induce a larger grid or add additional landmarks. The goal state location also impacts the type of strategy trained by the human. With a randomized goal location, the human will prioritize obstacle avoidance over minimizing expected cost.

We directly let the human *control* the agent towards a conveyable goal, such as avoiding obstacles while moving to the goal area. We store all data regarding the human control inputs in a *training set*. For a POMDP, this means that at each visited state of the underlying MDP we store the corresponding observation and the human's action choice. We now collect data from several (randomly-generated) environments until statistically the human input will not significantly change the strategy by collecting further data. In fact, the training set contains *likelihoods* of actions.

C. Strategy Generation from Behavior Cloning

We compute an *environment-independent strategy* for the agent by casting the collected data into probability distributions over actions for each observation at each state of the system. Intuitively, the strategy is independent from a concrete environment but compatible with all concrete scenarios the training environment captures. Generally, linear [14] or softmax regression [46] offers a means to interpret the likelihoods over actions into probabilities. Formally, we get an *observation-based strategy* of the POMDP. The computed strategy mimics typical human choices in all possible situations.

So far, such a strategy requires a large training set that needs a long time to be established, as we need human input for all different observations and actions. If we do not have a sufficiently large training set, that is, we have a *lack of data*, the strategy is *underspecified*.

We use data augmentation [24] to *reduce the observation-action space* upon which we train the strategy. Our assumption is that the human acts similarly upon similar observations of the environment. For instance, take two different observations describing that a moving obstacle is to the right border or to the left of the agent's range of vision. While these are in fact different observations, they may trigger similar actions (moving into opposite directions – see Fig. 3) where on the left we see the possible observations for the agent and on the right two observations triggering similar actions (away from the obstacle). Therefore, we define an *equivalence relation* on observations and actions. We then weigh the likelihoods of equivalent actions for each state with equivalent observations and again cast these weighted likelihoods into probability distributions. Summarized, as this method reduces the observation-action space, it also reduces the required size of the training set and the required number of human inputs.

D. Refinement through Model Checking and Counterexamples

We apply the computed strategy to a POMDP for a concrete scenario. As we resolve all nondeterminism and partial observability, the resulting model is an MC. To efficiently verify this MC against the given specification, we employ *probabilistic model checking* using PRISM. For instance, if for this MC the probability of reaching the goal area without bumping into obstacles is above a certain threshold, the computed strategy provably induces this exact probability.

In case PMC reveals that the obtained strategy does not satisfy the requirements, we need to improve the strategy for the specific POMDP we are dealing with. Here, we again take advantage of the human-in-the-loop principle. First, we generate a *counterexample* using, e. g., the techniques described in [17]. Such a counterexample highlights critical parts of the state space of the induced MC. We then immerse the human into critical parts in the virtual environment corresponding to critical states of the specific POMDP. By gathering more data in these apparently critical situations for this scenario we strive to improve the human performance and the quality of the strategy.

IV. Randomized Strategies

Deciding if there is an observation-based strategy for a POMDP satisfying a specification as in Sec. II-C typically requires unbounded memory and is undecidable in general [27]

If we restrict ourselves to the class of memoryless strategies (which decide only depending on the current observation), we need to distinguish two sub-classes: (1) finding an optimal deterministic memoryless strategy is NP-complete [26], (2) finding an optimal randomized memoryless strategy NP-hard, SQRT-SUM-hard, and in PSPACE [41]. From a practical perspective, randomized strategies are much more powerful as one can – to a certain extent – simulate memory by randomization. The following example illustrates this effect and its limitations.

Example 1 *In the POMDP in Fig. 4, observations are defined by colors. The goal is to reach s_7 with maximal probability. The only states with nondeterminism are s_3, s_4, and s_5 (blue). For a memoryless deterministic strategy selecting "up" in all blue states, the optimal value is $2/3$.*

A memoryless randomized strategy can select "up" with probability $0 < p < 1$ and "down" with probability $1-p$ for blue states. Then both from s_3 and s_4, the target states are eventually reached with probability 1 and from s_5 with probability p. Therefore the probability to reach s_7 from the initial state is $2/3 + 1/3p < 1$.

Finally, deterministic strategies with memory can distinguish state s_5 from s_3 and s_4 because their predecessors have different observations. An optimal strategy may select "up" in a blue state if its predecessor is yellow, and otherwise "up" if a blue state has been seen an even number of times and "down" for an odd number, yielding probability 1 to reach s_7.

Summarized, computing randomized memoryless strategies for POMDPs is – while still a hard problem – a powerful alternative to the harder or even undecidable problem of computing strategies with potentially infinite memory.

V. Strategy Generation

We detail the four phases to behavior cloning from human inputs: (1) building a training set, (2) feature-based data augmentation, (3) the initial strategy generation, and (4) refining the initial strategy using counterexamples.

A. Training Set

We first provide a series of randomly-generated sample environments to build a human-based training set. The environments are randomized in size, location and number of obstacles as well as the location of initial and goal states. The training set Ω^E is represented as a function $\Omega^E: Z \times Act \to \mathbb{N}$, where $\Omega^E(z,a) = n_a$ means that n_a is the number of times action a is selected by the human for observation z. The *size* of the training set is given by $|\Omega^E| = \sum_{z\in Z, a\in Act} \Omega^E(z,a)$.

In each sample environment, the human is given a map of the static environment – the locations of static obstacles and goal states – as well as an observation in the current state. This observation may, for instance, refer to the position of a visible obstacle. Moreover, the human is provided with one or more specifications. Proscribing a threshold on the probability of reaching the goal to a human seems unpractical. Instead, to have the human act according to the specifications outlined in Sect. II, we can for instance ask the human to maximize the probability of reaching the goal whilst minimizing an expected cost. In practice, this just means that the human attempts to maximize the probability of reaching the goal without crashing and as quickly as possible. The human observes the obstacles one-step from the agent (see Fig. 3), but is not aware of the agent's precise position or if observed obstacles are static or dynamic. For an unknown initial position, there are two phases [36], [37]:

1) **Exploration:** The human will first try to determine their position while taking advantage of knowledge of the static environment.
2) **Exploitation:** When the human is confident about their current position they will start moving towards the goal.

The human acts on each (randomly generated) concrete scenario until they either reach the goal or crash. We continue collecting data until the human's inputs no longer significantly change the strategy. The statistically-derived minimum required size $|\Omega^E|$ of the initial training set is bound by Hoeffding's inequality [46]. In particular, we derive an upper bound $\varepsilon \in \mathbb{R}$ (with a confidence of $1-\delta$ for $\delta \in [0,1]$) for the difference between (1) a strategy that is independent from further training with other concrete environments and (2) the strategy derived from the training set of size $|\Omega^E|$. The number of samples is bounded by

$$|\Omega^E| \geq \frac{1}{2\varepsilon^2}\left(\ln\frac{2}{\delta}\right). \tag{1}$$

B. Feature Representation

Human input – choices of observation-action pairs – for our simulations has limitations. First, it may not cover the entire observation-space so we may have observations without (sufficient) human choices of actions; the resulting strategy will be *underspecified*. Additionally, many of the observation-action pairs are equivalent in nature since – in our example setting – the tendency for the human's action input is to move away from neighboring obstacles. Similar equivalences may be specified depending on the case study at hand. We introduce a *feature-based representation* to take advantage of such similarities to reduce the required size of a training set.

Consider therefore the gridworld scenario in Fig. 1. Recall that the agent has restricted range of vision, see Fig. 3. The set of positions in the grid $\text{Grid}_x \times \text{Grid}_y \subseteq \mathbb{N} \times \mathbb{N}$ is

$$\text{Pos} = \big\{(x,y) \,\big|\, x \in \{0,\dots,\text{Grid}_x\}, y \in \{0,\dots,\text{Grid}_y\}\big\}.$$

For one dynamic obstacle, an agent state consists of the position $(x_a, y_a) \in \text{Pos}$ of agent a and the position $(x_o, y_o) \in \text{Pos}$ of the dynamic obstacle o, i.e., $s = (x_a, y_a, x_o, y_o) \in \text{Pos} \times \text{Pos}$. The agent's actions $Act = \{(-1,0),(1,0),(0,1),(0,-1)\}$ describe the one-step directions "left", "right", "up", "down". The set B of obstacle positions is $B = \{(x_o,y_o),(x_{l_1},y_{l_1}),\dots,(x_{l_m},y_{l_m}) \mid (x_o,y_o) \in \text{Pos}, (x_{l_i},y_{l_i}) \in \text{Pos}, 1 \leq i \leq m\}$ for dynamic obstacle o and landmarks $l_1,\dots,l_m$.

The *observations* describe the relative position of obstacles with respect to the agent's position, see Fig. 3. We describe these positions by a set of Boolean functions $O_i: S \times 2^{Pos} \to \{0,1\}$ where $S = \text{Pos}_x \times \text{Pos}_y$ is the agent's position and for a visibility distance of 1, O_i is defined for $1 \leq i \leq 8$ by:

$$\begin{aligned}
O_1(s,B) = 1 \text{ iff } & ((x_a-1, y_a-1) \in B) \vee (x_a = 0) \vee (y_a = 0),\\
O_2(s,B) = 1 \text{ iff } & ((x_a-1, y_a) \in B) \vee (x_a = 0),\\
O_3(s,B) = 1 \text{ iff } & ((x_a-1, y_a+1) \in B) \vee (x_a = 0) \vee (y_a = n),\\
O_4(s,B) = 1 \text{ iff } & ((x_a, y_a+1) \in B) \vee (y_a = n),\\
O_5(s,B) = 1 \text{ iff } & ((x_a+1, y_a+1) \in B) \vee (x_a = n) \vee (y_a = n),\\
O_6(s,B) = 1 \text{ iff } & ((x_a+1, y_a) \in B) \vee (x_a = n),\\
O_7(s,B) = 1 \text{ iff } & ((x_a+1, y_a-1) \in B) \vee (x_a = n) \vee (y_a = 0),\\
O_8(s,B) = 1 \text{ iff } & ((x_a, y_a-1) \in B) \vee (y_a = 0).
\end{aligned}$$

Note that for a visibility distance of 2, O_i is defined for $1 \leq i \leq 24$. Consequently, an observation $z = O(s)$ at state s is a vector $z = (z^{(1)}, \dots, z^{(8)}) \in \{0,1\}^8$ with $z^{(i)} = O_i(s,B)$. The observation space $Z = \{z_1, \dots, z_{256}\}$ is the set of all observation vectors.

Providing a human with enough environments to cover the entire observation space is inefficient. [39] To simplify this space, we introduce *action-based features* [31], which capture the short-term human behavior of prioritizing to avoid obstacles for current observations. Particularly, we define features $f\colon Z \times Act \rightarrow \mathbb{N}$. In our example setting we have

$$f_1(z,a) = \sum_{i=1}^{8} z^{(i)} \ ,$$

$$f_2(z,a) = \left| a_x - \sum_{i=1}^{3} z^{(i)} + \sum_{i=5}^{7} z^{(i)} \right| \ ,$$

$$f_3(z,a) = \left| a_y - \sum_{i\in\{1,7,8\}} z^{(i)} + \sum_{i=3}^{5} z^{(i)} \right| \ ,$$

where f_1 describes the number of obstacles in the observations. f_2 and f_3 are the respective x and y directional components of the difference between the motion of the agent's action (a_x and a_y respectively) and position of the obstacles in its observation. Then, the comprised feature function is $f\colon Z \times Act \rightarrow \mathbb{N}^3$ with $f(z,a) = \big(f_1(z,a), f_2(z,a), f_3(z,a)\big)$.

We define a component-wise "equivalence" of observations-action features:

$$f(z_1,a_1) = f(z_2,a_2) \iff \bigwedge_i \big(f_i(z_1,a_1) = f_i(z_2,a_2)\big) \ .$$

In Fig. 3, both observations see an obstacle in the corner of the observable space. For the left-hand case, the obstacle is on the bottom-left and action "right" is taken to avoid it. In the right-hand case, the obstacle is on the top-right and action "left" is taken to avoid it. These observation-action cases are considered equivalent in our feature model.

In developing a strategy for the POMDP, we iterate through the observation-action space $Z \times Act$ and find *feature-equivalent inputs* based on the above criteria. A set of feature-equivalent inputs is then $\hat{F} = \{(z_1,a_1), \dots, (z_k,a_k)\}$ where $f(z_1,a_1) = f(z_k,a_k)$. By using the feature-equivalent inputs we are guaranteed to require less human inputs. The maximum possible size of the equivalent-feature set is $|\hat{F}| \leq \binom{8}{4} = 70$, due to the number of permutations of f_1. So at best our feature method can allow for 70 times fewer inputs. The efficiency gained by the introduction of features is at least $|\hat{F}| \geq 1 + \frac{4}{n}$ for an empty n sized gridworld, the worst possible case. The majority of observations in sparse gridworlds are zero- or single-obstacle observations with an average efficiency of approximately $\mathbb{E}[|\hat{F}|] \in [\binom{8}{0} = 1, \binom{8}{1} = 8]$, which gives us a conservative lower bound on the efficiency from a feature-based representation.

C. Initial Strategy Generation

The human training set Ω^E has been generated from a series of similar but randomly-generated environments. Therefore the initial strategy generated from the training set is *independent* from the particular environment that we synthesize a strategy for. For all $(z,a) \in Z \times Act$ we assign the probability of selecting an action $\sigma_z(z,a)$ from its corresponding feature's $f(z,a)$ frequency in the training set compared to the set of all features with observation z:

$$\sigma_z(z,a) = \sum_{(z_j,a_j)\in\hat{F}} \left(\frac{\Omega^E(z_j,a_j)}{\sum_{a_i\in Act} \Omega^E(z_j,a_i)} \right) \ .$$

For the cases where a sequence has no equivalence, we evenly distribute the strategy between the action choices Act (such occasions are rare and our refinement procedure will improve any negative actions after model checking).

$$\sigma_z(z,a) := \frac{1}{|Act|} \quad \text{if} \sum_{a_i\in Act(z)} \Omega^E(z,a_i) = 0 \ .$$

For the strategy σ_z, we perform model checking on the induced MC D^{σ_z} to determine if the specification is satisfied.

D. Refinement Using Counterexamples

When the specification is refuted, we compute a counterexample in form of a *set of critical states* $S' \subseteq S$. Note the probability of satisfying the specification will be comparatively low at these states. The human is then requested to update the strategy for the observations $z = O(s)$ for all $s \in S'$. For an observation z with an action update selection of a_i, the observation-action strategy update parameter $\omega^E(z,a)$ is:

$$\omega^E(z,a) = \begin{cases} \frac{1}{\sum_{s\in S} \Pr(s|z)\Pr_{reach}(s)} & \text{if } a = a_i \ , \\ 1 & \text{otherwise} \ . \end{cases}$$

We perform a Bayesian update with normalization constant c to calculate the stochastic strategy where $c = \sum_{a\in Act} \sigma_z{}'(z,a)$

$$\sigma_z{}'(z,a) = \frac{1}{c}\omega^E(z,a)\sigma_z(z,a) \ .$$

Thereby, at each control loop the probability of the human input a_i in the strategy is increased.

Bounds on Optimality. As discussed in Sect. IV, a randomized memoryless strategy for a POMDP may not induce optimal values in terms of reachability probabilities or expected cost. Moreover, in our setting, there is a limit on the human's capability – for instance if there are too many features to comprehend. An optimal strategy for the *underlying MDP* of a POMDP provides bounds on optimal values for the POMDP. These values are a measure on what is achievable, although admittedly this bound may be very coarse. Such a bounding within a reinforcement learning context is discussed in [5].

VI. Implementation and Experiments

We implemented the motion planning setting as in Fig. 1 inside an interactive MATLAB environment. Grid size, initial state, number of obstacles, and goal location are variables. We use PRISM [25] to perform probabilistic model checking of the induced MC of the POMDP, see Sect. III-D. We use the PRISM POMDP prototype [29] and a point-based value iteration (PBVI) solver [6], [28] for comparison with other

TABLE I: Expected cost improvement – 4×4 gridworld

Iteration	$\Pr(\neg B \cup G)$	Expected Cost ($\mathrm{EC}_{=?}[C]$)
0	0.225	13.57
1	0.503	9.110
2	0.592	7.154
3	0.610	6.055
4	0.636	5.923
Optimal	– n. a. –	5

TABLE II: Expected cost of initial strategy from training sets

Training Grids	$\Pr(\neg B \cup G)$	Expected Cost ($\mathrm{EC}_{=?}[C]$)
Variable	0.425	10.59
Fixed-4	0.503	9.27
Fixed-10	0.311	14.53
Optimal	– n. a. –	3

TABLE III: Comparison to existing POMDP tools

	HiL Synth		PRISM-POMDP		PBVI	
grid	states	time (s)	states	time (s)	states	time (s)
3×3	277	43.74	303	2.20	81	3.86
4×4	990	121.74	987	4.64	256	2431.05
5×5	2459	174.90	2523	213.53	625	– MO –
6×6	5437	313.50	5743	– MO –	1296	– MO –
10×10	44794	1668.30	54783	– MO –	– MO –	– MO –
11×11	– MO –	– MO –	81663	– MO –	– MO –	– MO –

TABLE IV: Strategy refinement – 4×4 gridworld

Iteration	Construction (s)	Model Checking (s)	$\Pr(\neg B \cup G)$
0	2.311	1.533	0.129
1	2.423	1.653	0.521
2	2.346	1.952	0.721
3	2.293	1.727	0.799
4	2.293	1.727	0.799

tools. Note that there exists no available tool to compute optimal randomized memoryless strategies. All experiments were conducted on a 2.5 GHz machine with 4 GB of RAM.

A. Efficient Data Collection

A human user trains an initial "generic" strategy through a simulation of multiple randomly-generated environments, varying in size, number of obstacles and goal location. In order to more accurately reflect the partially observable nature of the problem, the human is only shown a map of the "known" features (landmarks and goal location) in the environment as well as the observation associated with the current state.

The goal is to obtain a strategy from the data that is independent of a change in the environments. We gather inputs according to Sect. V-A and Sect. V-B. For a bound of $\varepsilon = 0.05$ with confidence of $1 - \delta = 0.99$, we require $\left|\Omega^E\right| = 1060$ samples, see Eq. 1. Furthermore, the efficiency factor introduced by the feature equivalence depends on the generated scenarios, i. e., the number of features. For our examples, we conservatively assume an efficiency factor of 4, so we require $\left|\Omega^E\right| = 265$ samples. If the specification is refuted, we compute a critical part $S' \subseteq S$ of the state space S, i. e., a counterexample. By starting the simulation in concrete scenarios at locations induced by S', we "ask" the human for specific inputs that *refine* the strategy at critical parts.

B. Experiments

a) Strategy Refinement: In Table IV we show 5 iterations of counterexample-based *strategy refinement* for a 4×4 gridworld. In each iteration, we measure the time to *construct* the MC and the time to *model check*. These running times are negligible for this small example, important however is the probability for safely reaching a target, namely $\Pr(\neg B \cup G)$. One can see that for the initial, generic strategy this probability is rather low. Having the simulation start in critical parts iteratively improves this probability up to nearly 0.8, at which point we find no measurable improvement. For this example, the upper bound on the maximum probability derived from MDP model checking is 1. Figure 5 shows a *heatmap* of this improving behavior where darker coloring means higher probability for safely reaching the goal.

b) Fixed goal location: When we fix the goal-location parameter to the top-right of the grid, we can examine the strategy refinement's impact on the expected number of steps to the goal (see I). The grid-size space was randomly sampled between $n \in [4, 11]$, we also compare the impact of fixing the grid-size for the training set. There is clearly a benefit to restricting the samples from the training set to samples of similar problem styles. In a 4×4 gridworld, a fixed training set of similar sized environments outperforms the strategies generated by a varying set of environment sizes (see Table II).

c) Comparison to Existing Tools and Solvers: We generated POMDP models for several grid sizes with one landmark and one dynamic obstacle. We list the number of model states and the solution times for our human-in-the-loop synthesis method, PRISM-POMDP and PBVI. From Table III we can see that for the smaller problem sizes, the existing tools perform slightly better than our method. However, as the problem grows larger, both PRISM-POMDP and PBVI run out of memory and are clearly outperformed. The advantage of our memoryless approach is that the strategy itself is independent of the size of the state space and the problem scales with the size of the verification for the induced MC.

VII. Conclusion and Future Work

We introduced a formal approach to utilize humans' inputs for strategy synthesis in a specific POMDP motion planning setting, where strategies provably adhere to specifications. Our experiments showed that with a simple prototype we could raise the state-of-the-art, especially in the combination with formal verification. In the future, we will investigate how to infer decisions based on memory and how to employ human-understandable counterexamples [43].

Acknowledgment

This work has been partly funded by ONR N00014-15-IP-00052, NSF 1550212, and DARPA W911NF-16-1-0001.

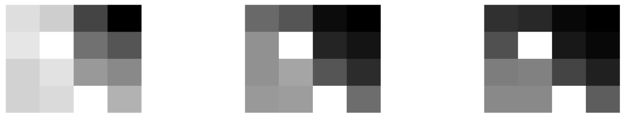
(a) Iteration 0 (b) Iteration 1 (c) Iteration 2

Fig. 5: Heatmap for the quality of agent strategies with dynamic obstacle location $(2,0)$ and static landmark at $(1,2)$.

References

[1] Erika Ábrahám, Bernd Becker, Christian Dehnert, Nils Jansen, Joost-Pieter Katoen, and Ralf Wimmer. Counterexample generation for discrete-time Markov models: An introductory survey. In *SFM*, volume 8483 of *LNCS*, pages 65–121. Springer, 2014.

[2] Christopher Amato, Daniel S. Bernstein, and Shlomo Zilberstein. Optimizing fixed-size stochastic controllers for POMDPs and decentralized POMDPs. *Autonomous Agents and Multi-Agent Systems*, 21(3):293–320, 2010.

[3] Brenna D. Argall, Sonia Chernova, Manuela Veloso, and Brett Browning. A survey of robot learning from demonstration. *Robotics and Autonomous Systems*, 57(5):469–483, 2009.

[4] Nicholas Armstrong-Crews and Manuela Veloso. Oracular partially observable Markov decision processes: A very special case. In *ICRA*, pages 2477–2482. IEEE, 2007.

[5] Anthony R. Cassandra and Leslie Pack Kaelbling. Learning policies for partially observable environments: Scaling up. In *ICML*, page 362. Morgan Kaufmann, 2016.

[6] Anthony R. Cassandra, Leslie Pack Kaelbling, and Michael L. Littman. Acting optimally in partially observable stochastic domains. In *AAAI*, volume 94, pages 1023–1028, 1994.

[7] Krishnendu Chatterjee, Martin Chmelík, Raghav Gupta, and Ayush Kanodia. Qualitative analysis of POMDPs with temporal logic specifications for robotics applications. In *ICRA*, pages 325–330. IEEE, 2015.

[8] Krishnendu Chatterjee, Martin Chmelík, Raghav Gupta, and Ayush Kanodia. Optimal cost almost-sure reachability in POMDPs. *Artificial Intelligence*, 234:26–48, 2016.

[9] Krishnendu Chatterjee, Luca de Alfaro, and Thomas A. Henzinger. Trading memory for randomness. In *QEST*. IEEE, 2004.

[10] Paul Christiano, Jan Leike, Tom B. Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. *CoRR*, abs/1706.03741, 2017.

[11] Martin A. Conway. *Cognitive models of memory*. The MIT Press, 1997.

[12] Christian Dehnert, Sebastian Junges, Joost-Pieter Katoen, and Matthias Volk. A Storm is coming: A modern probabilistic model checker. In *CAV (2)*, volume 10427 of *LNCS*, pages 592–600. Springer, 2017.

[13] Kurt Dresner and Peter Stone. A multiagent approach to autonomous intersection management. *Artificial Intelligence*, 31:591–656, 2008.

[14] Krishnamurthy Dvijotham and Emanuel Todorov. Inverse optimal control with linearly-solvable MDPs. In *ICML*, pages 335–342, 2010.

[15] Ronald A. Howard. *Dynamic Programming and Markov Processes*. The MIT Press, 1960.

[16] Nils Jansen, Murat Cubuktepe, and Ufuk Topcu. Synthesis of shared control protocols with provable safety and performance guarantees. In *ACC*, pages 1866–1873. IEEE, 2017.

[17] Nils Jansen, Ralf Wimmer, Erika Ábrahám, Barna Zajzon, Joost-Pieter Katoen, Bernd Becker, and Johann Schuster. Symbolic counterexample generation for large discrete-time markov chains. *Sci. Comput. Program.*, 91:90–114, 2014.

[18] Edwin T. Jaynes. On the rationale of maximum-entropy methods. *Proceedings of the IEEE*, 70(9):939–952, 1982.

[19] Sebastian Junges, Nils Jansen, Ralf Wimmer, Tim Quatmann, Leonore Winterer, Joost-Pieter Katoen, and Bernd Becker. Permissive finite-state controllers of pomdps using parameter synthesis. *CoRR*, abs/1710.10294, 2017.

[20] Leslie Pack Kaelbling, Michael L. Littman, and Anthony R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101(1):99–134, 1998.

[21] Joost-Pieter Katoen. The probabilistic model checking landscape. In *LICS*, pages 31–45. ACM, 2016.

[22] Piyush Khandelwal et al. BWIBots: A platform for bridging the gap between AI and human–robot interaction research. *Int'l Journal of Robotics Research*, 2017.

[23] Konrad P. Körding and Daniel M. Wolpert. Bayesian decision theory in sensorimotor control. *Trends in Cognitive Sciences*, 10(7):319–326, 2006.

[24] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[25] Marta Kwiatkowska, Gethin Norman, and David Parker. PRISM 4.0: Verification of probabilistic real-time systems. In *CAV*, volume 6806 of *LNCS*, pages 585–591. Springer, 2011.

[26] Michael L. Littman. Memoryless policies: Theoretical limitations and practical results. In *SAB*, pages 238–245. The MIT Press, 1994.

[27] Omid Madani, Steve Hanks, and Anne Condon. On the undecidability of probabilistic planning and infinite-horizon partially observable Markov decision problems. In *AAAI*, pages 541–548. AAAI Press, 1999.

[28] Nicolas Meuleau, Kee-Eung Kim, Leslie Pack Kaelbling, and Anthony R. Cassandra. Solving POMDPs by searching the space of finite policies. In *UAI*, pages 417–426. Morgan Kaufmann, 1999.

[29] Gethin Norman, David Parker, and Xueyi Zou. Verification and control of partially observable probabilistic systems. *Real-Time Systems*, 53(3):354–402, 2017.

[30] Joelle Pineau, Geoff Gordon, and Sebastian Thrun. Point-based value iteration: An anytime algorithm for POMDPs. In *IJCAI*, volume 3, pages 1025–1032, 2003.

[31] David L. Poole and Alan K. Mackworth. *Artificial Intelligence: foundations of computational agents*. CUP, 2010.

[32] Stephanie Rosenthal and Manuela Veloso. Modeling humans as observation providers using POMDPs. In *RO-MAN*, pages 53–58. IEEE, 2011.

[33] Sheldon M. Ross. *Introduction to Stochastic Dynamic Programming*. Academic Press, Inc., 1983.

[34] Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *AISTATS*, pages 627–635, 2011.

[35] Guy Shani, Joelle Pineau, and Robert Kaplow. A survey of point-based POMDP solvers. *Autonomous Agents and Multi-Agent Systems*, 27(1):1–51, 2013.

[36] David R. Shanks, Richard J. Tunney, and John D. McCarthy. A re-examination of probability matching and rational choice. *Journal of Behavioral Decision Making*, 15(3):233–250, 2002.

[37] Robert Sim and Nicholas Roy. Global a-optimal robot exploration in slam. In *ICRA*, pages 661–666. IEEE, 2005.

[38] Trey Smith and Reid Simmons. Heuristic search value iteration for POMDPs. In *UAI*, pages 520–527. AUAI Press, 2004.

[39] Martin A Tanner and Wing Hung Wong. The calculation of posterior distributions by data augmentation. *Journal of the American Statistical Association*, 82(398):528–540, 1987.

[40] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic Robotics*. The MIT Press, 2005.

[41] Nikos Vlassis, Michael L. Littman, and David Barber. On the computational complexity of stochastic controller optimization in POMDPs. *ACM Trans. on Computation Theory*, 4(4):12:1–12:8, 2012.

[42] Michael P. Wellman et al. Designing the market game for a trading agent competition. *IEEE Internet Computing*, 5(2):43–51, 2001.

[43] Ralf Wimmer, Nils Jansen, Andreas Vorpahl, Erika Ábrahám, Joost-Pieter Katoen, and Bernd Becker. High-level counterexamples for probabilistic automata. *Logical Methods in Computer Science*, 11(1), 2015.

[44] Leonore Winterer, Sebastian Junges, Ralf Wimmer, Nils Jansen, Ufuk Topcu, Joost-Pieter Katoen, and Bernd Becker. Motion planning under partial observability using game-based abstraction. In *CDC*. IEEE, 2017.

[45] Tichakorn Wongpiromsarn and Emilio Frazzoli. Control of probabilistic systems under dynamic, partially known environments with temporal logic specifications. In *CDC*, pages 7644–7651. IEEE, 2012.

[46] Brian D. Ziebart, Andrew L. Maas, J. Andrew Bagnell, and Anind K. Dey. Maximum entropy inverse reinforcement learning. In *AAAI*, pages 1433–1438. AAAI Press, 2008.