2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
Daejeon Convention Center
October 9-14, 2016, Daejeon, Korea

# Inferring Human Intent from Video by Sampling Hierarchical Plans

Steven Holtzen*[1], Yibiao Zhao*[2], Tao Gao[2], Joshua B. Tenenbaum[2], Song-Chun Zhu[1]

*Abstract*— **This paper presents a method which allows robots to infer a human's hierarchical intent from partially observed RGBD videos by imagining how the human will behave in the future. This capability is critical for creating robots which can interact socially or collaboratively with humans. We represent intent as a novel hierarchical, compositional, and probabilistic And-Or graph structure which describes a relationship between actions and plans. We infer human intent by reverse-engineering a human's decision-making and action planning processes under a Bayesian probabilistic programming framework. We present experiments from a 3D environment which demonstrate that the inferred human intent (1) matches well with human judgment, and (2) provides useful contextual cues for object tracking and action recognition.**

## I. Introduction

Reasoning about humans is central to a number of applications in robotics, from human-robot interaction to self-driving vehicles. Humans utilize many contextual cues gained from observing others, such as object affordances and detections. These cues form a foundation of human perception. Inferring the intent of a human will be essential for integrating robots into our everyday lives. A robot that interacts with humans must be able to infer a human's goals based on passive observation.

We consider a scene with two agents: an actor, who wishes to accomplish some goal, and a passive observer (e.g. robot) who is trying to infer the goal of the actor for a collaborative task. In our setting, the actor has a well-defined hierarchical intention, such as making coffee: the goal of the observer is to infer the actor's hidden plan. The observer has the visual perception capabilities of a modern robot, namely object and skeleton tracking. We assume that the actor is capable of performing their chosen plan optimally (called the rationality assumption) [11] and that the actor has perfect knowledge of the state of the scene. The key issues are (1) how to infer an actor's intent, and (2) how to model the state of the scene. The robot observer does not have perfect scene knowledge due to the inherent ambiguity in modern computer vision object and skeleton tracking algorithms, as well as occlusions. In this scenario, the observer must effortlessly integrate partial information as it is revealed over time into a cohesive representation of not just the scene, but also the human's mind.

* These authors contributed equally to this work.
[1] Steven Holtzen and Song-Chun Zhu are with the Center for Vision, Cognition, Learning and Autonomy at the University of California, Los Angeles. E-mail: sholtzen@cs.ucla.edu, sczhu@stat.ucla.edu
[2] Yibiao Zhao, Tao Gao, and Joshua Tenenbaum are with the Computational Cognitive Science group at the Department of Brain and Cognitive Sciences at the Massachusetts Institute of Technology. Tao Gao is also affiliated with G.E. Global Research. E-mail: ybz@mit.edu, taogao@mit.edu, jbt@mit.edu.

To address the above challenges, in this paper we focus on one important aspect of modeling a human mind: inferring intent from limited knowledge and partial observations. We can use the inferred plan of the actor to refine the scene understanding of the observer (i.e. by observing the actor making coffee, the machine can infer the presence of an occluded cup). We treat the observed human's intent as a deterministic but unobservable program; we seek to uncover the inputs to this latent process.

We consider the joint distribution over all possible plans, actions, and world states parameterized by a And-Or graph (AoG), a stochastic context-sensitive grammar. This representation is used because it is (1) hierarchical, making it capable of representing realistic human plans; (2) compositional, allowing for a compact representation of the many possible configurations of plans and actions; (3) probabilistic, which naturally encodes the uncertainty resulting from partial or noisy observations. To perform plan inference on this structure, we use an algorithm similar to particle filtering (PF). Like PF, we track the most likely explanation over time as a number of parse graphs drawn from an And-Or graph. Unlike in typical PF, which tracks a low-dimensional distribution, parse graphs are encoded as a complex And-Or graph structure over time.

### A. Overview of our method

The overall goal of our method is: given an observed 3D skeleton trajectory in a continuous 2D plane, construct an And-Or graph that explains the observed sequence of actions. An example of such a system is visualized in Fig. 1. This synthesizing process can be thought of as inverting the hierarchical planning process in a human's mind, similar to what is suggested in [2]. We assume that humans plan optimally to the best of their current knowledge, a concept known as the *principle of rationality* [2]. As demonstrated through various psychological studies [11], humans routinely apply the rationaly assumption to one-another when making goal inferences. Using this assumption, we generate optimal trajectories using a low-level continuous motion planner. With each subsequent observation, we tune the production probabilities of the And-Or graph to better predict future motions and actions.

The remainder of this paper will be organized as follows. First, we will review the relevant literature and highlight our contributions. Then we will formalize the above intuition about plan inference: (1) define the posterior distribution over plans; (2) introduce how to compute probabilities over the And-Or graph and specific parse graphs; (3) simulate trajectories for a given parse graph; (4) compare simulated

978-1-5090-3762-9/16/$31.00 ©2016 IEEE

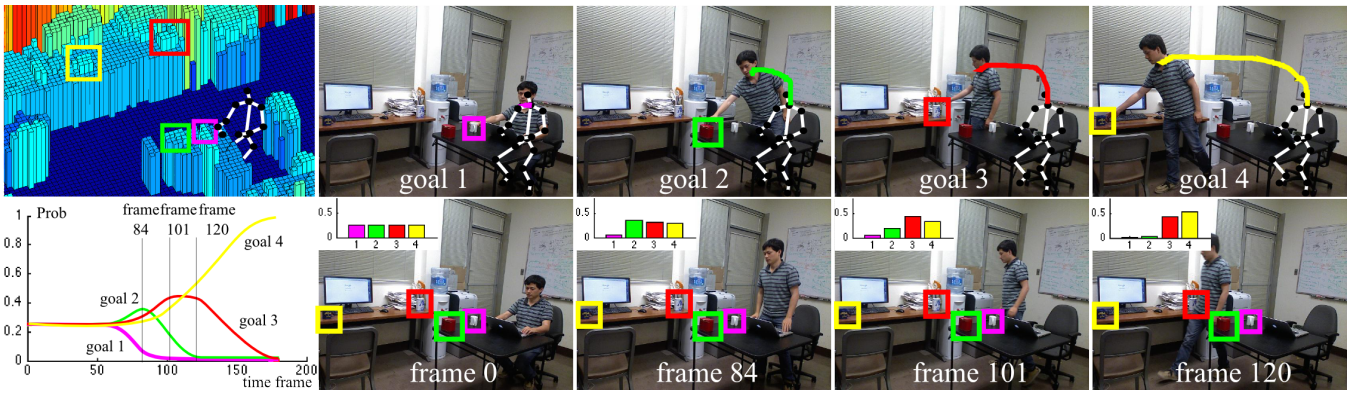


Fig. 1. The plan inference task, seen from the perspective of a observing robot. The top left panel shows 4 different goals (target objects) in a 3D scene. The bottom left panel shows one outcome of the proposed method: the marginal probability of each terminal action over time. We note that terminal actions are marginal probabilities over the probability density described by the hierarchical And-Or graph. The remaining four images on the first row show four rational hierarchical plans for different goals: Goal 1 is within reach, which does not require standing up; Goal 2 requires standing up and reaching out; Goal 3 and Goal 4: require standing up, moving, and reaching for different objects. The second row shows a progression of time corresponding to the bottom left panel. The action sequence and its corresponding probability distributions for each of these four goals are visualized in the bar plots in the upper left of each frame. We can see that when the agent stands up, the posterior probability of the first goal drops suddenly.

and observed trajectories; (5) update the distribution of plans to maximize the probability of generating trajectories that match the observations. In the experiments section, we will demonstrate (1) our method performs well on the task of inferring a plan compared against three baselines; (2) our method improves object tracking and detection results in challenging occluded settings.

### B. Related work

**Psychological motivation.** Our work is motivated by a series of cognitive science studies by Baker et al. [4], [2], [3] about the *Bayesian Theory of Mind* (ToM), which suggests an intentional agent's behavior is based on the *principle of rationality*: the expectation that agents will behave rationally to efficiently achieve their goals given their beliefs about the world, as demonstrated in [11]. This assumption is similar to those made in other plan recognition works like [12], [10]. To simplify our task, we also assume that humans are displaying their intent explicitly. This assumption is a simplification of a traditional cognitive science research task, which typically focuses on difficult cases where the intention is hidden. However, even with this assumption, plan inference remains an interesting and challenging problem for A.I. research as an initial step toward solving the more general plan inference problem.

**Inverse planning.** Plan inference has previously been posed as the problem of inferring a most-likely sequence of transitions in a dynamic Bayesian network (DBN) for trajectory disambiguation as in [8], [2], [6], [5], [21], [35], [40], [26], [23], [17]. The proposed method is distinct in its ability to infer hierarchical goals in the form of context-sensitive grammars, giving it considerably more flexibility than a DBN model. One major limitation of plan inference based on DBN methods is its inability to cope with long-term dependencies or plans with an arbitrarily deep hierarchy. The Markov property implicit in these solutions makes tracking long-term dependencies cumbersome. Some extensions of Bayesian networks to handle long-term dependencies such as [5] have been proposed; however, these methods are still limited to expressing context-free plans with limited depths, unlike the proposed method which has equivalent expressibility to a context-sensitive grammar.

Plan inference has classically been explored as a problem of lifted inference in first-order logic [19]. This framework has recently been applied to the problem of inferring a human's intention in a robot-human teaming scenario in [32]. We seek to extend this plan representation to a context-sensitive stochastic grammar, which affords probabilistic inference strategies such as particle filtering. This extension allows for a principled conditioning process based on Bayes rule for incorporating observations.

Plan inference with context-sensitive grammars has been explored in [12], [28], [27]; however, they only consider the problem of inferring intent in a discrete planning domain. This simplifies the search process by disregarding information from partially observed trajectories; integrating this partial information is one of our contributions.

One-step goal inference has been posed as a problem of inverse optimal control or inverse reinforcement learning as in [40], [24], [1], [7]. We are distinct in our use of a particle sampler which simulates a hierarchical multi-step trajectory, rather than computing a most likely Markov decision process with a single goal.

**Forward planning.** We choose to use an And-Or graph representation over the many alternative forward planning algorithms such as hierarchical task networks [9], [25] due to its ability to represent and model uncertainty [39]. The inverse planning problem is inherently probabilistic and requires representing distributions over all possible plans. The And-Or graph has been considered for forward planning in [22] and inverse planning in [27]. The proposed method extends these result to allow for the inference of complete plans from partial trajectories, as well as introducing a new

sampling and conditioning process.

**Probabilistic programming.** Probabilistic programming is a paradigm in which probability distributions are defined by deterministic programs parameterized by random variables [13], [14]. We view plan inference in a similar way. We use a deterministic program (a planner which provides a valid sequence of sub-goals and continuous trajectories) parameterized by random variables (the presence of terminal nodes and production probabilities). To perform inference, we sample from a deterministic planner according to the distributions defined by the And-Or graph structure.

### C. Contributions

In comparison with the above literature, we make the following contributions:

- We model human intention using a generative hierarchical, compositional, and probabilistic And-Or graph.
- We propose a particle-filtering-based process for performing hierarchical plan inference on an And-Or graph. This process is capable of inferring *long-term planning dependencies* and *context-sensitive policies*.
- We benchmark the efficacy of our approach through experiments which jointly infer object recognition, action detection, and intent. The additional context gained by plan recognition improves object and action detection rates in previously challenging occluded settings.

## II. Modeling Hierarchical Intent

In this section, we will introduce our algorithm to infer the hidden causes behind observed human movement and object displacements with visual uncertainties. This mirrors a typical environment in which a robot is observing a human interacting with a 3D scene. An overview of the method is presented in Algorithm 1.

We use a graph structure called a *temporal And-Or Graph* (T-AoG), dubbed "temporal" due to the fact that AND nodes constrain their children to be executed in sequence. This graph is a representation of a stochastic context-sensitive grammar $S = \langle S, V_n, T, R, P \rangle$, where $S$ is the root node, $V_n$ is the set of non-terminal nodes, $T$ is the set of terminal nodes, $R$ is the set of production rules, $P$ is the set of probabilities on production rules, similar to the formulations in [39], [27]. We consider a *parse graph pg* to be a valid sentence from this grammar, i.e. a plan. From here on we use the term *pg* and plan interchangeably. We further decompose the non-terminal nodes into two sets: the AND-nodes $V_{\text{and}}$ and the OR-nodes $V_{\text{or}}$. The $V_{\text{and}}$ nodes encode a temporal relationship between its children (i.e. in Fig. 2, action $B$ must occur before action $C$ in $pg_1$). The $V_{\text{and}}$ nodes have a production probability of 1 and thus and must always occur for a valid plan. The $V_{\text{or}}$ nodes form a production rule with an associated probability $\omega_i$, i.e. you may choose one of its children each weighted with a certain probability. The terminal nodes $T$ are non-decomposable observations (i.e. walk, stand, sit, etc.). Terminal nodes also have an associated production probability which corresponds to the likelihood that a given action or object state is observed.

Thus the inverse planning problem can be formulated in a Bayesian framework as computing the T-AoG that maximizes the likelihood of generating the observed human actions $X_{\text{obs}}$. We learn the parameters $\omega$ by drawing sample *pg*s from the T-AoG and comparing them with $X_{\text{obs}}$. The maximum a-posteriori (MAP) *pg*, denoted $pg^\star$, is therefore the most likely plan. In addition to producing $pg^\star$, we also produce a probability distribution over all possible other plans in the form of a learned T-AoG, giving it distinctly more expressive power than traditional plan recognition strategies based on parsing a stochastic context-sensitive grammar, e.g. those in [12] which are limited to producing MAP estimates.

We formulate the plan recognition task in terms of approximate sampling and prediction from the posterior distribution:

$$\begin{aligned} P(pg \mid X_{\text{obs}}) &\propto P(pg)P(X_{\text{obs}} \mid pg) \\ &\propto \sum_{X_{\text{pred}}} P(pg)P(X_{\text{pred}} \mid pg)P(X_{\text{obs}} \mid X_{\text{pred}}) \\ &\propto \sum_{X_{\text{pred}}} P(pg)\delta_{f(pg,X)}(X_{\text{pred}})P(X_{\text{obs}} \mid X_{\text{pred}}), \end{aligned} \tag{1}$$

where *pg* is sampled from the T-AoG, $X_{\text{obs}}$ is the observed trajectory (i.e. it includes the trajectory from the initial time to the current time), and $\delta_{f(pg,X)}(X_{\text{pred}})$ is a delta function representing whether current $X_{\text{pred}}$ can be generated from the *pg*, $f(pg,X)$ is a deterministic program generating trajectories, i.e. the hierarchical planner including a high-level planner HTN or T-AoG, and low-level planner $RRT*$. We can interpret $pg^\star$ as the MAP distribution of the plans in the T-AoG:

$$pg^\star = \arg\max_{pg} P(pg \mid X_{\text{obs}}). \tag{2}$$

Then $pg^\star$ is the most likely plan an agent is currently executing. A key observation of this formulation is the integration of the deterministic function $\delta$; this determinism inside a stochastic process is analogous to a probabilistic programming approach taken in [31], which treats cognitive processes as latent deterministic sub-programs with probabilistic inputs (see Sec. I-B). This gives us generality beyond a typical DBN or grammar-based approach by allowing us to formulate arbitrary programs as latent causes for agent's actions, over which we can do Bayesian inference. The following subsections show how we (1) construct this deterministic planning procedure and (2) perform inference to compute $P(pg \mid X_{\text{obs}})$.

### A. Modeling Intent

Here we formulate $P(pg)$. An actor's intent is modeled by a temporal *pg*, as seen in Fig. 2, which describes a hierarchical decomposition of events and actions that an actor performs in order to accomplish a goal. The derivation of the tree is specified as a *probabilistic grammar* similar to [12] or the PHATT system described in [10]; however, we do inverse planning in a continuous space, which requires integrating partial information about trajectories. The terminal nodes

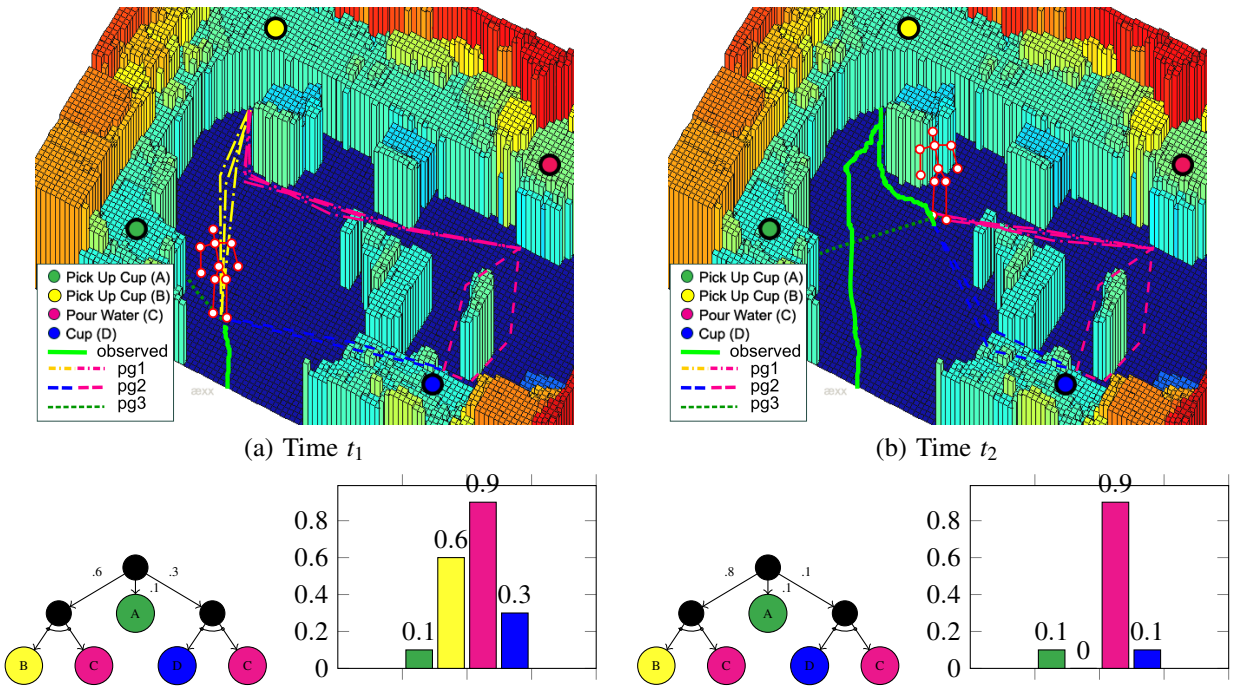


Fig. 2. The particle filtering process. We see an agent in a 3D scene executing a small hierarchical plan. The top row shows the agent situated in a 3D scene with four terminal objectives. The bottom row shows both the hierarchical And-Or graph representation of the agent's plan and the marginal probability for a given terminal action to be executed in the future. Each non-terminal node is a black filled circle. Edges with arcs are AND nodes, edges with annotated production probabilities are OR nodes. We see that over time the AoG production weights are updated in order to increase the likelihood of the observed trajectory. The updated AoG at time $t_2$ imagines trajectories according to the posterior probability density, incorporating the information about the observed trajectory.

in our grammar represent atomic actions that can not be decomposed further. The non-terminal nodes represent either AND structures, whose children must be executed in order, or OR structures, of which only one child must be executed.

We represent intent as a temporal And-Or graph (T-AoG), as seen in Fig. 2, similar to those used in [39], [27]. The T-AoG encodes a probability distribution over all plans, $P(pg)$, as shown in Fig. 2.

We are interested in computing the probability of a $pg$; this would correspond to the likelihood of a given plan (See Fig. 2). The probability of a given terminal node $T_i$ is computed in Sec. II-C. The probability of an OR-node $O$ to take branch $i$ is $\omega_i$, again shown in Fig. 2. These $\omega_i$ along with $T_i$ parameterize the entire distribution of plan proposals; the goal of inference is to learn these $\omega$ to maximize the posterior distribution given in Eq. 1.

We compute the probability of a given $pg$ from the prior distribution over $pg$s recursively, as given in Fig. 2, according to:

$$q(n) = \begin{cases} \omega_i \times q(\text{child}(n)) & \text{If } n \text{ is an OR-node} \\ \prod_i q(\text{children}(n)) & \text{If } n \text{ is an AND-node} \\ T_i & \text{If } n \text{ is a terminal node.} \end{cases} \tag{3}$$

Therefore by taking $q(pg)$, i.e. computing the recursive probability of the root of a given $pg$, we compute the probability of a given $pg$. Note that OR-nodes in a $pg$ always have a single child with one production probability $\omega_i$ while AND-nodes have many.

Thus the proposal distribution for inferring intent involves drawing $pg$s from the distribution defined by the T-AoG:

$$P\left(\delta_{f(pg,X)}(X_{\text{pred}}) = pg\right) \sim q(pg). \tag{4}$$

This forms the foundation for the particle filtering inference algorithm outlined in Sec. II-D and is motivated from a probabilistic programming perspective (See Sec.I-B). We do not consider here the problem of initializing the weights $\omega_i$ or the grammar encoded by this T-AoG; this could be learned through observations of human interactions using the method shown in [29].

### B. Rapidly-Exploring Random Tree* (RRT*)

We show how to generate the terminal nodes $T$ from the T-AoG, e.g. lets us generate $X_{\text{make coffee}}$. RRT* is a low-level subroutine invoked by the T-AoG whose purpose is to find the minimum-cost path from one location in space to another. RRT* executes terminal sub-goals to simulate rational behaviors $f(pg, B) \rightarrow X_{\text{pred}}$. The variable $B$ represents a background collision map, which is encoded in the actor's beliefs of the scene. See Fig. 3(a) for a visualization of this process in two dimensions.

An RRT* planner (based on the algorithm in [18]) is used for generating approximate rational plans in a complex environment, which allows the planner to generate `walkto` and `reach`, which are not discrete events but rather sequences of observable motions in a continuous space. See

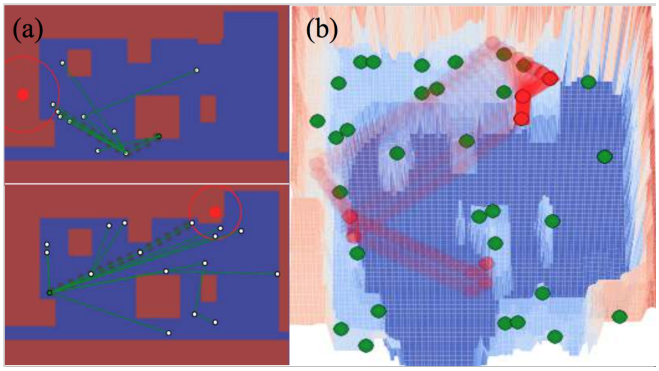


Fig. 3. Visualization of the RRT* algorithm sampling trajectories in a (a) 2D occluded plane corresponding to an occlusion map computed from the office scene in Fig. 2 and (b) 3D voxel space. The goals are shown with green dots; this setting is office scene from Sec. III. In (b) we see how our modified DTW algorithm integrates the motion of humans (shown with red dots) to accomplish a 2-step hierarchical goal. Each trajectory shown above is a particle.

the synthesized trajectories in Fig. 2 for an example of how RRT* can be used to sample from the distribution of plans defined by a T-AoG.

A predicted mental status $pg$ sampled from the AoG according to $P(pg)$ could be translated into a sequence of actions $X_{\text{pred}}$ by RRT*. It is very unlikely that a simulated trajectory will exactly match the observed behavior $X_{\text{obs}}$. Instead of requiring exact matches, our formulation relaxes the matching problem by a stochastic likelihood $P(X_{\text{obs}}|X_{\text{pred}})$, extending its generality beyond a simple most-likely parse such as those in [12].

### C. Dynamic Time Warping (DTW)

Here we model $P(X_{\text{obs}} \mid X_{\text{pred}})$, which is a measure of how well the observed trajectory matches the synthesized trajectories. A Dynamic Time Warping (DTW) algorithm [33] measures the similarity between two temporal sequences which may vary in time or speed. The DTW algorithm outputs the shortest mean distance as well as matching correspondences between two matched sequences. The shortest mean distance is fed to the stochastic likelihood function in the form of a simple Gaussian function. The Gaussian variance controls the tolerance of the model. The output matching correspondences provide the detailed parsing for each frame of the observed sequence.

We modified the original DTW algorithm in order to make it more suitable for matching multi-dimensional sequences and sub-sequences. By using sub-sequence matching, we allow a complete predicted sequence match to be compared to an incomplete observed sequence.

DTW provides a distance between two trajectories, which is used as an energy in a Boltzmann distribution assessing how likely a given sequence is to be the actor's course of action. For example, if the actor is currently walking away from the coffee pot and the next action expected is `make_coffee`, then the distance between the observed and predicted trajectories will be large, making $P(X_{\text{obs}}|X_{\text{make coffee}})$ correspondingly smaller.

### D. Stochastic Inference

Here we model $P(pg \mid X_{\text{obs}})$. The online parsing algorithm has two major steps: stochastic filtering and stochastic memoization. The stochastic filtering algorithm works like a particle filtering algorithm [16] in spirit, which uses importance sampling to filter out the proposal samples. The stochastic memoization technique is used to reuse previous computation and generate better proposals for the coming frames at the same time.

We make the approximate Bayesian inference with probabilistic programs, sampling $pg$s from the T-AoG and producing trajectories from $\delta$. At each time iteration, the algorithm performs two iterative steps:

**Stochastic filtering** predicts possible human actions by sampling from our generative program $\delta$ as defined in Eq. 4, and re-weights the samples by comparing the samples with newly observed motion trajectory. Traditional particle filtering requires a strong assumption on the Markov property along the time axis, which does not hold in our case as our grammar model encodes long-term relationships. Here we encode the posterior distribution in the form of production probabilities for each statement in the stochastic grammar. This representation allows us to encode long-term and complex relations between observed and predicted trajectories efficiently.

We perform re-weighting to improve the likelihood that the T-AoG will generate plans that match the observations using the loss computed by DTW. Let $\#(O_i^j)$ be the number of synthesized particles that go down path $i$ of OR-node $j$, and $\#(O^j)$ be the number of synthesized particles that reached OR-node $j$, and let $N$ be the total number of particles synthesized. Then by the law of large numbers,

$$\lim_{N\to\infty} \frac{\#(O_i^j)}{\#(O^j)} = \omega_i. \tag{5}$$

This equation provides the following update procedure for the weights for each OR-node.

Let $X_{\text{pred}}^k$ be the $k$th synthesized trajectory from the T-AoG. Compute the loss of each trajectory (denoted $X^k$) by calculating $P_{\omega^t}(X_{\text{obs}} \mid X_{\text{pred}}^k)$ from DTW. Here we make explicit that $P$ is parameterized over the production probabilities $\omega$ at a current time $t$. This loss is decomposable along each OR-node production; thus we can optimize over each OR-node individually to maximize the posterior probability of generating $X_{\text{obs}}$ by choosing $\omega_i^{t+1}$ such that the AoG at time $t+1$ prefers distributions that match $X_{\text{obs}}$:

$$\omega_i^{t+1} \propto \frac{\#(O_i^j)}{\#(O^j)} \times \prod_{k\in A} P_{\omega_i^t}(X_{\text{obs}} \mid X_{\text{pred}}^k), \tag{6}$$

where $A$ is the set of particles that contain OR-node $j$ that take branch $i$. Due to Eq. 5 this can be interpreted in the limit as $N \to \infty$ as rescaling $\omega_i^t$ by some factor corresponding to how well its particles predict the observed trajectory. The worse a particle performs (the lower $P_{\omega_i^t}(X_{\text{obs}} \mid X_{\text{pred}}^k)$), the more penalized its production probability is by the

update rule. Once we complete the sampling process, we re-normalize all the $\omega_i^{t+1}$ so that the sum of all transitions for each OR-node equals 1. In the event that no particles reach a particular production rule, the production probabilities from the previous time-step are propagated.

**Stochastic memoization.** In the next time frame, we sample anew from the learned $\omega_i^{t+1}$ from the previous time-frame in order to prioritize exploring more likely future goals. This process is known as *stochastic memoization*, outlined in [14], which records a posterior grammar of possible previous explanations according to the weighted samples from the previous time-step. This allows the algorithm to make use the previous posterior beliefs to generate better proposals for coming frames. We summarize the particles from the previous time step in the *posterior grammar*. The posterior grammar shares an identical structure with the prior grammar mentioned before, but has different branching probabilities due to the updating procedure in Eq. 6. We weight the updating of each OR branch based on (1) the number of particles that pass through that branch and (2) the error of the particle produced by that branch.

We summarize the stochastic inference algorithm in Algorithm 1.

**Data:** 3D Scene, Video Frames ($V$), Dictionary ($\Delta$)
**Result:** $\omega$ (Parameterized T-AoG)
CollisionMap = SCENERECONSTRUCTION($V$);
RRT* = RRT*Planner(CollisionMap);
P = Planner($\Delta$);
Particles = [];
**for** *Frame* $v_t$ *in* $V$ **do**
  ObservedTrajectory = ObjectTracking($v_1, \cdots, v_t$);
  PredictedPlan = Planner.sample(Particles);
  PredictedTrajectories = RRT*.search(PredictedPlan);
  Loss = DTW(ObservedTrajectory,
   PredictedTrajectory);
  Planner.reweight(Loss);
  Particles = PredictedTrajectories;
**end**
**return** Planner.weights

**Algorithm 1:** The stochastic inference algorithm for finding the most likely *pg* given a sequence of observed actions. Here the weights of the planner represent the MAP estimate of the T-AoG.

## III. EXPERIMENTS

The general goal of our experiments is to show (a) the effectiveness of our inference procedure and (b) how this inference procedure can be used to improve robot perception. We evaluate our algorithm on three tasks: intent prediction, action recognition and object tracking. Each of these tasks is essential for constructing a robot platform capable of interacting with humans in a meaningful way.

### A. Experimental Apparatus

Our experimental apparatus was comprised of an active sensing robotics platform which tracks human movements in an environment with an RGBD sensor as shown in Fig. 4. We transformed the point cloud (2.5D) into a voxel world (3D) and used it to recognize object affordance according to a recent approach by Zheng et al. [38]. Each voxel is a volumetric pixel whose color represents 3D height. We used a state-of-the-art RGBD tracking algorithm [15], [30] to detect and track 3D objects in the video frames. We tested our system in both a living room scene shown in Fig. 2 and an office scene in Fig. 4. We recorded about 1 hour RGBD video for the experiments. The dataset details are reported in Table I.

TABLE I
SUMMARY OF OUR DATASET

| # frames | # clips | # actions | # objects | # tracking BBs |
|---|---|---|---|---|
| 13981 | 31 | 29 | 34 | 116261 |

### B. Intent Prediction

The intent prediction is made very challenging by the goal ambiguity problem: there are 30 candidate goals for any given frame. We presented the count of goal objects as they appear in the video. Some of the objects are very close to each other as shown in Fig. 4, so it is almost impossible to identify all the goals given noisy tracking trajectories. We compare our algorithm to three baseline algorithms on the intent prediction task. The result is presented in Table II; the "% of observation" row shows what percent of the observed trajectory was presented to each model. We evaluate the four algorithms given different percentages of observations, i.e. trajectories with different lengths from the goal.

The first baseline is the Euclidean distance from current human position to the goal. This baseline is quite simplistic; randomly guessing has a 3% chance of being correct, while the baseline is even worse since it prefers nearby goals.

The second baseline is the Euclidean distance combined with a grammar prior, which is a substantial improvement on the first baseline. The prior uses knowledge of action sequences to weight the intents, which leads to higher overall accuracy. This baseline is quite similar to a recent work by [27] on action prediction, which built a grammar in which terminal nodes are defined by Euclidean distances.

The third baseline is an algorithm based on inverse planing without hierarchy. Baker et. al.'s work [3], [2], [4] and Kitani et. al.'s work [20] are similar to this baseline. However, their approaches are based on variants of Markov decision process (MDP) which only are demonstrated for low-dimensional discrete space. We implemented our planner based on RRT*, which is designed for fast optimization in high-dimensional continuous spaces, such as the human body movements. We use a limited planning dictionary with 29 terminal nodes. We ultimately measure the percentage of frames in which the most likely *pg* matches the ground truth *pg*.

### C. Action Recognition

We evaluate our action recognition algorithm by observing a sequence of action before and after the goal changing

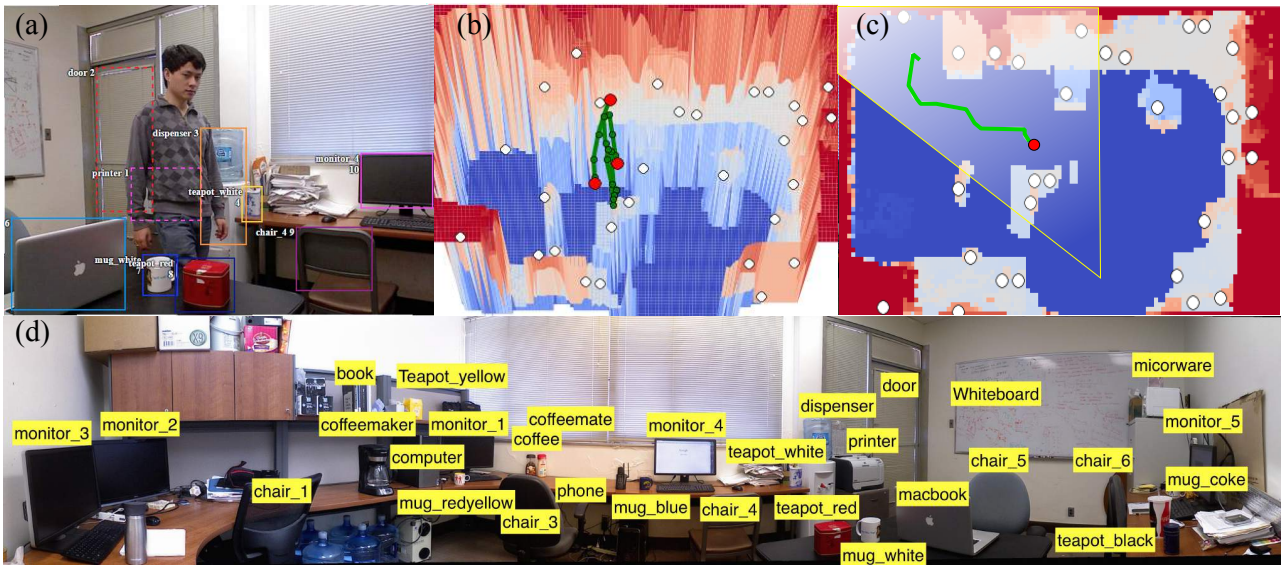


Fig. 4. (a) Observed RGBD video with ground truth labels of object bounding boxes and goals for each frame. The RGBD video is collected by a robotics system which can automatic track the agent's movement. At the beginning of each video clip, the robot stitches the scene together. We show in (d) the manually labeled objects in the stitched panoramic scene, which is used as an initialization for object state tracking. In (b,c) we see how the labeled objects are mapped to the 3D reconstructed environment for each frame, and how these skeletons are projected into a 3D and 2D space.

TABLE II
INTENT PREDICTION ACCURACY

| % of observation | 90% | 70% | 30% | 10% |
|---|---|---|---|---|
| Euclidean Distance (E.D.) | 5% | 0% | 0% | 0% |
| E.D. w/ Grammar Prior | 13.1% | 10.5% | 6% | 3% |
| Inverse Planning w/o Hierarchy | 15.8% | 13.2% | 10.5% | 10.5% |
| Ours | 28.9% | 13.3% | 18.4% | 15.8% |

TABLE III
ACTION RECOGNITION ACCURACY

| | SVM | ICCV13 [36] | Ours |
|---|---|---|---|
| walk | 88% | 98% | 91% |
| stand_up | 68% | 94% | 92% |
| sit_down | 65% | 92% | 92% |
| grasp | 43% | 64% | 59% |
| put | 25% | 44% | 53% |
| fetch | 33% | 54% | 83% |
| touch | 35% | 41% | 54% |
| drink | 70% | 91% | 91% |
| call | 65% | 89% | 94% |
| eat | 22% | 54% | 73% |

states. For example, we observe 3 seconds before and after the "grasp" action. We compare action recognition of terminal actions (operators) with an SVM as a baseline and a state-of-the-art algorithm [36] with similar interacting action categories. Our algorithm outperforms the other algorithms on the task of jointly inferring actions and object states as shown in Table III.

### D. Object Tracking

We manually labeled 116,261 bounding boxes (BBs) for evaluating object tracking by Vatic [34]. Each bounding box has an occlusion label. We are able to recover the states of the scene from the state of the terminal nodes of the most likely AoG by assuming the actor has an ideal observation and understanding of the scene. We evaluate the tracking of objects with a state-of-the-art RGB tracking algorithm [15] and RGBD tracking algorithm [30]. The tracking problem is very challenging due to heavy occlusions and camera movement. To compensate for this, we evaluate tracking accuracy by checking whether or not two bounding boxes have 10% overlap with each other. The results in Table IV demonstrate that our algorithm not only improves the tracking result for non-occluded objects, but also successfully tracks objects when they are occluded or interacting with actors. Our errors primarily result from misalignment between 2D images, 3D scenes, and skeletons. The algorithm can successfully identify the relations, such as there being a mug in the actor's hand, but occasionally fails to locate the exact object position due to noisy hand tracking data.

TABLE IV
OBJECT TRACKING ACCURACY

| | ICCV11[15] | ICCV13[30] | Ours |
|---|---|---|---|
| No Occlusion | 34% | 72% | 74% |
| With Occlusion | - | - | 35% |
| All Frames | - | - | 65% |

## IV. CONCLUSION

Inferring an agent's hierarchical objective is an essential tool for creating systems involving human-robot interaction. A robot must be able to understand a human's actions: we propose to do this by inverting the planning process of a human [2]. By imagining a human's plans, a robot can truly understand human intent and motivations.

Our implementation seeks to make plan inference computationally tractable without limiting the expressiveness of

human intent. We focus our search space based on what we have observed, and model our predictions based on past observations, without limiting ourselves to plans of a specific depth or with limited long-term relationships.

Our work could be improved by incorporating better object recognition. In its current iteration, we manually labeled all the objects in the 3D scene. Understanding additional aspects of scene dynamics, by using a method such as [38], could help to disambiguate cases where scene dynamics confound observations of intentional human actions. In addition, we use a relatively naive Euclidean distance cost function for our rational low-level planner (RRT*). The synthesized trajectories could be made more realistic by incorporating a more sophisticated low-level rational planner, such as one learned through reinforcement learning [7]. A further limitation is that the planning dictionary must be provided a-priori; it would be useful for a robot to be able to learn a planning dictionary through passive observations with only the ability to detect terminal actions. An example of such a system is outlined in [29].

## V. Acknowledgments

The authors would like to thank Yixin Zhu and three anonymous reviewers for their helpful feedback and review of our drafts. Steven Holtzen is supported by a National Physical Sciences Consortium fellowship through Sandia National Laboratories. This work is supported by the DARPA SIMPLEX Award N66001-15-C-4035 and ONR MURI grant N00014-16-1-2007.

## References

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

[2] C.L. Baker, R. Saxe, and J.B. Tenenbaum. Action understanding as inverse planning. *Cognition. 2009 Dec*, 113:329–349, 2009.

[3] C.L. Baker and J.B. Tenenbaum. Modeling human plan recognition using bayesian theory of mind. In G.G. Sukthankar, R.P. Goldman, C. Geib, D. Pynadath, and H.H. Bui, editors, *Plan, Activity, and Intent Recognition*. Elsevier, 2014.

[4] C.L. Baker, J.B. Tenenbaum, and R.R. Saxe. Bayesian models of human action understanding. In *NIPS*, volume 18, pages 99–106, 2006.

[5] H.H. Bui. A general model for online probabilistic plan recognition. *International Joint Conference on Artificial Intelligence*, pages 1309–1318, 2003.

[6] H.H. Bui, S. Venkatesh, and G. West. Policy recognition in the abstract hidden markov model. *Journal of Artificial Intelligence Research*, 17:451–499, 2002.

[7] A. Byravan, M. Monfort, B. Ziebart, B. Boots, and D. Fox. Graph-Based Inverse Optimal Control for Robot Manipulation. (Ijcai):1874–1880, 2015.

[8] E. Charniak and R.P. Goldman. A Bayesian Model of Plan Recognition. *Artif. Intell.*, 64(1):53–79, nov 1993.

[9] K. Erol, J. Hendler, and D. Nau. Htn planning: Complexity and expressivity. In *AAAI*, 1994.

[10] C.M. Geib and R.P. Goldman. A probabilistic plan recognition algorithm based on plan tree grammars. *Artificial Intelligence*, 173(11):1101–1132, 2009.

[11] G. Gergely, H. Bekkering, and I. Kirly. Rational imitation in preverbal infants. *Nature*, 415:755, 2002.

[12] R.P. Goldman, C.W. Geib, and C.A. Miller. A New Model of Plan Recognition. *Artificial Intelligence*, (July):245–254, 1999.

[13] N.D. Goodman, V.K. Mansinghka, D.M. Roy, K. Bonawitz, and J.B. Tenenbaum. Church: A language for generative models. In *UAI*, pages 220–229, 2008.

[14] N.D. Goodman and J.B. Tenenbaum. *Probabilistic Models of Cognition (electronic)*. Retrieved 07/2016.

[15] S. Hare, A. Saffari, and P.H.S. Torr. Struck: Structured output tracking with kernels. In *ICCV*, 2011.

[16] M. Isard and A. Blake. Condensation – conditional density propagation for visual tracking. In *International Journal of Computer Vision*, volume 29(1), pages 5–28, 1998.

[17] Richard K., Alireza T., C. King, A. Ambardekar, L. Wigand, M. Nicolescu, and M. Nicolescu. *Intent Recognition for HumanRobot Interaction*. Elsevier Inc., 2014.

[18] S. Karaman, M. Walter, A. Perez, E. Frazzoli, and S. Teller. Real-time motion planning using the RRT*. In *ICRA*, April 2011.

[19] H. Kautz and J Allen. Generalized Plan Recognition. *In Proceedings of the 5th National Conference on Artificial Intelligence*, pages 32–37, 1986.

[20] K.M. Kitani, B.D. Ziebart, J.A. Bagnell, and M. Hebert. Activity forecasting. In *ECCV*, 2013.

[21] L. Liao, D. Fox, and H. Kautz. Learning and Predicting Transportation Routines. *Artificial Intelligence*, 2007.

[22] Luiz L.S.H. and A.C. Sanderson. AND/OR graph representation of assembly plans. *IEEE Transactions on Robotics and Automation*, 6(2):188–199, 1990.

[23] R. Nakahashi, C.L. Baker, and J.B. Tenenbaum. Modeling Human Understanding of Complex Intentional Action with a Bayesian Nonparametric Subgoal Model. *CoRR*, 2015.

[24] A.Y. Ng and S. Russell. Algorithms for inverse reinforcement learning. In *ICML*, pages 663–670, 2000.

[25] N.T. Nguyen, A. Grzech, R.J. Howlett, and L.C. Jain. Expressivity of strips-like and htn-like planning. In *KES-AMSTA*, volume 4496 of *Lecture Notes in Computer Science*, pages 121–130. Springer, 2007.

[26] N.T. Nguyen, D.Q. Phung, S. Venkatesh, and H.H. Bui. Learning and Detecting Activites from Movement Trajectories Using the Hierarchical Hidden Markov Model. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 18(1):955–960, 2005.

[27] M. Pei, Y. Jia, and S.C. Zhu. Parsing video events with goal inference intent prediction. In *ICCV*, 2011.

[28] D.V. Pynadath and M.P. Wellman. Probabilistic state-dependent grammars for plan recognition. *Uncertainty in Artificial Intelligence Proceedings*, pages 507–514, 2000.

[29] Z. Si, M. Pei, B. Yao, and S.C. Zhu. Unsupervised learning of event and-or grammar and semantics from video. In *ICCV*, pages 41–48, 2011.

[30] S. Song and J. Xiao. Tracking revisited using rgbd camera: Unified benchmark and baselines. In *ICCV*, 2013.

[31] A. Stuhlmüller and N.D. Goodman. Reasoning about reasoning by nested conditioning: Modeling theory of mind with probabilistic programs. 2013.

[32] K. Talamadupula, G. Briggs, T. Chakraborti, M. Scheutz, and S. Kambhampati. Coordination in human-robot teams using mental modeling and plan recognition. *IEEE International Conference on Intelligent Robots and Systems*, (Iros):2957–2962, 2014.

[33] T.K. Vintsyuk. Speech discrimination by dynamic programming. *Cybernetics*, 4:52–57, 1968.

[34] C. Vondrick, D. Patterson, and D. Ramanan. Efficiently scaling up crowdsourced video annotation. *International Journal of Computer Vision*, pages 1–21, 2012.

[35] Z. Wang, M.P. Deisenroth, H.B. Amor, D. Vogt, B. Schölkopf, and J. Peters. Probabilistic Modeling of Human Movements for Intention Inference. *Robotics: Science and Systems (RSS)*, 32(2):1–8, 2012.

[36] P. Wei, N. Zheng, Y. Zhao, and S.C. Zhu. Concurrent action detection with structural prediction. In *ICCV*, pages 3136–3143, 2013.

[37] D. Xie, S. Todorovic, and S.C. Zhu. Inferring dark matter and dark energy from videos. In *The IEEE International Conference on Computer Vision (ICCV)*, December 2013.

[38] B. Zheng, Y. Zhao, J.C. Yu, K. Ikeuchi, and S.C. Zhu. Beyond point clouds: Scene understanding by reasoning geometry and physics. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 3127–3134, 2013.

[39] S.C. Zhu and D. Mumford. A stochastic grammar of images. *Foundations and Trends in Computer Graphics and Vision*, 2, 2006.

[40] B.D. Ziebart, N. Ratliff, G. Gallagher, C. Mertz, K. Peterson, J.A. Bagnell, M. Hebert, A.K. Dey, and S. Srinivasa. Planning-based prediction for pedestrians. In *IROS*, 2009.