Plan Explicability and Predictability for Robot Task Planning

Yu Zhang[1], Sarath Sreedharan[1], Anagha Kulkarni[1], Tathagata Chakraborti[1], Hankz Hankui Zhuo[2] and Subbarao Kambhampati[1]

***Abstract*— Intelligent robots and machines are becoming pervasive in human populated environments. A desirable capability of these agents is to respond to goal-oriented commands by autonomously constructing task plans. However, such autonomy can add significant cognitive load and potentially introduce safety risks to humans when agents behave in unexpected ways. Hence, for such agents to be helpful, one important requirement is for them to synthesize plans that can be easily understood by humans. While there exists previous work that studied socially acceptable robots that interact with humans in "natural ways", and work that investigated legible motion planning, there is no general solution for high level task planning. To address this issue, we introduce the notions of plan *explicability* and *predictability*. To compute these measures, first, we postulate that humans understand agent plans by associating abstract tasks with agent actions, which can be considered as a labeling process. We learn the labeling scheme of humans for agent plans from training examples using conditional random fields (CRFs). Then, we use the learned model to label a new plan to compute its explicability and predictability. These measures can be used by agents to proactively choose or directly synthesize plans that are more explicable and predictable to humans. We provide evaluations on a synthetic domain and with a physical robot to demonstrate the effectiveness of our approach.**

I. INTRODUCTION

Intelligent robots and machines are becoming pervasive in human populated environments. Examples include robots for education, entertainment and personal assistance just to name a few. Significant research efforts have been invested to build autonomous agents to make them more helpful. These agents are expected to respond to goal specifications instead of basic motor commands. This requires them to autonomously synthesize task plans and execute those plans to achieve the goals. However, if the behaviors of these agents are incomprehensible, it can increase the cognitive load of humans and potentially introduce safety risks to them.

As a result, one important requirement for such intelligent agents is to ensure that the synthesized plans are comprehensible to humans. This means that instead of considering only the planning model of the agent, plan synthesis should also consider the interpretation of the agent behavior from the human's perspective. This interpretation is related to human's modeling of other agents. More specifically, we tend to have expectations of others' behaviors based on our understanding (modeling) of their capabilities [28] and mental states (belief, desire and intent) [22]. If their behaviors do not match with these expectations, we would often be confused. One of the major reasons of this confusion is due to the fact that our understanding of others' models is often partial and inaccurate. This is also true when humans interact with intelligent agents. For example, to darken a room that is too bright, a robot can either adjust the window blinds, switch off the lights, or stand in front of the window to block the sunlight. While standing in front of the window may well be the least costly plan to the robot in terms of energy consumption (e.g., navigating to and stopping in front of the window requires the least amount of battery power), it is clear that the other two options would better match the human's expectation. One of the challenges here is that the human's understanding of the agent model is inherently hidden. Thus, its interpretation from the human's perspective can be arbitrarily different from the agent's own model. While there exists previous work that studied social robots [12], [13], [23], [19] that interact with humans in "natural ways" and "legible" ways [7], there exists no general solution for high level task planning.

In this paper, we introduce the notions of plan *explicability* and *predictability* which are used by autonomous agents (e.g., robots) to synthesize "explicable" and "predictable" plans, respectively, that can be easily understood and predicted by humans. Our problem settings are as follows: an intelligent agent is given a goal by a human (so that the human knows the goal of the agent) working in the same environment and it needs to synthesize a plan to achieve the goal. As suggested in psychological studies [26], [6], we assume that humans naturally interpret a plan as achieving abstract tasks (or subgoals), which are functional interpretations of agent action sequences in the plan. For example, a robot that executes a sequence of manipulation actions may be interpreted as achieving the task of "$picking$ up cup". Based on this assumption, intuitively, the easier it is for humans to associate tasks with actions in a plan, the more explicable the plan is. Similarly, the easier it is to predict the next task given actions in the previous tasks, the more predictable the plan is. In this regard, explicability is concerned with the association between human-interpreted tasks and agent actions, while predictability is concerned with the connections between these abstract tasks.

Since the association between tasks and agent actions can be considered as a labeling process, we learn the labeling scheme of humans for agent plans from training examples using conditional random fields (CRFs). We then use the learned model to label a new plan to compute its

[1]Yu Zhang, Sarath Sreedharan, Anagha Kulkarni, Tathagata Chakraborti and Subbarao Kambhampati are with the Computer Science and Engineering Department at Arizona State University {yzhan442, ssreedh3, akulka16, tchakra2, rao}@asu.edu

[2]Hankz Hankui Zhuo is with the Computer Science Department at Sun Yat-sen University. zhuohank@mail.sysu.edu.cn

Preprint submitted to 2017 IEEE International Conference on Robotics and Automation. Received September 15, 2016.

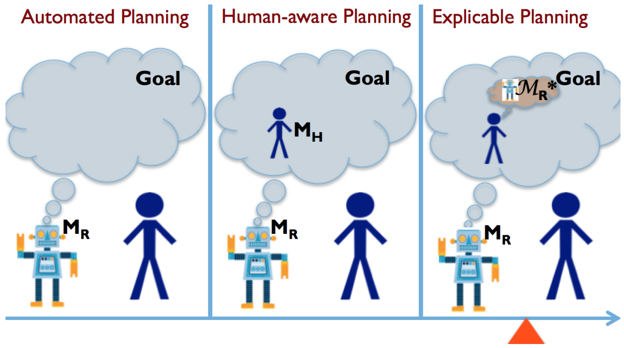

Fig. 1. From left to right, the scenarios illustrate the differences between automated task planning, human-aware planning and explicable planning (this work). In human-aware planning, the robot needs to maintain a model of the human (i.e., M_H) which captures the human's capabilities, intents and etc. In explicable planning, the robot considers the differences between its model from the human's perspective (i.e., $\mathcal{M}_R^*$) and its own model M_R.

explicability and predictability. These measures are used by agents to proactively choose or directly synthesize plans that are more explicable and predictable without affecting the quality much. Our learning approach does not assume any prior knowledge of the human's interpretation of the agent model. We provide evaluation on a synthetic domain in simulation and with human subjects using physical robots to demonstrate the effectiveness of our approach.

II. Related Work

To build autonomous agents (e.g., robots), one desirable capability is for such agents to respond to goal-oriented commands via automated task planning. A planning capability allows agents to autonomously synthesize plans to achieve a goal given the agent model (M_R as shown in the first scenario in Fig. 1) instead of following low level motion commands, thus significantly reducing the human's cognitive load. Furthermore, to work alongside of humans, these agents must be "human-aware" when synthesizing plans. In prior works, this issue is addressed under human-aware planning [24], [5], [3] in which agents take the human's activities and intents into account when constructing their plans. This corresponds to human modeling in human-aware planning as shown in the second scenario in Fig. 1. A prerequisite for human-aware planning is a plan recognition component, which is used to infer the human's goals and plans. This information is then used to avoid interference, and plan for serendipity and teaming with humans. There exists a rich literature on plan recognition [15], [4], [21], [17], and many recent works use these techniques in human-aware planning and human-robot teaming [25], [3], [27]. In contrast, the modeling in this work is one level deeper: it is about the interpretation of the agent model from the human's perspective ($\mathcal{M}_R^*$ in Fig. 1). In other words, R needs to understand the model of *itself* in H's eyes. This information is inherently hidden, difficult to glean, and can be arbitrarily different from R's own model (M_R in Fig. 1).

There exists work on generating legible robot motions [7] which considers a similar issue in motion planning. We are, on the other hand, concerned with task planning. Note that two different task plans may map to exactly the same motions which can be interpreted vastly differently by humans. For example, compare a robot that navigates while holding a cup with the robot executing the same motion trajectory while holding a knife. In such cases, considering only motion becomes insufficient. Nevertheless, there exists similarities between [7] and our work. For example, legibility there is analogous to predictability in ours.

In the human-robot interaction (HRI) community, there exists prior work on how to enable fluent interaction [12], [13], [23], [19] to create more socially acceptable robots [8]. These works, however, apply only to behaviors in specific domains. Compared with model or preference learning via expert inputs, such as learning from demonstration [2], inverse reinforcement learning [1] and tutoring systems [18], which is about learning the "right" model or preferences of the teachers, our work, on the other hand, is concerned with learning model differences. Furthermore, as an extension to our work, when robots cannot find an explicable plan that is also cost efficient, they need to signal to the humans as a "heads-up". In this regard, our work is also related to excuse [10] and explanation generation [11]. Finally, while our learning approach appears to be similar to information extraction [20], we use it to proactively guide planning instead of passively extracting information.

III. Explicability and Predictability

In our setting, an agent R needs to achieve a goal given by a human in the same environment (so that the human knows about the goal of the robot). The agent has a model of itself (referred to as M_R) which is used to autonomously construct plans to achieve the goal. In this paper, we assume that this model is based on PDDL [9], a general planning domain definition language. As we discussed, for an agent to generate explicable and predictable plans, it must not only consider M_R but also $\mathcal{M}_R^*$, which is the interpretation of M_R from the human's perspective.

A. Problem Formulation

Given a domain, the problem is to find a plan for a given initial and goal state that satisfy the following:

$$\underset{\pi_{M_R}}{\operatorname{argmin}}\, cost(\pi_{M_R}) + \alpha \cdot dist(\pi_{M_R}, \pi_{\mathcal{M}_R^*}) \tag{1}$$

where π_{M_R} is a plan that is constructed using M_R (i.e., the agent's plan), $\pi_{\mathcal{M}_R^*}$ is a plan that is constructed using $\mathcal{M}_R^*$ (i.e., the human's anticipation of the agent's plan), $cost$ returns the cost of a plan, $dist$ returns the distance between two plans (capturing their differences), and α is the relative weight. The goal of Eq. (1) is to find a plan that minimizes a weighted sum of the cost of the agent plan and the differences between the two plans. Since the agent model M_R is given, the challenge lies in the second part in Eq. (1).

Note that if we know $\mathcal{M}_R^*$ or it can be learned, the only thing left would be to decide on a proper $dist$ function. However, as discussed previously, $\mathcal{M}_R^*$ is inherently hidden, difficult to glean, and can be arbitrarily different from M_R. Hence, our solution is to use a learning method to directly approximate the returned distance values. We postulate that humans understand agent plans by associating abstract tasks with actions, which can be considered as a labeling process. Based on this, we assume that $dist(\pi_{M_R}, \pi_{\mathcal{M}_R^*})$ can be functionally decomposed as:

$$dist(\pi_{M_R}, \pi_{\mathcal{M}_R^*}) = F \circ \mathcal{L}^*(\pi_{M_R}) \quad (2)$$

where F is a domain independent function that takes plan labels as input, and $\mathcal{L}^*$ is the labeling scheme of the human for agent plans based on $\mathcal{M}_R^*$. As a result, Eq. (1) now becomes:

$$\underset{\pi_{M_R}}{\operatorname{argmin}}\, cost(\pi_{M_R}) + \alpha \cdot F \circ \mathcal{L}(\pi_{M_R} | \{S_i | S_i = \mathcal{L}^*(\pi^i_{M_R})\}) \quad (3)$$

where $\{S_i\}$ is the set of training examples and $\mathcal{L}$ is the learned model of $\mathcal{L}^*$. We can now formally define plan explicability and predictability in our context. Given a plan of agent R as a sequence of actions, we denote it as π_{M_R} and simplified below as π for clarity:

$$\pi = \langle a_0, a_1, a_2, ...a_N \rangle \quad (4)$$

where a_0 is a null action that denotes the start of the plan. Given the domain, we assume that a set of task labels T is provided to label agent actions:

$$T = \{T_1, T_2, ...T_M\} \quad (5)$$

1) Explicability Labeling: Explicability is concerned with the association between abstract tasks and agent actions; each action in a plan is associated with an action label. The set of action labels for explicability is the power set of task labels: $L = 2^T$. When an action label includes multiple task labels, the action is interpreted as contributing to multiple tasks; when an action label is the empty set, the action is interpreted as inexplicable. When a plan is labeled, we can compute its explicability measure based on its action labels in a domain independent way. More specifically, we define:

Definition 1 (Plan Explicability): Given a domain, the explicability θ_π of an agent plan π is computed by a mapping, $F_\theta : \mathbf{L}_\pi \rightarrow [0, 1]$ (with 1 being the most explicable).
$\mathbf{L}_\pi$ above denotes the sequence of action labels for π. An example of F_θ used in our evaluation is given below:

$$F_\theta(\mathbf{L}_\pi) = \frac{\sum_{i \in [1,N]} \mathbf{1}_{L(a_i) \neq \emptyset}}{N} \quad (6)$$

where N is the plan length, $L(a_i)$ returns the action label of a_i, and $\mathbf{1}_{formula}$ is an indicator function that returns 1 when the $formula$ holds or 0 otherwise. Eq. (6) basically computes the ratio between the number of actions with non-empty action labels and the number of all actions.

2) Predictability Labeling: Predictability is concerned with the connections between tasks in a plan. An action label for predictability is composed of two parts: a current label and a next label (i.e., $L \times L$). The current label is also the action label for explicability. The next label (similar to the current label) is used to specify the tasks that are anticipated to be achieved next. A next label with multiple task labels is interpreted as having multiple candidate tasks to achieve next; when empty, it is interpreted as that the next task is unpredictable, or there are no more tasks to be achieved.

Definition 2 (Plan Predictability): Given a domain, the predictability β_π of a plan π is computed by a mapping, $F_\beta : \mathbf{L}^2_\pi \rightarrow [0, 1]$ (with 1 being the most predictable).
$\mathbf{L}^2_\pi$ denotes the sequence of action labels for predictability. An example of F_β is given below which is used in our evaluation when assuming that the current and next labels are associated with at most one task label:

$$F_\beta(\mathbf{L}^2_\pi) = \frac{\sum_{i \in [0,N]} \mathbf{1}_{|L(a_i)|=1} \wedge (\mathbf{1}_{L^2(a_i)=L(a_j)} \vee \mathbf{1}_{L^2(a_{i:N})=\emptyset})}{N+1} \quad (7)$$

where $a_j (j > i)$ is the first action that has a different current label as a_i or the last action in the plan if no such action is unfound, $L^2(a_i)$ returns the next label of a_i and $\mathbf{1}_{L^2(a_{i:N})=\emptyset}$ returns 1 only if the next labels for all actions after a_i (including a_i) are $\emptyset$. Eq. (7) computes the ratio between number of actions that we have correctly predicted the next task and the number of all actions.

Although Eqs. (6) and (7) seem to allow the padding of redundant actions to increase the scores (especially when α in Eq. (3) is set to be large), it is unlikely that such actions can be arbitrarily padded and still labeled as explicable. This is because the task label of an action depends on the context instead the individual action. Note that Eqs. (6) and (7) consider only the case where any action label contains a single task label, which we found to be more convenient for human subjects. Eqs. (6) and (7) can be extended to more complex forms when multiple task labels are used.

B. A Concrete Example

Before discussing how to learn the labeling scheme of the human from training examples, we provide a concrete example to connect the previous concepts and show how training examples can be obtained. In this example, there is a rover in a grid environment working with a human. An illustration of this example is presented in Fig. 2. There are resources to be collected which are represented as boxes. There is one storage area that can store one resource which is represented as an open box. The rover can also make observations. The rover actions include $\{navigate\ l_{from}\ l_{to}\}$, $\{observe\ l\}$ $\{load\ l\}$, and $\{unload\ l\}$, each representing a set of actions since l (i.e., representing a location) can be instantiated to different locations (i.e., $0 - 8$ in Fig. 2). $navigate$ (or nav) can move the rover from a location to one of its adjacent locations; $load$ can be used to pick up a resource when the rover is not already loaded; $unload$ can be used to unload a resource at a storage area if the area is empty; $observe$ (or obs) can be used to make an observation. Once a location is observed, it remains observed. The goal is for the rover to make the storage area non-empty and observe two locations that contain the eye symbol in Fig. 2.

In this domain, we assume that there are three abstract tasks that may be used by the human to interpret the rover's plans: COLLECT (C), STORE (S) and OBSERVE (O). Note

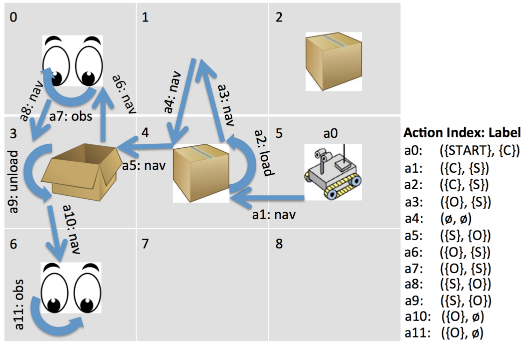

Fig. 2. Example for plan explicability and predictability with action labels (on the right) for a given plan in the rover domain.

that we do not specify any arguments for these tasks (e.g., which resource the rover is collecting) since this information may not be important to the human. This also illustrates that M_R and $\mathcal{M}_R^*$ can be arbitrarily different. In Fig. 2, we present a plan of the rover as connected arrows starting from the its initial location.

Human Interpretation as Training Examples: Let us now discuss how humans may interpret this plan (i.e., associating labels with actions) as the actions are observed *incrementally*: when labeling a_i, we only have access to the plan prefix $\langle a_0, ..., a_i \rangle$. At the beginning for labeling a_0, the observation is that the rover starts at l_5. Given the environment and knowledge of the rover's goal, we may infer that the first task should be COLLECT (the resource from l_4). Hence, we may choose to label a_0 as ({START}, {C}). The first action of the rover (i.e., $nav\ l_5\ l_4$) seems to match with our prediction. Furthermore, given that the storage area is closest to the rover's location after completing COLLECT, the next task is likely to be STORE. Hence, we may label a_1 as ({C}, {S}) as shown in the figure. The second action (i.e., $load\ l_4$) also matches with our expectation. Hence, we label a_2 too as ({C}, {S}). The third action, $nav\ l_4\ l_1$, however, is unexpected since we predicted STORE in the previous steps. Nevertheless, we can still explain it as contributing to OBSERVE (at location l_0). Hence, we may label this navigation action (a_3) as ({O}, {S}). For the fourth action, the rover moves back to l_4, which is inexplicable since the rover's behavior seems to be oscillating for no particular reason. Hence, we may choose to label this action as ($\emptyset$, $\emptyset$). The labeling for the rest of the plan continues in a similar manner. This thought process reflects how training examples can be obtained from human labelers.

IV. Learning Approach

To compute θ_π and β_π from Defs. (1) and (2) for a given plan π, the challenge is to provide a label for each action. This requires us to learn the labeling scheme of humans (i.e., $\mathcal{L}^*$ in Eq. (2)) from training examples and then apply the learned model to π (i.e., $\mathcal{L}$ in Eq. (3)). To formulate a learning method, we consider the sequence of labels as hidden variables. The plan that is executed by the agent (which also captures the state trajectory), as well as any cognitive cues that may be obtained (e.g., from sensing) during the plan execution constitute the observations. The graphical model that we choose for our learning approach is conditional random fields (CRFs) [16] due to their abilities to model sequential data. An alternative would be HMMs; however, CRFs have been shown to relax assumptions about the input and output sequence distributions and hence are more flexible. The distributions that are captured by CRFs have the following form where Z is a normalization factor:

$$p(\mathbf{x}, \mathbf{y}) = \frac{1}{Z} \Pi_A \Phi(\mathbf{x}_A, \mathbf{y}_A) \tag{8}$$

In the equation above, $\mathbf{x}$ represents the sequence of observations, $\mathbf{y}$ represents the sequence of hidden variables, and $\Phi(\mathbf{x}_A, \mathbf{y}_A)$ represents a factor that is related to a subgraph in the CRF model associated with variables $\mathbf{x}_A$ and $\mathbf{y}_A$. In our context, $\mathbf{x}$ are the observations made during the execution of a plan; $\mathbf{y}$ are the action labels. Each factor is associated with a set of features that can be extracted during the plan execution.

A. Features for Learning

Given an agent plan, the set of features that we have access to is the plan and its associated state trajectory:

Plan Features: Given the agent model (specified in PDDL), the set of plan features for a_i includes the action description and the state variables after executing the sequence of actions $\langle a_0, ..., a_i \rangle$ from the initial state. This information can be easily extracted given the model. For example, in our rover example in Fig. 2, this set of features for a_1 includes $navigate$, $at\ rover\ l_4$, $at\ resource0\ l_2$, $at\ resource1\ l_4$, $at\ storage0\ l_3$. Other features (e.g., motion features) can be easily incorporated which we will discuss in future work.

In this work, we use a linear-chain CRF. Although linear-chain CRF is very limited in capturing history information, we show that it is sufficient to distinguish between behavioral patterns in our evaluation domains (including the synthetic domain in Fig. 2) via a combination of the state and action information (i.e., plan features described above). Moreover, our formulation is easily extensible to more general classes of CRFs. Given an agent plan $\pi = \langle a_0, a_1, a_2, ... \rangle$, each action is associated with a set of features. Hence, each training example is of the following form:

$$\langle (F_0, L_0^2), (F_1, L_1^2), (F_2, L_2^2), ... \rangle \tag{9}$$

where L_i^2 is the action label for predictability (and explicability) for a_i. F_i is the set of features for a_i.

B. Using the Learned Model

Given a set of training examples in the form of Eq. (9), we can train the CRF model to learn the labeling scheme in Eq. (3). We discuss two ways to use the learned CRF model.

1) Plan Selection: The most straightforward method is to perform plan selection on a set of candidate plans which can simply be a set of plans that are within a certain cost bound of the optimal plan. Candidate plans can also be generated to be diverse with respect to various plan distances. For each plan, the agent must first extract the features of the actions as we discussed earlier. It then uses the trained model (denoted by $\mathcal{L}_{CRF}$) to produce the labels for the actions in the plan. θ and β can then be computed given the mappings in Defs. (1) and (2). These measures can then be used to choose a plan that is more explicable and predictable.

2) Plan Synthesis: A more efficient way is to incorporate these measures as heuristics into the planning process. Here, we consider the FastForward (FF) planner with enforced hill climbing [14] (see Alg 1). To compute the heuristic value given a planning state, we use the relaxed planning graph to construct the remaining planning steps. However, since relaxed planning does not ensure a valid plan, we can only use action descriptions as plan features for actions that are beyond the current planning state when estimating the θ and β measures. These estimates are then combined with the relaxed planning heuristic (which only considers plan cost) to guide the search.

Algorithm 1 Synthesizing Explicable and Predictable Plans

Input: agent model M_R, trained human labeling scheme $\mathcal{L}_{CRF}$, initial state I and goal state G.
Output: π_{EXP}

Push I into the open set O.
while open set is not empty **do**
  $s =$ GetNext(O).
  if G is reached **then**
    return s.plan (the plan that leads to s from I).
  end if
  Compute all possible next states N from s.
  for $n \in N$ **do**
    Compute the relaxed plan π_{RELAX} for n.
    Concatenate s.plan (with plan features) with π_{RELAX} (with only action descriptions) as $\bar{\pi}$.
    Compute and add other relevant features.
    Compute $\mathbf{L}^2_\pi = \mathcal{L}_{CRF}(\bar{\pi})$.
    Compute θ and β based on $\mathbf{L}^2_\pi$ for $\bar{\pi}$.
    Compute $h = f(\theta, \beta, h_{cost})$ (f is a combination function; h_{cost} is the relaxed planning heuristic).
  end for
  if $h(n^*) < h^*$ ($n^* \in N$ with the minimum h) **then**
    Clear O.
    Push n^* into O; $h^* = h(n^*)$ (h^* is initially MAX).
  else
    Push all $n \in N$ into O.
  end if
end while

The capability to synthesize explicable and predictable plans is useful for autonomous agents. For example, in domains where humans interact closely with robots (e.g., in an assembly warehouse), more preferences should be given to plans that are more explicable and predictable since there would be high risks if the robots act unexpectedly. One note is that the relative weights of explicability and predictability may vary in different domains. For example, in domains where robots do not engage in close interactions with humans, predictability may not matter much.

V. Evaluation

We first evaluate the labeling prediction performance on a synthetic dataset based on the rover domain. The aim is to verify the generalizability of our approach to new scenarios and analyze its sensitivity to model differences (between M_R and $\mathcal{M}^*_R$). Then, we evaluate it with human subjects using physical robots to validate that 1) the predicted labels are indeed consistent with human labeling, and 2) the synthesized plans are more explicable to humans in a blocks world domain. We focused on explicability only for physical robot evaluation due to the resource constraint of human subjects; the two measures are correlated by definition and hence the conclusions are expected to carry over. For generalizability, we introduced new scenarios in the testing phase for both evaluations. In the synthetic domain, we showed that our approach can generalize from scenarios with small model differences to large model differences (with only slight performance degradation). In the physical robot experiment, while the training samples only involved plans to form three block towers, the testing samples included towers of varying heights (3 - 5 blocks). Moreover, our approach does not require a large number of training examples. In the physical robot experiment, we use only 23 unique plans.

A. Systematic Evaluation with a Synthetic Domain

Using a synthetic domain, we evaluate how well the learning approach can capture an arbitrary labeling scheme, as well as the effectiveness of plan selection and synthesis with respect to the θ and β measures.

1) Dataset Synthesis: To simplify the data synthesis process, we make the following assumptions: all rover actions have the same cost; all rover actions are associated with at most one task label (i.e., $L = T \cup \{\emptyset\}$ instead of $L = 2^T$). To construct a domain in which the optimal plan (in terms of cost) may not be the most explicable (in order to differ M_R from $\mathcal{M}^*_R$), we add "oscillations" to the plans of the rover. These oscillations are incorporated by randomly adding locations for the rover to visit as *hidden goals*. For these locations, the rover only needs to visit them. As a result, it may demonstrate "unexpected" behaviors given only the *public goal*, denoted by G, which is known to both the rover and human. We denote the goal that also includes the hidden goals as G'. Given a problem with a public goal G, we implement a labeling scheme as follows to provide the "ground truth" of a rover plan, which is constructed for G'.

Given a plan we label it incrementally by associating each action with a current and next label. These labels are chosen from $\{\{\text{COLLECT}\}, \{\text{STORE}\}, \{\text{OBSERVE}\}, \emptyset\}$. We denote the plan prefix $\langle a_0, ...a_i\rangle$ for a plan π as π_i, the state after applying π_i as s_i from the initial state, and a plan that is constructed from s_i to achieve G (i.e., using s_i as the initial state) as $P(s_i)$. For the current label of a_i:

- If $|P(s_i)| \geq |P(s_{i-1})|$, we label a_i as $\emptyset$ (i.e., inexplicable). This rule means that humans may label an action as inexplicable if it does not contribute to achieving G.
- If $|P(s_i)| < |P(s_{i-1})|$, we label a_i based on the distances from the current rover location to the targets

Preprint submitted to 2017 IEEE International Conference on Robotics and Automation. Received September 15, 2016.

(i.e., storage areas or observation locations), current state of the rover (i.e., loaded or not), and whether a_i moves the rover closer to these targets. For example, if the closest target is a storage area and the rover is loaded, we label a_i as $\{\text{STORE}\}$. When there are ties, we label a_i as $\emptyset$ (i.e., interpreted as inexplicable).

For the next label of a_i:

- This label is determined by the target that is closest to the rover state after the current task is achieved. When there are ties, a_i is labeled as $\emptyset$ (i.e., unclear and hence interpreted as unpredictable). If the current label is $\emptyset$, we also label a_i as $\emptyset$ (i.e., unpredictable).
- If the current task is also the last task, we label a_i as $\emptyset$ since there is no next task.

For evaluation, we define F_θ and F_β as in Eqs. (6) and (7). We randomly generate problems in a 4×4 environment. For each problem, we randomly generate $1-3$ resources as a set RE, $1-3$ storage areas as a set ST, $1-3$ observation locations as a set OB. The public goal G of a problem, first, includes making all storage areas non-empty. To ensure a solution, we force $|RE| = |ST|$ if $|RE| < |ST|$. Furthermore, the rover must make observations at the locations in OB. G' for the rover includes G above, as well as a set of hidden goals. Locations of the rover, RE, ST, OB and hidden goals are randomly generated in the environment and do not overlap in the initial state. Although seemingly simple, the state space of this domain is on the order of 10^{20}.

2) Results: We use only plan features here. First, we evaluate our approach to learning the labeling scheme (i.e., $\mathcal{L}_{CRF}$) as the difference between M_R and $\mathcal{M}_R^*$ gradually increases (i.e., as the number of hidden goals increases). Afterwards, we evaluate the effectiveness of plan selection and synthesis with respect to the θ and β measures. To verify that our approach can generalize to different problem settings, we fix the level of oscillation when generating training samples while allowing it to vary in testing samples.

a) Using CRFs for Plan Explicability and Predictability: In this evaluation, we randomly generate $1-3$ hidden goals to include in G' in 1000 training samples. After the model is learned, we evaluate it on 100 testing samples in which we vary the maximum number of hidden goals from 1 to 6 with step size 1. The result is presented in Fig. 3. We can see that the ratios between θ and β computed based on $\mathcal{L}_{CRF}$ and $\mathcal{L}^*$) is generally between $50\%-150\%$, which reflects the predication performance. We can also see that the oscillation level does not seem to influence the prediction performance much. This shows that our approach is effective whether M_R and $\mathcal{M}_R^*$ are similar or largely different.

b) Selecting Explicable and Predictable Plans: We evaluate plan selection using θ and β measures and compare the selected plans (denoted by EXPD-SELECT) with plans selected by a baseline approach (denoted by RAND-SELECT). Given a set of candidate plans, EXPD-SELECT selects a plan according to the highest predicted explicability or predictability measure while RAND-SELECT randomly selects a plan from the set of candidate plans. To implement

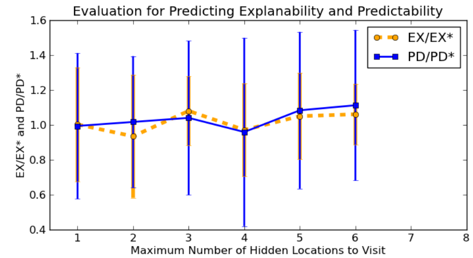

Fig. 3. Evaluation for predicting θ and β as the difference between M_R and $\mathcal{M}_R^*$ increases (as the maximum number of hidden goals increases).

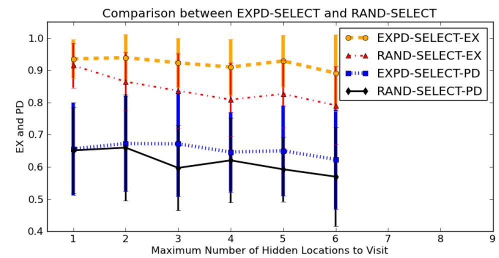

Fig. 4. Comparison of EXPD-SELECT and RAND-SELECT

this, for a given public goal G, we randomly construct 20 problems with a given level of oscillation as determined by the maximum number of hidden goals. Each such problem corresponds to a different G' and a plan is created for it. The set of plans for these 20 problems associated with the same G is the set of candidate plans for G. For each level of oscillation, we randomly generate 50 different Gs and then construct the set of candidate plans for each G. The model here is trained with 1900 samples using the same settings as in a) and we gradually increase the level of oscillation.

We compare the θ and β values computed from the ground truth labeling of the chosen plans. The result is provided in Fig. 4. When the oscillation is small, the performances of both approaches are similar. As the oscillation increases, the performances of the two approaches diverge. This is expected since RAND-SELECT randomly chooses plans and hence its performance should decrease as the oscillation increases. On the other hand, EXPD-SELECT is not influenced as much although its performance also tends to decrease. This is partly due to the fact that the model used in this evaluation is trained with samples having a maximum of 3 hidden goals.

In Fig. 4 for explicability, almost all results are significantly different at 0.001 level (except at 1); for predictability, results are significantly different at 0.01 level at 3, 5 and 6. The trend to diverge is clearly present. Note that linear-chain CRFs is limited in its modeling capability. We anticipate performance improvement with more general CRFs.

c) Synthesizing Explicable and Predictable Plans: We evaluate here plan synthesis using Alg. 1. More specifically, we compare FF planner that considers the predicted θ and β values in its heuristics with a normal FF planner that only considers the action cost. The FF planner with the new heuristic is called FF-EXPD. In this evaluation, we set the

Preprint submitted to 2017 IEEE International Conference on Robotics and Automation. Received September 15, 2016.

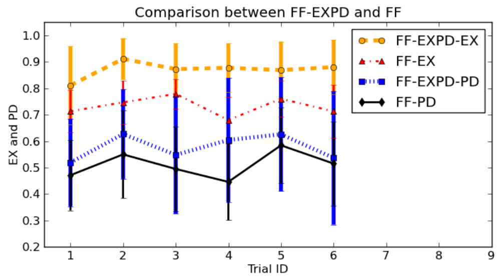

Fig. 5. Comparison of FF and FF-EXPD considering only θ.

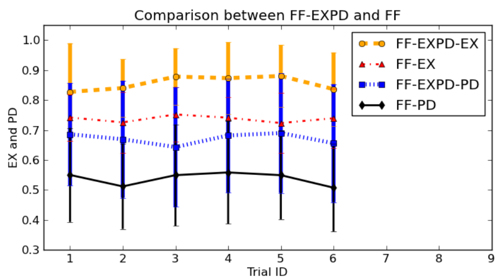

Fig. 6. Comparison of FF and FF-EXPD considering only β.

maximum number of hidden locations to visit to be 6. For each trial, we generate 100 problems and apply both FF and FF-EXPD to solve the problems. Given that we are interested in comparing the cases when explicability is low, we only consider problems when the predicted plan explicability for the plan generated by FF is below 0.85.

First, we consider the incorporation of θ only. The result is presented in Fig. 5. For the explicability measure, we see a significant difference in all trials. Another observation is that the difference in plan predictability is present but not as significant. This evaluation suggests that our heuristic search can produce plans of high explicability.

Next, we consider the incorporation of β only. The result is presented in Fig. 6. Similarly, we see a significant difference in all trials for both θ and β. One observation is that improving on plan predictability also improves plan explicability which is expected given Eqs. (6) and (7)).

Plan Cost: We consider plan cost here for the evaluation in Fig. 6. The result is presented below. We can see that the plan length for FF-EXPD is longer than the plan produced by FF in general. This is expected since FF only considers plan cost. However, in all settings, FF-EXPD penalizes the plan cost slightly (about 10%) to improve the plan explicability and predictability measures.

Trial ID	1	2	3	4	5	6
FF (avg. # steps)	21.9	24.0	24.1	23.9	22.1	22.4
FF-EXPD (avg. # steps)	23.5	26.3	25.2	24.0	23.4	25.0

B. Evaluation with Physical Robots

Here, we evaluate in a blocks world domain with a physical robot. It simulates a smart manufacturing environment where robots are working beside humans. Although the human and robot do not have direct interactions – the robot's goal is independent of the human's, generating explicable plan is still an important issue since it will allow humans focus more on their own tasks by reducing distractions. Here, we evaluate plans generated by the robot using FF-EXPD and a cost-optimal planner (OPT). We analyze the consistency between human and system labeling, and compare the plans with human subjects in terms of their explicability.

1) Domain Description: In this domain, the robot's goal (which is known to the human) is to build a tower of a certain height using blocks on the table. The towers to be built have different heights in different problems. There are two types of blocks, light ones and heavy ones, which are indistinguishable externally but the robot can identify them based on the markers. Picking up the heavy blocks are more costly than the light blocks for the robot. Hence, the robot may sometimes choose seemingly more costly (i.e., longer) plans to build a tower from the human's perspective.

2) Experimental Setup: We generated a set of 23 problems in this domain in which towers of height 3 are to be built. The plans for these problems were manually generated and labeled as the training set. For 4 out of these 23 problems, the optimal plan is not the most explicable plan. To remove the influence of grounding, we also generated permutations of each plan using different object names for these 23 problems, which resulted in a total of about 15000 training samples. We then generated a set of 8 testing problems for building towers of various heights (from $3-5$) to verify that our approach can generalize. Testing problems were generated only for cases where plans are more likely to be inexplicable. For each problem, we generated two plans, one using OPT and the other using FF-EXPD, and recorded the execution of these plans on the robot. We recruited 13 subjects on campus and each human subject was tasked with labeling two plans (generated by OPT and FF-EXPD respectively) for each of the 8 testing problems, using the recorded videos and following a process similar to that in training. After labeling each plan, we also asked the subject to provide a score ($1-10$ with 10 being the most explicable) to describe how comprehensible the plan was overall.

3) Results: In this evaluation, we only use one task label "*building tower*". For all testing problems, the labeling process results in 77.8% explicable actions (i.e., actions with a task label) for OPT and 97.3% explicable actions for FF-EXPD. The average explicability measures for FF-EXPD and OPT are 0.98 and 0.78, and the average scores are 9.65 and 6.92, respectively. We analyze the results using a paired T-test which shows a significant difference between FF-EXPD and OPT in terms of the explicability measures (using Eq. (6)) computed from the human labels and the overall scores ($p < 0.001$ for both). Furthermore, after normalizing the scores from the human subjects, the Cronbach's α value shows that the explicability measures and the scores are consistent for both FF-EXPD and OPT ($\alpha = 0.78, 0.67$, respectively). These results verify that: 1) our explicability measure does capture the human's interpretation of the robot plans and 2) our approach can generate plans that are more explicable to humans. In Fig. 7, we present the plans for a testing scenario. The left part of the figure shows the plan

Preprint submitted to 2017 IEEE International Conference on Robotics and Automation. Received September 15, 2016.

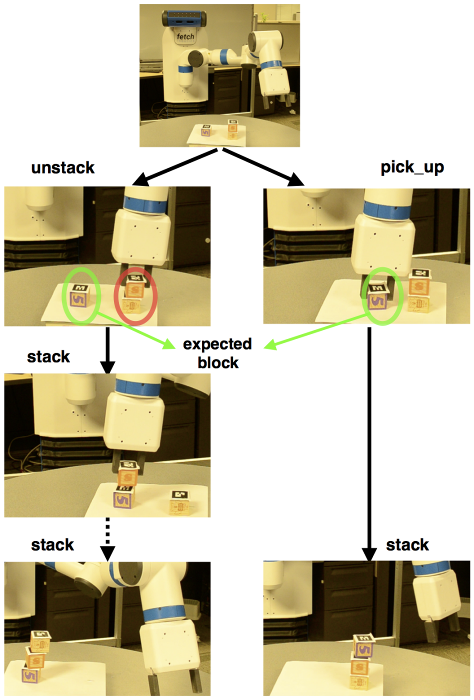

Fig. 7. Execution of two plans generated by OPT (left) and FF-EXPD (right) for one out of the 8 testing scenarios. The top figure shows the setup where the goal is to build a tower of height 3. The block initially on the left side of the table is a heavy block. The optimal plan involves manipulating the light blocks (i.e., putting the two light blocks on top of the heavy one); the explicable plan is more costly since it requires moving the heavy one.

generated by OPT and the right part shows the plan generated by FF-EXPD. A video is also attached showing the different behaviors with the two planners in this scenario.

VI. Conclusion

In this paper, we introduced plan explicability and predictability for intelligent robots so that they can synthesize plans that are more comprehensible to humans. To achieve this, they must consider not only their own models but also the human's interpretation of their models. To the best of our knowledge, this is the first attempt to model plan explicability and predictability for task planning which differs from previous work on human-aware planning. To compute these measures, we learn the labeling scheme of humans for agent plans from training examples based on CRFs. We then use this learned model to label a new plan to compute its explicability and predictability.

In future work, we will investigate additional features. For example, motion features can capture the smoothness of the execution; features can also be extracted from sensors such as video cameras. Moreover, we plan to evaluate with more general CRFs and deep learning approaches (e.g., LSTM). Finally, while we focus on generating more explicable and predictable robot behavior, our approach also has other interesting applications. For example, many defense applications use planning to create unpredictable and inexplicable plans, which can help deter or confuse enemies and are useful for testing defenses against novel or unexpected attacks.

References

[1] P. Abbeel and A.Y. Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the Twenty-first International Conference on Machine Learning*, ICML, 2004.

[2] B.D. Argall, S. Chernova, M. Veloso, and B. Browning. A survey of robot learning from demonstration. *Robot. Auton. Syst.*, 57(5):469–483, May 2009.

[3] T. Chakraborti, G. Briggs, K. Talamadupula, Y. Zhang, M. Scheutz, D. Smith, and S. Kambhampati. Planning for serendipity. In *IROS*, 2015.

[4] E. Charniak and R.P. Goldman. A bayesian model of plan recognition. *Artificial Intelligence*, 64(1):53 – 79, 1993.

[5] M. Cirillo, L. Karlsson, and A. Saffiotti. Human-aware task planning for mobile robots. In *ICAR*, pages 1–7, June 2009.

[6] G. Csibra and G. Gergely. Obsessed with goals: Functions and mechanisms of teleological interpretation of actions in humans. *Acta psychologica*, 124(1):60–78, 2007.

[7] A. Dragan and S. Srinivasa. Generating legible motion. In *RSS*, Berlin, Germany, June 2013.

[8] T.W. Fong, I. Nourbakhsh, and K. Dautenhahn. A survey of socially interactive robots. *Robotics and Autonomous Systems*, 2003.

[9] M. Fox and D. Long. PDDL2.1: An extension to pddl for expressing temporal planning domains. *J. Artif. Int. Res.*, 20(1), 2003.

[10] M. Gbelbecker, T. Keller, P. Eyerich, M. Brenner, and B. Nebel. Coming up with good excuses: What to do when no plan can be found. In *ICAPS*, 2010.

[11] M. Hanheide, M. Gbelbecker, G. S. Horn, A. Pronobis, K. Sj, A. Aydemir, P. Jensfelt, C. Gretton, R. Dearden, M. Janicek, H. Zender, G. Kruijff, N. Hawes, and J.L. Wyatt. Robot task planning and explanation in open and uncertain worlds. *Artificial Intelligence*, 2015.

[12] G. Hoffman and C. Breazeal. Cost-based anticipatory action selection for human–robot fluency. *IEEE Transactions on Robotics*, 23(5):952–961, 2007.

[13] G. Hoffman and C. Breazeal. Effects of anticipatory action on human-robot teamwork efficiency, fluency, and perception of team. In *HRI*, pages 1–8, 2007.

[14] J. Hoffmann and B. Nebel. The FF planning system: Fast plan generation through heuristic search. *J. Artif. Int. Res.*, 14(1):253–302, May 2001.

[15] H.A. Kautz and J.F. Allen. Generalized Plan Recognition. In *AAAI*, pages 32–37, 1986.

[16] J.D. Lafferty, A. McCallum, and F.C.N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*, pages 282–289, 2001.

[17] S.J. Levine and B.C. Williams. Concurrent plan recognition and execution for human-robot teams. In *ICAPS*, 2014.

[18] T. Murray. Authoring Intelligent Tutoring Systems: An analysis of the state of the art. *Intl. J. of Artif.. Int. in Education*, 10:98–129, 1999.

[19] V. Narayanan, Y. Zhang, N. Mendoza, and S. Kambhampati. Automated planning for peer-to-peer teaming and its evaluation in remote human-robot interaction. In *HRI (Extended abstract)*, 2015.

[20] F. Peng and A. McCallum. Information extraction from research papers using conditional random fields. *Information Processing & Management*, 42(4):963 – 979, 2006.

[21] M. Ramírez and H. Geffner. Probabilistic plan recognition using off-the-shelf classical planners. In *AAAI*, 2010.

[22] A.S. Rao and M.P. Georgeff. Bdi agents: From theory to practice. In *ICMAS*, pages 312–319, 1995.

[23] J. Shah, J. Wiken, B. Williams, and C. Breazeal. Improved human-robot team performance using chaski, a human-inspired plan execution system. In *HRI*, pages 29–36, 2011.

[24] E.A Sisbot, L.F. Marin-Urias, R. Alami, and T. Simeon. A human aware mobile robot motion planner. *IEEE Transactions on Robotics*, 23(5):874–883, Oct 2007.

[25] K. Talamadupula, G. Briggs, T. Chakraborti, M. Scheutz, and S. Kambhampati. Coordination in human-robot teams using mental modeling and plan recognition. In *IROS*, pages 2957–2962, 2014.

[26] R.R. Vallacher and D.M. Wegner. What do people think they're doing? action identification and human behavior. *Psychological Review*, 94(1):3–15, 1987.

[27] Y. Zhang, V. Narayanan, T. Chakraborty, and S. Kambhampati. A human factors analysis of proactive assistance in human-robot teaming. In *IROS*, 2015.

[28] Y. Zhang, S. Sreedharan, and S. Kambhampati. Capability models and their applications in planning. In *AAMAS*, 2015.